Rob Pooley · Jennifer Coady
Henry Linger · Chris Barry
Michael Lang · Christoph Schneider *Editors*

Information Systems Development

Reflections, Challenges and New Directions

Springer

Information Systems Development

Rob Pooley • Jennifer Coady
Henry Linger • Chris Barry • Michael Lang
Christoph Schneider
Editors

Information Systems Development

Reflections, Challenges and New Directions

Springer

Editors
Rob Pooley
Department of Computer Science
Heriot-Watt University Edinburgh
Edinburgh, UK

Jennifer Coady
Department of Computer Science
Heriot-Watt University Edinburgh
Edinburgh, UK

Henry Linger
Faculty of Information Technology
Monash University
Clayton, VIC, Australia

Chris Barry
National University of Ireland
Galway, Ireland

Michael Lang
National University of Ireland
Galway, Ireland

Christoph Schneider
City University of Hong Kong
Kowloon, Hong Kong SAR

ISBN 978-1-4614-4950-8 ISBN 978-1-4614-4951-5 (eBook)
DOI 10.1007/978-1-4614-4951-5
Springer New York Heidelberg Dordrecht London

Library of Congress Control Number: 2013933239

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

Information Systems Development (ISD) has reached its twentieth anniversary, with the 2011 conference being held at Heriot-Watt University in Edinburgh. As ever, the range of papers and presentations is remarkable, but all have in common the need for higher quality and more reliable information systems to support our increasingly data hungry world.

As well as papers ranging across current issues surrounding the design and implementation of high-performance systems, keynote presentations from Nick Taylor, of Heriot-Watt University, discussing the issues of privacy and personalisation in a world of pervasive systems, and from Ian Somerville of St Andrews University – on the sociotechnical and political factors underpinning successful developments – gave those attending much food for thought. We are lucky that Nick's presentation has been captured in an invited paper.

Thus, we see that ISD remains relevant and challenging. Those attending found much to fuel their imaginations and to spark fresh ideas. Of the 93 papers submitted, 54 were accepted, showing that ISD maintains its high standards.

The selection process was managed using Easy Chair, which lives up to its name. Thanks are also due to all those on the Programme Committee who provided such careful reviews.

The organisers, Professor Rob Pooley, Jenny Coady and Tessa Berg, hope everyone enjoyed and benefitted from all that was offered and pass on their good wishes to next year's team, in Prato, Italy.

Edinburgh, UK

Rob Pooley, Programme Chair
Jennifer Coady, General Chair
Tessa Berg, Local Chair

Contents

Chapter 1
Is There Really a Conflict Between Privacy and Personalisation?

Nicholas K. Taylor, Elizabeth Papadopoulou, Sarah Gallacher, and Howard M. Williams

1 Introduction

In 1991, Mark Weiser described his vision of ubiquitous computing, a world in which technology aids the user unobtrusively in their everyday life (Weiser 1991). Twenty years later, the world is closer to achieving Weiser's dream, but with every innovation comes consequences. For ubiquitous or pervasive system to function effectively and weave itself into the user's everyday life, it needs a wealth of information about that user. Automated systems are generic and offer the same automated functionality for all users, whereas a pervasive system attempts to automate processes on behalf of an individual user. To accomplish such task, a pervasive system requires the possession, processing and inference of knowledge about the user such as the user's preferences, interests, goals, intents, environmental (context information) and personal information. While the benefit of pervasive systems has been defined, the consequences to personal privacy when disclosing large amounts of such information are among the reasons this technology has not taken over our reality. Hence, a divide has been building between those that favour the proliferation of information and those that oppose it for fear of eliminating the right to personal privacy.

In theory, personalisation and privacy are at odds with each other. The more information there is about a user, the better the system can adapt to the user's needs, and if no information exists about a user, all personal privacy is protected beyond any doubt. In reality, people share information everyday with other people, organisations, companies, schools, hospitals and virtually any entity they interact with without worrying about their personal privacy. The people choose what information to disclose and whom to disclose it to, they know the reasons for

N.K. Taylor (✉) • E. Papadopoulou • S. Gallacher • H.M. Williams
PUMA Lab, Heriot-Watt University, Riccarton Campus, Edinburgh EH14 4AS, UK
e-mail: N.K.Taylor@hw.ac.uk; E.Papadopoulou@hw.ac.uk; S.Gallacher@hw.ac.uk; M.H.Williams@hw.ac.uk

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_1,

disclosing it, they decide what level of detail to disclose, and they choose when to disclose it. Hence, a pattern can be observed, that people tailor their privacy to suit their own needs. Such practices are analogous to what personalisation provides for services. Personalisation and specifically user preferences can be used to customise how the system discloses information. More significantly, it can relieve the user from the burden of editing privacy settings for every entity they interact with by automating this process based on user preferences.

The next chapter provides an overview of different forms and applications of personalisation. Chapter 3 describes the attitudes of services towards respecting the user's privacy and the attitudes of users with respect to protecting their own privacy. Chapter 4 presents an approach to privacy protection using personalisation, and Chap. 5 illustrates an example of this approach in an implemented prototype. Chapter 6 concludes with a discussion of future work and possible enhancements.

2 Personalisation

Most of the published research papers regarding personalisation refer to the personalisation or customisation of web services or mobile services, specifically addressing personalised service composition (Jørstad et al. 2004). In general, the term personalisation refers to the adaptation of a system according to the needs and preferences of an individual user. In most cases, personalisation systems use rule models and rule engines to represent the user's preferences and evaluate them according to changes in the user's context. User preferences can be used to personalise a system in the following ways.

- Service Ranking and Filtering. The results of a service discovery query can be filtered and ranked based on user preferences and user context. Service filtering can be personalised to discard services that are not preferable to the user, and service ranking can be personalised to sort the list of services-based personalised parameters such as distance, cost, quality and provider preference.
- Service Selection. User preferences can suggest specific services to be selected for particular uses and in certain situations. A simple example of personalised service selection would be using Google Search as the preferred search engine.
- Service Management. User preferences can be used to represent rules for service initialisation, termination and session adaptation. A simple example of personalised service management is to turn on the air-conditioning service when a user enters their office building.
- Service Adaptation. The most common use of user preferences is to personalise the parameters of a service. This can range from setting the wallpaper of a user's desktop to adapting the heating temperature of a room based on activity in the room.
- Proactive Personalisation. Pervasive systems are dependent on the availability of information about the user's context. Context-aware user preferences can be

evaluated during runtime and adapt services and the system to perform differently under specific context conditions. A simple example of proactive personalisation is muting the mobile phone when a user enters a meeting room.

Hence, user preferences can drive a system to function in almost every area according to the user's preferences. However, as the number of used services multiplies over time, it is not reasonable to expect the user to manually enter preferences for every service they use and manage these large structures themselves.

2.1 Degrees of Personalisation

The extent to which a system is adapted by personalisation components depends on the type of user preferences used and the availability of information in the system. The less information that exists, the less impact the personalisation components have on the system. The degrees of personalisation a system can exhibit can be split into three categories.

Static Personalisation. A system that applies static personalisation uses simple static rules or simply static settings to configure the system. This type of user preference is applied once in a service and does not change unless the user manually edits the preference. Many if not all applications today employ some form of static personalisation by providing a graphical user interface in which the user is able to change the settings for the appearance or behaviour of the application.

Dynamic Personalisation. A system that applies dynamic personalisation uses context-dependent user preferences. A context-dependent user preference implies that under different context conditions, the system will behave in a different way. A dynamic personalisation component has to evaluate the user preferences against the current context of the user and apply the corresponding preference outcome. Depending on the complexity of the user preference and the available context sources in the system, a service can be constantly personalised to fit the needs of the user. Context-dependent user preferences can be very useful as they adapt the services and the system based on changes in the environment the user is in and the activities that the user is involved in.

Proactive Personalisation. This is the automated type of personalisation in which the system is constantly monitoring the context of the user and evaluates preferences affected by changes in that context. A system that applies proactive personalisation is automating the behaviour of the system. Services do not need to request preference information explicitly. When the context of the user changes, the system evaluates the preferences and re-personalises the services immediately. Proactive personalisation is the most effective type as the personalisation of the services is instant. In many cases, this is very important. An example of proactive personalisation is turning on and off the heating in a house or an office by monitoring the user's context such as the location and the time of day.

Table 1.1 Respondents' level of concern with respect to privacy

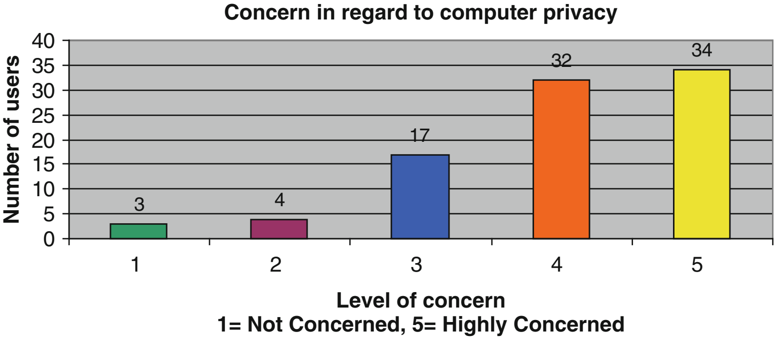
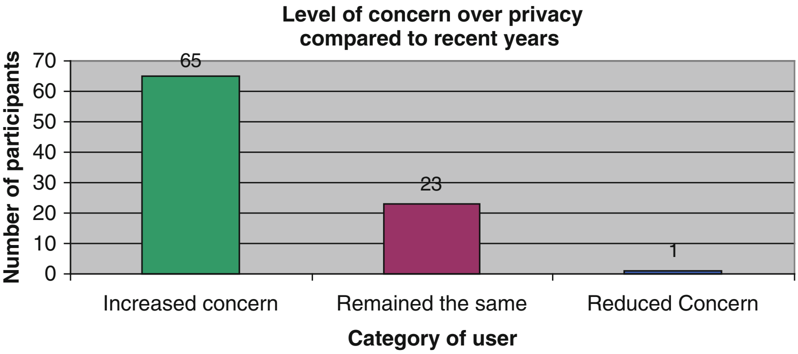


Table 1.2 Respondents' change in level of concern with respect to privacy

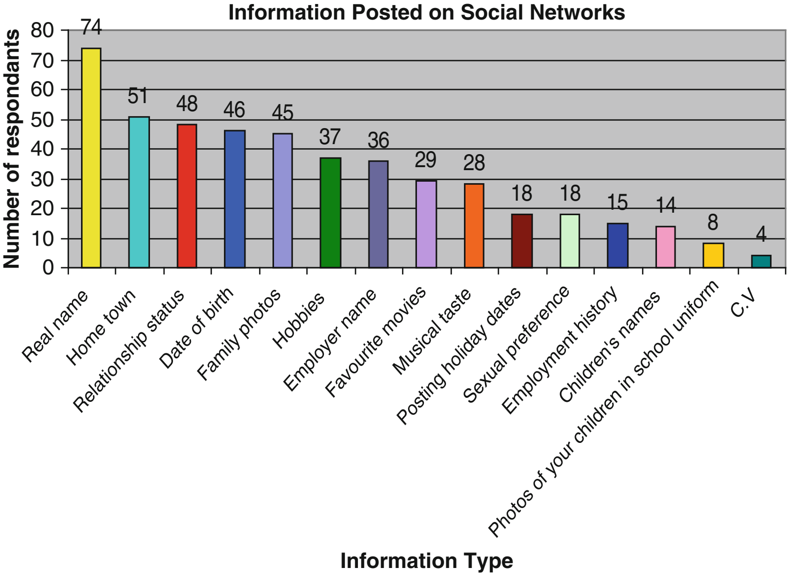


3 Attitudes Towards Privacy

Issues of privacy violation, personal data loss and security attacks often make the news headlines and incur the public's anger towards the practices of large corporations and governments regarding the handling of their personal data. Towards the end of 2010 through to the start of 2011, Simpson (2011) conducted a survey to obtain a snapshot of computer users' current views on data protection and their privacy practices. There were 90 participants and the demographic covered an age range from 16 to 60 and occupations which, while approximately 50 % academic, also included clerical workers, nurses and even a beauty therapist and a bus driver.

The respondents to the survey expressed significant concern about computer privacy, (see Table 1.1) and this concern has been increasing in recent years as a result of the increasing demands for information by websites and social networks (see Table 1.2).

Table 1.3 Respondents' disclosures on social networking sites

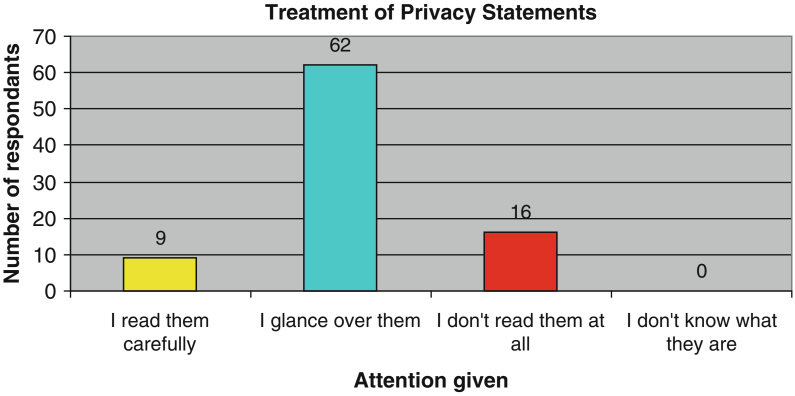


However, when asked what types of information they were prepared to disclose on social networks, Simpson's respondents appear to put their privacy concerns to one side (see Table 1.3).

One of the factors influencing this trend is the lack of privacy protection mechanisms for use by the average user. Current laws, such as the EU Directive 95/46/EC (EU Data Protection Directive 95/46/EC 1995) and the UK Data Protection Act of 1998 (Data Protection Act 1998), have defined the right of users to personal privacy and require service providers (referred to as data controllers) to provide privacy policies that outline among other things, the types of data collected, the kind of processing to be applied to the data and the purpose for which the data is collected. Users must agree to the terms and conditions of a service which implies that they have read and agreed to the privacy policy provided by the service. However, Simpson's, and other, surveys have repeatedly shown that users do not assess privacy policies in a rational manner before agreeing to the terms and conditions of services (see Table 1.4).

Despite the media attention given to privacy violations and security breaches, there appears to be a worrying acceptance on the part of the public that these things are inevitable in the world of digital information. This is probably because users have little choice; if they wish to use a service, such as a popular social network site, then they have no option but to accept the service's privacy policy. Foregoing use of the service because of privacy concerns would not appear to be a realistic option for the vast majority of users.

Table 1.4 Respondents' attitudes to privacy statements



4 Privacy Protection Using Personalisation

It is believed that the more information exists in the system, the better the personalisation. This is true in a generic way. A system can make better decisions if it has all the necessary information, and in order to ensure this in every situation, the only way to do this is to include all the information. However, this does not mean that the personalisation system and the information it uses such as profile information, user preferences and user context has to be controlled by an external entity. Such systems should be centred on the user and any automatic behaviour must be controlled by the user using user preferences as the driver. Advertising and product marketing have caused personalisation to be misunderstood as their purpose is to target users with relevant products by monitoring the users' activity on the Web such as what websites they visit and what products they purchase online. This type of personalisation may be beneficial in some cases since users will see advertisements that they are interested in. However, users are using online services for business use as well as personal use, and it is not always appropriate to link this information together. It will be beneficial to users as well as service providers that the information is disclosed in a controlled manner. The user's privacy is protected viably, and the service providers can offer quality of service as well as receive the reputation of respecting its users' privacy.

The approach presented in this chapter attempts to provide the user with the necessary tools to protect their personal privacy with minimum hassle.

4.1 Personalised Identity Selection

Users employ different personas depending on the situation they are in and the entities with which they interact with. As a simple example, consider a person's behaviour when they are in an office surrounding compared to being in their home

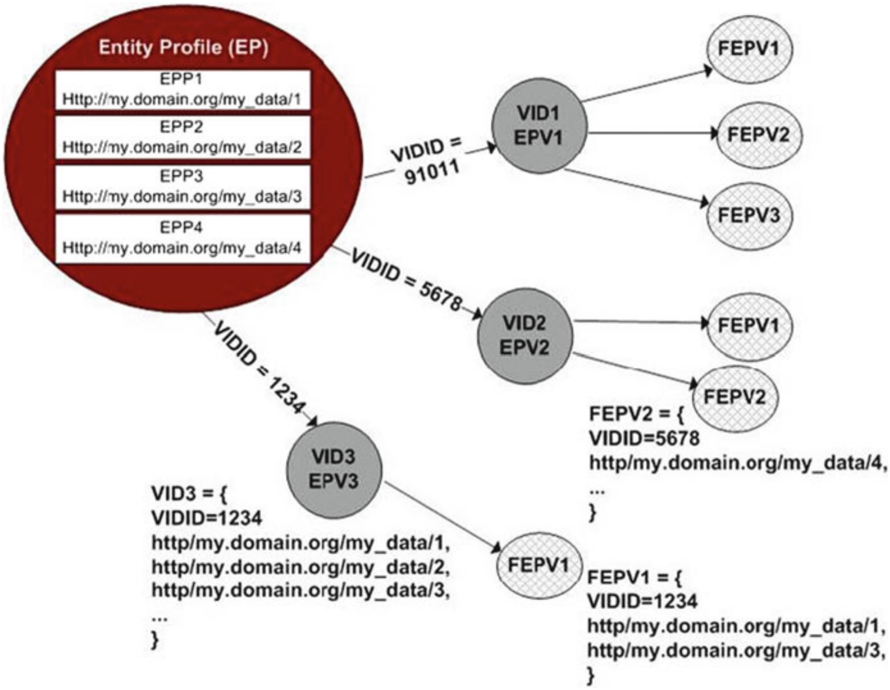


Fig. 1.1 Associating personal data attributes with multiple identities. The Daidalos Virtual model approach

with their family and friends. Supplying the user with mechanisms that support the use of multiple digital identities will have a positive effect on the manner in which personalisation is conducted as well as how privacy is protected. A digital identity can be considered as a digital representation of a user. Identity management is a large research area, and many systems have been implemented such as OpenID (The OpenID Foundation Website), Microsoft's Cardspace (Microsoft Cardspace) and LiveID (Microsoft LiveID), Shibboleth (The Shibboleth System) and the Daidalos Virtual Identities (Girao et al. 2006). All these schemes are centred around the idea of the federated identity. Federated identities can be provided by multiple identity providers and can be used to represent the user when they interact with online services. Using multiple federated identities, the user can link different personal information such as profile information, user preferences and user context to different identities and represent themselves differently to the entities they interact with. The benefit of federated identities is that an identity can be reused in different services and the information collected by one service can be reused by another service.

From the point of view of the user, a tool is required to manage the user's different identities and to suggest to the user which identity should be used in each interaction. The example in Fig. 1.1 shows how multiple identities are linked to attributes from the user's profile data using the Daidalos Virtual Identity model. Each attribute has a different URI for addressing it. When an attribute is linked to more than one identity, there is a different URI constructed for it for each identity that it is linked to.

User preferences can be used to personalise the use of multiple identities. A graphical user interface must exist for the user to control the use of multiple identities by creating user preferences to drive the identity selection. Also, the system can log every transaction each identity has been used in and automatically create preferences based on the type of transaction and the type of data disclosed during the transaction.

4.2 Personalised Privacy Policy Negotiation and Access Control

The privacy policy negotiation process allows users to negotiate the terms and conditions of a service and choose what data to disclose with each service. Personalisation can play an important part in the automation of this process by allowing the user to create privacy preferences for each data attribute that exists in their profile and equipping the system with the tools necessary to protect the user's privacy in a proactive manner. Current approaches to privacy policy negotiation suggest that the user creates their privacy policy documents manually, and based on these documents, the system is able to negotiate about the privacy policy of the service. It is not reasonable to expect the average user to have the knowledge required to create such documents themselves. The proposed approach attempts to alleviate the user from such burden by suggesting that users only create privacy preferences for specific attributes of data. The system will then evaluate these privacy preferences against the current context and create a privacy policy for the user customised as a response to the privacy policy of the service. Context-aware privacy preferences provide the user with greater flexibility in their choices. Different access control rules can be enforced based on the context of the user and the service or entity that requests access to the data. For example, the user can allow their employer to access their location during office hours and block access outside office hours. Depending on the level intelligence built into the personalisation system, the system can perform certain functions on behalf of the user to aid them. Based on the user's disclosure practices, the system could infer that certain attributes are more sensitive than other attributes. The use of a trust management system that attempts to rate the level of trustworthiness of entities such as service providers can give greater flexibility in a privacy protection framework. As well as context dependence, privacy preferences can include trust conditions for data disclosure. Depending on the level of trust the user has in the ability of a service to honour the privacy policy agreement and respect the personal privacy of its users, the system can assess the level of data access to grant to a service.

5 Conclusion

Personalisation and privacy are always considered as conflicting with each other. Personalisation needs information to function properly while privacy attempts to hide information to protect it from misuse. However, personalisation systems can coexist in harmony with privacy protection systems. Personalisation can aid privacy protection systems to protect the user's privacy in the manner the user wishes to do so. It should be noted that users have different views on how their privacy should be protected and what data they prefer to be disclosed to certain people and what data should remain private. Personalisation can play an important role in enforcing the user's wishes with regard to their privacy.

Acknowledgments The authors wish to acknowledge the contributions made by the many partners in the Daidalos, Persist and Societies projects and the European Commission in funding those projects. The views expressed here are those of the authors alone however.

References

EU Data Protection Directive 95/46/EC (1995). http://ec.europa.eu/justice/policies/privacy/docs/95-46-ce/dir1995-46_part1_en.pdf

Girao J, Sarma A, Aguiar R (2006) Virtual identities – a cross layer approach to identity and identity management. Presentation, 17th wireless world research forum, Heidelberg, Germany, November 2006

Jørstad I, van Do T, Dustdar S (2004) Personalisation of future mobile services. In: 9th international conference on intelligence in service delivery networks, Bordeaux, France

Microsoft Cardspace. http://msdn.microsoft.com/en-us/library/aa480189.aspx

Microsoft LiveID. https://accountservices.passport.net/ppnetworkhome.srf?lc=2057&mkt=EN-GB

Simpson S (2011) Issues in data protection and individual privacy. B.Sc. Information Systems Honours dissertation, Heriot-Watt University, Edinburgh, UK

The OpenID Foundation Website. http://openid.net/

The Shibboleth System. http://shibboleth.internet2.edu/

UK Data Protection Act (1998). http://www.legislation.gov.uk/ukpga/1998/29

Weiser M (1991) The computer for the twenty-first century. Sci Am 265(3):94–104

Chapter 2
Identify and Classify the Critical Success Factors for a Successful Process Deployment

Bayona Sussy, Calvo Manzano Jose, Cuevas Gonzalo, and San Feliu Tomás

1 Introduction

Nowadays, organizations need to respond to customer' demands with quality products and services. Models and standards have been developed to help organizations to achieve these objectives, such as Capability Maturity Model Integration (CMMI) and IDEAL. This highlights the importance of having an effective process deployment strategy in order that processes can be adopted, used, and institutionalized.

However, the implementation of these models and standards in organizations presents difficulties that include (1) improvement efforts are not aligned with business goals, (2) lack of leadership and visible commitment to improvement efforts, (3) the process does not respond to business needs, and (4) efforts to implement technical aspects ignore strategies based on the social aspects (Messnarz et al. 2008). According to Niazi (Niazi et al. 2005), the problem of process improvement is not the lack of standards or models, but the lack of a strategy to implement them. Not considering the social aspects of a strategy for process deployment threatens the institutionalization of the processes deployed.

Most researchers are focused on improving the technology. However, a few of them mention other important factors such as culture, change management, people, communication, and training. McDermid and Bennet (1999) have argued that human factors for software process improvement have been ignored. According to Zahran (1998), the inadequacy of proposals on the implementation of process improvement is one of the most common reasons for failure of improvement initiatives.

B. Sussy (✉) • C.M. Jose • C. Gonzalo • S.F. Tomás
Languages and Informatics Systems and Software Engineering Department,
Faculty of Computer Science, Polytechnic University of Madrid,
Campus Montegancedo, Boadilla del Monte, Madrid 28660, Spain
e-mail: sbayona@hotmail.com; gcuevas@fi.upm.es; tsanfe@fi.upm.es

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_2,

Some issues are present when processes are deployed such as (1) the difficulty to identify the difference between implementation and deployment; (2) the human factors are ignored or are only focus on technical aspects; (3) the process deployment is a change, but it is not managed to minimize the change resistance; (4) the staff do not participate in the process definitions; (5) the processes are not suitable to the organization's needs, environment, or culture; (6) the processes deployed are not used; (7) the organization does not have a formal deployment methodology; or (8) the method used is not focused on human factors.

These issues are already well known both in academic and industrial context, but they are present in process deployment. The goal of this chapter is to present a procedure to identify the critical success factors of process deployment and also to present a case study about impact of the critical success factors identified in five organizations and a proposal of a process deployment method. This method is focused on the critical success factors for process deployment. The reference model used is CMMI (2006).

This chapter is organized as follows: Section 2 shows the research context. Section 3 presents the critical success factors for process deployment. Section 4 described the case study. Section 5 describes the proposed method "Method for Process Deployment in CMMI level 3 organizations" (MEDEPRO). Finally, the conclusions are presented.

2 The Research Context

The International Process Research Consortium (IPRC) has included the topic of process deployment in a list of research items. A reason is that process deployment is related to the person, and an intensive research into the human factor and change management is needed (Quinn 1999).

The successful implementation of a process instance establishes its basic functionality; its effective deployment establishes its true value to the implementing organization.

The purpose of process deployment is about getting people to use the new processes. It is frequent that an organization uses the term *process implementation* and not *process deployment*. There is a difference between the implementation and deployment concepts.

International Process Research Consortium (IPRC) (2006) indicates that "It is important to recognize the critical difference between process implementation and process deployment. The concept of deployment goes beyond the single instantiation of an implemented process, to address the effective deployment of a process specification to achieve multiple implementations across an organization, each tailored to suit its specific organizational context."

Process deployment is focused on people, and it incorporates topics like:

- Education and training to develop collective competencies and abilities
- Staff motivation

Fig. 2.1 Process deployment elements

- Actual usage of a developed or standardized process into organization's projects and operations
- Metrics of process use and adoption
- Appraisal of the deployed processes to determine their fidelity and capability as well as to evaluate whether the usage has achieved the goals of adding value added by the process or product

According to our research work, process deployment elements are organization, Process Asset Library, processes, people, process deployment methodology, deployment process, reference model, and change management. The organization has a vision defined, values, culture, policies, and business objectives. Figure 2.1 shows the process deployment elements.

Each element is described next.

- *Organization*. IT is the place where the new processes will be deployed (including the organizational and functional structure). Every organization has a vision, mission, business objectives, values, and culture organizational.
- *Process Asset Library (PAL)*. It is a repository for process descriptions and assets, such as process adaptation guide, templates, checklists, and metrics. This repository may be updated using baseline, to have feedback mechanisms and give access to project team.
- *Processes*. A process is a goal-directed, interrelated series of actions, events, mechanisms, or steps that are constituted by (1) purpose, (2) roles and responsibilities of people in order to carry out their work, (3) entry and exit criteria, (4) inputs and outputs, (5) procedures and methods that define how to carry out the tasks and their relationship (activities/steps), and (6) measures. Processes are supported by the tools and equipment that allow them to perform their work.

- *People*. With the required knowledge, skills, and attitudes, tasks can be carried out satisfactorily and thereby ensure that all activities are completed. Aspects like people capacity and competence, teamwork, alignment to the organizational vision, process improvement proposals, participation of those involved (Christiansen and Johansen 2007), training, and communication are all key factors to develop human resources (Constantine 2001).
- *Process Deployment Methodology*. It establishes the guidelines, the procedures, and the rules to deploy the processes. It contains the activities that should be developed for training and motivating people in the new processes and the communication strategies. Also, it establishes deployment roles and responsibilities.
- *Deployment Process*. It consists on selecting a process deployment strategy where information, communication, training, and evaluation are the principal items of this strategy. An important aspect is the people's disposition to continue and apply the processes to their daily work, in such a way to assure that the processes are continued.
- *Reference Model*. It is a model of something that embodies the basic idea of something and can then be looked at as a reference for various purposes. A characteristic of a software process deployment strategy is the selection of an appropriate reference model to base the definition of the processes to be used in software projects.
- *Change Management*. The implementation of a new process in the organization is a change. Deploying the new or modified process is a change that should be managed. An organization that has the aptitude to manage the change minimizes the resistance of the personnel to change.

It is important to know the relationship among process, projects, and people.

Organizations develop software projects in different places. People are involved in software projects. These projects need staff with skills, abilities, and motivation. These projects use the processes that are contained in the organizational Process Asset Library (PAL). The PAL contains the processes, the models, the standards, the procedures and the adaptation guides, the metrics, and the lesson learned that will be used by the projects.

These processes require roles to develop the tasks. To develop the tasks, the people that develop software need to know the processes and the adaptation guides. As a result of process deployment, the lessons learned and the processes improved are documented and included in the PAL. Figure 2.2 shows the relationship among processes, projects, and people.

3 Critical Success Factors for Process Deployment

The critical success factors were obtained by a systematic review of articles and publications related to deploying, improving, and implementing processes.

The method was used in accordance with systematic review guidelines (Kitchenham 2007; Pino et al. 2008; Biolchini et al. 2005).

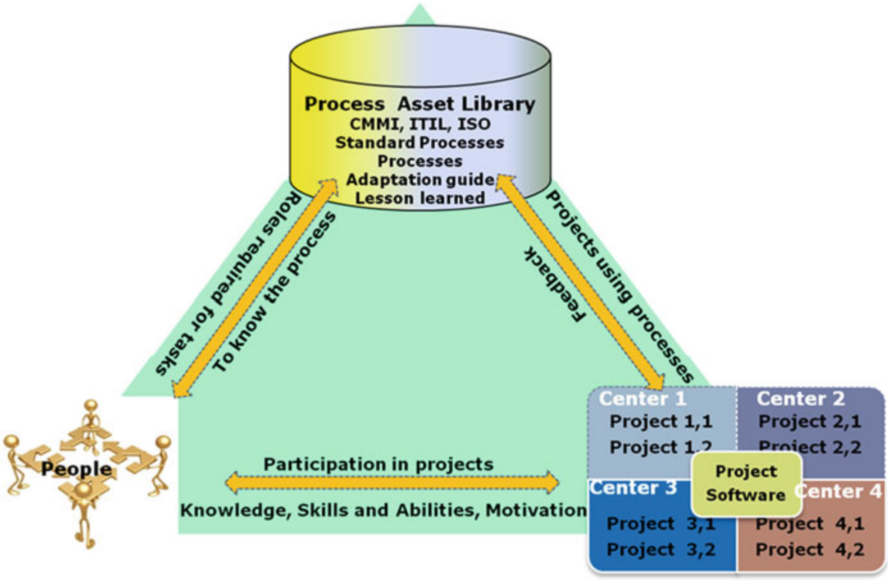


Fig. 2.2 Relationship among process, projects, and people

It has been used in the following: articles, presentations, and technical reports contained in specialized databases such as ScienceDirect, IEEE Computer, ACM Digital Library, SpringerLink, Institute for Scientific Information (ISI) Web Knowledge, and Wiley InterScience and articles and conference presentations specialized as Software Engineering Process Group, European Systems & Software Process Improvement and Innovation (EUROSPI), in addition to reports, articles, and presentations by Software Engineering Institute (SEI), Crosstalk, IT Governance, and Google Scholar.

The systematic review consists of the following stages:

1. *Identify the need for systematic review* to identify these factors that influence process implementation/process deployment in organizations and how they impact on the adoption of the processes.
2. *Review a proposed protocol that allows formulating research questions* is the most important activity of the review to identify the primary studies that respond to questions about the needs. The protocol also indicates how they will locate, exploit, and synthesize the studies.
3. *Conduct the review* to carry out a thorough and comprehensive search of primary studies. The studies identified are evaluated and recorded in the designed formats.
4. *Analyze the results* the data and information as a result of the primary studies are reviewed and analyzed, and the questions are answered.
5. *Report the results of the review.*

The results of the systematic review are (1) an inventory of critical success factors that conditioned the implementation/deployment process of software processes and (2) the categories used by different authors to classify the critical success factors.

The results of the analysis of 31 primary studies related to the models and standards used by the organizations show that the models and standards most used are CMMI and ISO 9000. Most of the primary studies refer to more than one model. The extent review of the literature suggested that there are numerous critical success factors for a successful initiative of process improvement or process deployment. The critical success factors identified by the systematic review were commitment, training, alignment with the goals and business strategy, process definition, roles and responsibilities, staff involvement, organizational culture, methodology for process deployment, change management, communication, and motivation processes (Kaltio and Kinnula 2000; Hantos and Gisbert 2000; Tracy et al. 2002; Guerrero and Eterovic 2004; Wilson et al. 2007; Dybå 2005; Lepasaar et al. 2001; El Emam et al. 1998). Also, the result shows that authors used different terms to denominate a factor.

Next, some examples of the terms used by the authors to communication, commitment, and training factors are presented:

- Communication: encouraging communication and collaboration and providing enhanced understanding (to managers and staff members), communication and collaboration, communication channels, effective communication, and bidirectional communication channels between the workgroups and the software engineering group.
- Commitment: management commitment stakeholders' commitment, senior management commitment, commitment at all levels, commitment at the appropriate management level, high level, management support and commitment, top management support, and top-down commitment.
- Training is planned and made part of the initiative, process-related training, training flexible, training options and training.

Then, it is necessary to standardize and classify the critical success factors in order to use a common language. For this, a basic activity that has been performed is classifying the critical success factors inventory. The purpose of the classification is to enable organizations to identify the factors that may affect the deployment process and include an inventory of the items identified.

The main objectives are:

- Provide support during the preparation of the process deployment method.
- Facilitate the search and grouping of relevant information.

A method for classifying the critical success factors for the process deployment was developed.

Table 2.1 Phases activities to classify the critical success factors

Phase	Definition
Planning	The aim is the planning of the project that will result in the design and implementation of the critical success factors classification
Identification and extraction of information	The aim is to align the work plan with the information needs of the organization. At this stage, we identify the sources of information, the terms to use, and the definitions that will be part of the taxonomy, among others
Design and construction	Design and construction of taxonomy using the inventory of terms. Identify the first level of categorization and other levels to determine the final structure of the taxonomy
Testing and validation	In order to ensure that the taxonomy designed would be useful to users, the necessary tests and validation must be performed
Deployment of the taxonomy	The aim is that the users use the taxonomy, and it must be deployed throughout the organization

The method has been developed in order to serve as a guide for building the taxonomy of critical success factors of the deployment process. The proposed method consists of five phases. Table 2.1 shows the phases. As a result of the implementation of the classification of the critical factors for process deployment, a limited number of categories are identified.

The criteria used to divide and group categories were the categories identified in the systematic review (organization, people, and processes) and according to the context of the study.

Then, applying the top-down technique, a limited number of categories were identified. This taxonomy includes five categories related to the object of study.

The categories are:

1. *Organization*. Many factors which are not covered in the deployment process depend on the organization in which to carry out the deployment process.
2. *People*. The deployment process is based on people at all levels, groups, teams, and organization.
3. *Processes*. Processes are the input of the process deployment.
4. *Product*. Quality product, delivered on time and on budget, and required functionalities.
5. *Others*. This includes other factors not found in the previous categories.

Terminology control is performed by identifying duplication of concepts, synonyms, or terms that designate the same concept. The subcategories identified for people category are shown in Fig. 2.3. The subcategories identified for people category are leadership, communication, knowledge, motivation, values, training, teamwork, participation, change management, and roles and responsibilities.

Fig. 2.3 Critical success factors for subcategory people

4 Case Study

To identify the critical success factors of process deployment in organizations called centers, some research was conducted at five development and maintenance software centers in Spain and South America.

A survey with open questions was carried out in each center with the person in charge of process deployment. The survey consisted of three modules:

- Module 1 was related to the organizational aspects.
- Module 2 was related to use and adoption of the processes deployed.
- Module 3 was related to process deployment.

The survey was carried out in five centers. Each center established its own procedures to carry out the implementation of the processes.

The information analysis was carried out taking into account the following aspects: (1) the Process Asset Library that includes the deployed process, (2) the deployment methodology used, and (3) the impact on the use and adoption of processes.

The metric *process use* aims to analyze the process situation related to process use by the users. The answer types were *"In Use," "Partial Use," "To Modify,"* and *"No Use"* (Module 2). Table 2.2 shows the results of this research. The centers were called C1, C2, C3, C4, and C5.

The first column represents the factors. The second column represents the centers.

Table 2.2 Results of the research by centers

Factors	C1, C4	C2, C3, C5
Creation of the quality and process committee	Yes	Yes
Define and document the procedures with staff participation. Look for a model CMMI	Yes	No
Use of process percentage	>90 %	<50 %
The staff that used the processes participated in their definitions	Yes	No
Had a deployment strategy to decrease the resistance to change	Yes	No
Communication	Yes	Poor
Level of process acceptance (v. high)	>86.5 %	<73.7 %
Resistance to change	No	Yes
Formal training plan	Yes	No

The results of the survey show there are great differences among the centers depending on the process deployment strategy. It shows that the use of process percentage in centers 1 and 4 was greater than 90 % and in the case of centers 2, 3, and 5 was less than 50 %.

In centers 1 and 4, the staff participated in the process definition, whereas in centers 2, 3, and 5, they did not.

Centers 1 and 4 had a deployment strategy to decrease the resistance to change, but the other centers did not.

The level of the process acceptation was above 86 % in centers 1 and 4 which developed a deployment strategy to decrease the resistance to change; this obtained a higher level of acceptation and use of the process deployed in the organizations.

Figure 2.4 shows that at center 1 and center 4, the use of deployed processes percentage is greater than in the other centers.

At center 1, the staff that used the processes participated in their definitions. Center 4 tailored the processes from center 5 in order to get the CMMI certification. Besides, centers 1 and 4 had a deployment strategy to decrease the resistance to change.

At center 5, although they had defined their processes, they did not establish actions to reduce the resistance to change. As centers 2 and 3 had not participated in the process definition, their use was low.

Having identified the critical success factors of the deployment process by systematic review and organizational experiences, we are in a position to propose a method that integrates these factors to ensure the deployment process.

The taxonomy of critical success factors identified by the systematic review has shown the need for bearing not only on technical aspects but to incorporate social aspects in order to achieve the institutionalization of processes.

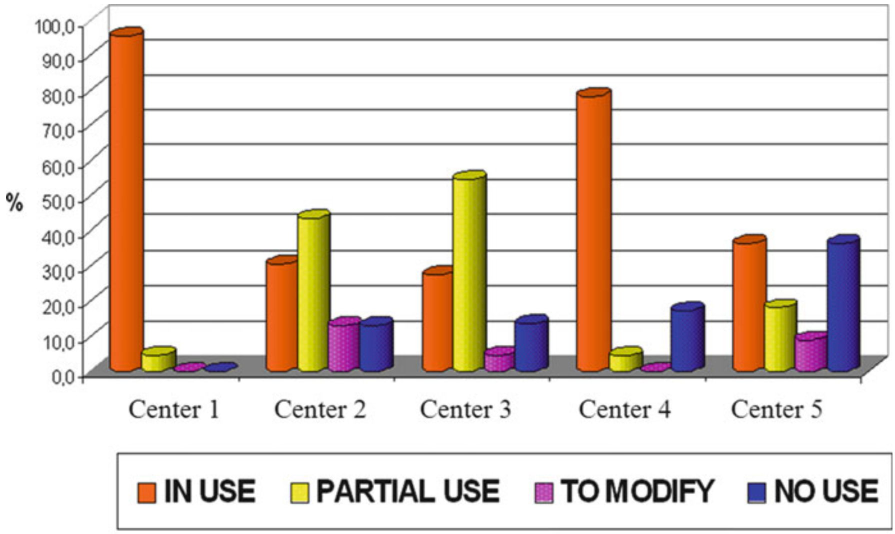


Fig. 2.4 Level of use of the processes deployed

5 MEDEPRO: Method for Process Deployment

A method is proposed to integrate the critical success factors. The objective is developing a method to promote effective and efficient use of processes throughout the organization. It includes the critical success factors such as infrastructure, training, communication, change management, adoption and motivation to use the processes, lessons learned, and metrics to evaluate the deployment process.

This method includes the aspects such as training to develop skills, an effective communication, and change management. This method can be applied to large companies.

We know that the organization has a vision, values, business, objectives, and policies. Also, the organization has an organizational culture, but each person of the organization has an individual culture that when we are going to deploy processes across the organization, it is necessary to take into account those factors.

To deploy the process, it is necessary to have a Process Asset Library. The engineering process is responsible for the process definition that will be used by the organization. The activities of the engineering process are:

1. Identify the organization’s needs.
2. Identify the processes needs.
3. Define the processes.
4. Establish the PAL.

The PAL is the input of the process deployment; for this reason, it is necessary that the processes have been well defined; if they have not, the process deployment can fail.

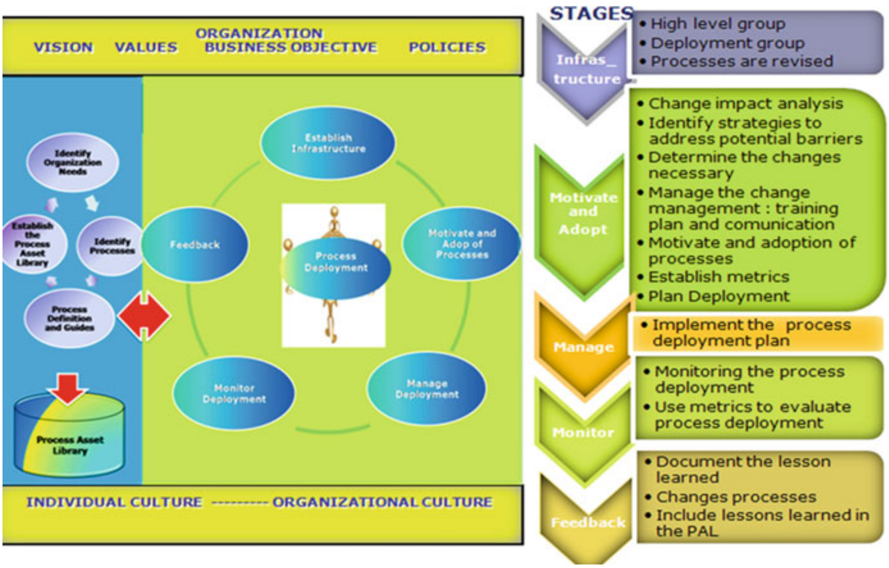


Fig. 2.5 Stages of MEDEPRO method

The method, MEDEPRO, stages are:

1. Establish the Infrastructure. The purpose is to establish the resources necessary to the process deployment.
2. Motivate and Adopt Processes. The purpose is to develop the process deployment plan that integrates the training plan, communication plan, change management plan, metrics process deployment plan, and others.
3. Manage Deployment. The purpose is deploying the processes using the process deployment plan.
4. Monitor Deployment. The purpose is effective process deployment monitor and control using the proposed metrics.
5. Feedback. The purpose is identified lesson learned, all of which must be included in the Process Asset Library.

Figure 2.5 shows the context and stages of the proposed model.

6 Conclusions

That the organization has a process deployment method is a key factor to design and plan a deployment. Another key factor is to involve staff in the process definition.

A strategy to manage resistance to change is fundamental for successful deployment. It is necessary to consider the technical and human aspects in the process deployment. The results of the survey show there are great differences among the centers depending on the process deployment strategy. It is a key factor to design and plan a deployment strategy very early with dedicated resources. The plan is needed to get management support. It is necessary to have a Process Asset Library with easy access and ease of use by staff. The PAL allows knowledge sharing across the organization. The implementation of new processes in organizations is a change. It is necessary to consider the technical and human aspects.

References

Biolchini J, Gomes M, Cruz N, Horta T (2005) Systematic review in software engineering. Technical report RT-ES679/05, Software Engineering and Computer Science Department

Carnegie Mellon Software Engineering Institute (2006) CMMI® for development, version 1.2, CMU/SEI-2006-TR-008. Accessed 10 Oct 2007

Christiansen M, Johansen J (2007) Improvability TM guidelines for low maturity organisations. EuroSPI 2007, pp 91–105

Constantine L (2001) Peopleware papers: the notes on the human side of software. Prentice Hall

Dybå T (2005) An empirical investigation of the key factors for success in software process improvement. IEEE Trans Softw Eng 31(5):410–424

El Emam K, Goldenson D, McCurley J, Herbsleb J (1998) Success or failure? Modeling the likelihood of software process improvement. Technical report no. ISERN-98-15, International Software Engineering Research Network

Guerrero F, Eterovic Y (2004) Adopting the SW-CMM in a small IT organization. IEEE Softw 12 (4):29–35, Pontificia Universidad Católica de Chile

Hantos P, Gisbert M (2000) Identifying software productivity improvement approaches and risks: construction industry case study. IEEE Softw 17(1):48–56

Kaltio T, Kinnula A (2000) Deploying the defined software process. Softw Process Improv Prac 5:65–83

Kitchenham BA (2007) Guidelines for performing systematic literature reviews in software engineering. EBSE technical report EBSE-2007-01, Keele University

Lepasaar M, Varkoi T, Jaakkola H (2001) Models and success factor of process change. In: The product focused software process improvement at PROFES 2001, Kaiserslautern, Germany, 10–13 Sept 2011, pp 68–77

McDermid J, Bennet K (1999) Software engineering research: a critical appraisal. IEEE Proc Softw Eng 146(4):179–186

Messnarz R, Ekert D, Reiner M, O'Suilleabhain G (2008) Human resources based improvement strategies – the learning factor. Softw Process Improv Prac 13(4):355–362

Niazi M, Wilson D, Zowghi D (2005) A framework for assisting the design of effective software process improvement implementation strategies. J Syst Softw 78:204–222

Pino J, García F, Piattini M (2008) Software process improvement in small and medium software enterprises: a systematic review. Softw Qual J 16(2):237–261, Springer

Quinn C (1999) Diagnosing and changing organizational culture. Jossey-Bass, San Francisco

The International Process Research Consortium (2006) A process research framework, Forrester E (ed). Software Engineering Institute. http://www.sei.cmu.edu/iprc/

Tracy H, Austen R, Nathan B (2002) Implementing software process improvement: an empirical study. Softw Process Improv Prac 7:3–15

Wilson DN, Hall T, Baddoo N (2007) A framework for evaluation and prediction of software process improvement success. J Syst Softw 59(2):135–142

Zahran S (1998) Software process improvement. Practical guidelines for business success. Addison-Wesley, Reading

Chapter 3
Problems in the Interplay of IS Development and IT Operations: An Alignment Analysis

Jon Iden, Bjørnar Tessem, and Tero Paivarinta

1 Introduction

The Application Management book was published as a part of the Information Technology Infrastructure Library (ITIL) version 2 initiative (Commerce, O.o.G. 2002). The book addresses the interplay of IS development and IT operations through every phase in the information system life cycle: requirement, design, build, deploy, operate, and optimise. Application management has, however, not yet become widely popular or prioritised among IT departments (Cater-Steel et al. 2009; Iden et al. 2007).

There are several reasons why the interplay of IS development and IT operations is important. First, IS development and IT operations often reside, due to specialisation, in different organisational units. This arrangement may be complicated by the fact that many companies have outsourced their computer centre, leading to a number of risk factors to be managed in their own right (Bahli and Rivard 2005). Second, although IS development, in character, is an activity performed under control of the system development organisation, the responsibility for running the new system in production is transferred to IT operations at the end of the project. Third, an information system is not an isolated entity. Once developed, it will run in an often complex, integrated operational environment and share

---

J. Iden (✉)
NHH – Norwegian School of Economics, Bergen, Norway
e-mail: jon.iden@nhh.no

B. Tessem
University of Bergen, Bergen, Norway
e-mail: bjornar.tessem@infomedia.uib.no

T. Paivarinta
University of Agder, Kristiansand S, Norway

Luleå University of Technology, Luleå, Sweden
e-mail: tero.paivarinta@uia.no

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_3,

hardware, networks, operating systems, and databases with other systems (Peppard and Ward 2004). The new system must comply with the standards, architectures, infrastructures, rules, and operational routines defined by IT operations; conditions that must be clarified and managed by the project. Fourth, deploying a new information system brings about changes in the operational environment, for example, infrastructure, configuration, software, hardware and network capabilities, security, procedures, documentation, service catalogues, service-level agreements, pricing models, personnel, competence, and vendors. It is argued, for example, by application management, that IT operations must be involved early in the system development project in order to make the necessary changes prior to deployment. Non-functional requirements is a commonly used term for many of these issues (Sommerville 2004).

The purpose of this chapter is to study the actual interplay of IS development and IT operations in system development projects, as seen by expert in IT operations. Alignment theory and data from a Norwegian Delphi study form a basis for discussing the contemporary challenges. The chapter proceeds as follows: First, it presents the conceptual basis for this study based on a literature study. Next, it discusses our research methodology. The chapter concludes by discussing the findings and what this may imply for practice and future research.

2 Theory

2.1 IS Development and IT Operations

System development is about building new information systems that support the needs of the business and is commonly organised in projects with fixed goals, budgets, and time limits, normally under the responsibility of IS development (Sommerville 2004). IT operations, on the other hand, are about running and controlling the various information systems in production, maintaining the technological IT infrastructure, and the handling of system incidents and user requests (Commerce, O.o.G. 2007). IT operations reside normally in the computer centre, including such functions as service desk, operation systems and storage management, database and transaction management, networks and telecommunication, and the centre for control and monitoring of online and batch production and networks (Commerce, O.o.G. 2007). Both IS development and IT operations have traditionally been organised under the IT department, but outsourcing and offshoring arrangements have introduced also other forms of organising.

Development of information systems may involve a number of failures; the project may exceed the budget, the system may be delivered behind schedule, and the system may fail to meet the users' requirements (Yeo 2002). The factors that cause projects to fail or their results to be challenged may include lack of effective project management skills, lack of adequate user involvement, incomplete or changing requirement specifications, lack of executive support, lack of training, and immature technology (Schmidt et al. 2001; Fowler and Horan 2007). Although similar research is harder to find for IT operations, this area is not without its failures either. A Gartner Group report, for example, says that nearly 80 % of production outages (unplanned downtime) occur as a result of what they call ‘operator error’ and ‘application failures’, while the remaining 20 % are a result of technology (referred to in Commerce, O.o.G. 2002).

Within IS development and IT operations, initiatives are recurrently taken to improve task effectiveness and outcome quality. New system development methods are explored and introduced (Boehm 1988; Jacobson et al. 1999; Schwaber 2004; Beck and Andres 2004), and the software process itself is consciously being improved (Conradi et al. 2006). By applying the recommendation in the Information Technology Infrastructure Library (ITIL), IT operations worldwide are on a course to change their practices from the traditional technical orientation to a more customer- and service-oriented style (Pollard and Cater-Steel 2009). However, studies show that improvement initiatives in IS development and IT operations are, in general, limited to challenges within each respective domain and seldom addresses topics that concern the both domains and their relationship (Tessem and Iden 2007). ITIL’s Application Management is a rare case.

2.2 Alignment

Alignment is a term often used to characterise the relationship between two parties, and it has been found to have an important effect on business value from IT (Chang et al. 1997; Papp 1999; Teo and King 1996). According to Reich and Benbasat (2000), alignment has two dimensions, one intellectual and one social. The intellectual dimension is identified by organisational structures and the existence of deliverables like plans and other documents, while the social dimension is identified by common knowledge and shared commitments. Tiwana et al. (2003) add a cognitive dimension which covers psychological relationship issues such as trust, mutual understanding, and commitment.

A related term that we find is encompassing and enriching the perspective, and content of alignment is partnership. According to Subramani et al. (1999), partnership is ‘patterns of cooperative interaction between independent actors’ (p. 5), a definition that is putting emphasis on mutual activities. Henderson (1990) on his side includes more of the social dimension in his understanding of partnership when he defines it as ‘working relationship that reflects a long-term commitment, a sense of mutual cooperation, shared risk and benefits, and other qualities consistent with concepts and theories of participatory decision making’ (p. 8). IS development–IT operations alignment is in this chapter understood as the process and outcome of the intellectual, social, and psychological relationships between the two domains in the system development project, both sensing a degree of partnership, and materialised through cooperative actions.

Reich and Benbasat (2000) studied factors influencing alignment in the IT – business context. They found support for that (1) shared domain knowledge, (2) communication between business and IT executives, and (3) connections between business and IT planning processes are necessary conditions for alignment. Our analysis in this study will be based on Reich and Benbasat's three factors. In addition, we will add a fourth factor based on the work of Luftman and Brier (1999), as they, in addition to the above factors, have found partnership to be an important alignment enabler.

Shared domain knowledge. This is understood as the ability of IS development and IT operations, at a deep level, to understand and be able to participate in the others' key activities and processes and to respect each other's unique contribution and challenges. The qualities of this dimension could be measured by assessing the actual education, knowledge, and experience members of one of the two domains have of the other domain (Reich and Benbasat 2000).

Communication between IS development and IT operations. This is understood as how participants in the respective domains create and share information with each other to reach mutual understanding. This could be by direct communication such as mail, written memos, and regular and ad hoc meeting but also by nominating liaisons roles (intermediates), task forces, and committees (Reich and Benbasat 2000).

Connections between the planning processes and activities of IS development and IT operations. This is understood as how individuals from the opposite domain participate in the activities arranged by the others and that each domain's planning processes and activities are integrated with the other. That is, to what extent they constitute a totality. For instance, IS development and IT operations are both taking part in the development project, and plans and results are ratified by the other part (Reich and Benbasat 2000).

IS development and IT operations partnership. This is understood as a working relationship that reflects a long-term commitment, a sense of mutual cooperation, shared goals, risk and benefits, and other qualities consistent with concepts and theories of participatory decision making" (Henderson 1990; Luftmann et al. 1999; Anandhi et al. 1999).

3 Research Method

A Delphi study which was based on the method provided by Schmidt (1997) and Schmidt et al. (2001) included 42 Norwegian IT experts divided into three panels, 20 system developers, 12 persons from IT operations, and 10 systems owners (persons from the business domain). Of the 12 experts in the IT operations panel, six represent in-house operations and six represent service providers. Data collection and analysis were divided into three distinct phases: brainstorming, reduction, and ranking. The problem of nonresponse was exceptionally low, as only one expert, a system owner, left during the later phases of the study. From the brainstorming phase in the Delphi study, a validated list of 66 problems was produced, together with rich textual descriptions by the experts on each of the problems. As the result of the reduction and ranking phases, a ranked list of the 17 most important problems, as viewed by the experts, was produced.

4 Analysis

The focus of the Delphi study was on the problems encountered in the interplay between IS development and IT operations in system development projects. The brainstorming phase of the Delphi study produced a validated list of 66 problems. The majority were easily categorised according to the four alignment dimensions:

- Shared domain knowledge: 18 problems (27 %)
- Communication between IS development and IT operations: six problems (9 %)
- Connections between the planning processes and activities of IS development and IT operations: 17 problems (26 %)
- IS development and IT operations partnership: 17 problems (26 %)

Eight problems (12 %) were not easily placed in any of the four categories. However, none of them were prioritised for the ranking phase of the study.

In the Delphi study, the experts also selected and ranked the 66 problems according to the problems that they considered most important among. This produced a ranked list of 17 problems which will be analysed according to the four alignment dimensions in Table 3.1. The final ranking set by IT operations is given in parenthesis.

5 Discussion

5.1 Shared Domain Knowledge

Three of the problems are related to this dimension. It is evident that system developers, in the eyes of IT operations, do not possess sufficient knowledge about the operational environment, the characteristics of the infrastructure, its configuration, or the routines that directs the work. As one expert in IT operations expressed it, ‘There is a big gap between IS development and IT operations. Consequently, the two groups obtain too little insight into each other’s domains. The result is mistrust and lack of shared knowledge’. Many experts from both domains raised the fact that system developers and operatives have very different competencies and skills. The criterion for ‘shared domain knowledge’ that they are able to understand and participate in each others’ key activities is thus not fulfilled.

Table 3.1 The 17 most important problems from the Delphi study categorised into the four alignment dimensions. Final ranking set by IT operations is given in parenthesis

Dimensions	Problems
Shared domain knowledge	The documentation that IT operations receive with a new system is not adequate (7)
	System development does not know how IT operations work or what routines apply to deployment (10)
	System developers are not well informed about how the entire production environment works, how the different systems work together (11)
Communication between IS development and IT operations	IT operations receive insufficient information about how the new system works because of missing knowledge transfer (3)
	Communication and the flow of information between IS development and IT operations are poor (8)
Connections between the planning processes and activities of IS development and IT operations	IT operations are not involved in the requirement specification; consequently, technological and operational aspects are thus not sufficiently taken care of (1)
	New systems are put into production too early, before they are complete (2)
	IT operations are involved in the system development project too late (4)
	Too little emphasis is set on establishing satisfactory test environments for volume, performance, and capacity testing (5)
	Operational routines for the new system are not established prior to deployment (6)
	The process for handing over a new system from IS development to IT operations is not well structured (12)
	IT operations are not formally involved in the system development project (13)
	The needs for scheduling are divergent. IT operations need predictability and a long-range planning perspective, while system development seeks flexibility (16)
IS development and IT operations partnership	IT personnel who have obligations in both system development and IT operations find it difficult to commit time to system development work (9)
	Initiatives for improving the processes in the interplay of IS development and IT operations are not being prioritised (14)
	IS development and IT operations are functionally separate working environments prioritising their own tasks (15)
	Personnel in IT operations do not possess the same level of ownership of a system as developers do (17)

5.2 Communications Between IS Development and IT Operations

Only 2 of the 17 problems are related to this dimension, and thus, it may seem this dimension is better off than the other dimensions. However, the rankings experts gave to these problems indicate that communication is a serious issue; one problem is ranked as third and the other as eight. By analysing the experts' comments, we find several explanations to the situation: 'Personnel do not meet, they communicate only through e-mail and documents,' 'system developers are requested to use an electronic service-desk system when requesting services from IT operations', 'the two groups use different terms for the same things, they do not use the same language', 'there is a formal bureaucracy between the members of the two groups', and 'it is not clear who to contact in the other area about a certain issue'. It is also worth to point out that the system owners' panel (business domain) gave 'poor communication and information flow' between the two domains, the highest ranking. This indicates that poor communication and information flow is highly recognised even by those observing the interplay from a distance. The criterion for 'communication' that they create and share information with each other to reach mutual understanding is thus not fulfilled.

5.3 Connections Between the Planning Processes and Activities of IS Development and IT Operations

Eight problems are related to this dimension, and some of the highest-ranked problems are found here. For example, 'IT operations are not involved in the requirement specification; consequently, technological and operational aspects are thus not sufficiently taken care of' is ranked highest by IT operations. Analysing the eight problems reveals three serious issues: IT operations are not formally defined as a member of the system development project; IT operations are not involved when requirements are being specified; and, in general, IT operations are involved too late in the development process. In addition, we find one problem stating that the process for handing over a system from the project to IT operations is not well structured, an issue that many of the experts, system developers, and operatives alike commented on. Consequently, operational routines for a new system are not ready at deployment, IT operations do not understand how the new system works, and the production environment that the system is deployed in is not optimal or cost-efficient. The criterion for 'connections between the planning processes and activities of IS development and IT operations' that their tasks are integrated and constitute a whole is thus not fulfilled.

5.4 IS Development and IT Operations Partnership

Four of the problems are related to this dimension. It is evident that the experts in IT operations consider the two domains as separate areas with different goals, priorities, and areas of focus. There seems to be little observable interest in improving the interplay. The four problems within this dimension are among those that received the lowest rankings. This can be interpreted in two ways: Either the experts do not consider these particular problems to be especially serious or the experts are not particularly concerned about or interested in partnership as an issue. IT operations made several comments about these problems: ‘There is an “us and them” situation.’, ‘we do not have the same focus and priorities’, ‘our goals are different and can sometimes stand on conflict with each other’. The criterion for ‘IS development and IT operations partnership’ that they have a sense of mutual cooperation where goals, risks, and benefits are shared is thus not fulfilled.

5.5 Implications for Practice

The issue of connecting the planning processes and activities of IS development and IT operations is perhaps the most prominent category of alignment issues which prevails in the Norwegian IS industry according to our results. This highlights the needs to rethink system development projects by involving IT operations to the development processes and giving IT operations adequate resources to be involved. Better practices for system handover from IS development to IT operations should also to be defined. Improvement of this interplay requires increased managerial attention on communication practices and, in case of outsourcing, better uses of rich communication tools between these stakeholders. Findings indicate the importance of synchronous meetings instead of relying on paper forms, e-mail, or even service-desk-based communication. It is also evident that both areas need more precise information about who to contact in the other area. Both domains would benefit if roles and responsibilities were defined better, an issue that has been highlighted with the widespread adaption of the Information Technology Infrastructure Library (ITIL) (Cater-Steel et al. 2009; Iden 2010). Communication and roles become even more important in the cases where IT operations and IS development are located in different administrative units or business organisations.

Improved communication and mutual planning processes would contribute further to the issue of shared domain knowledge. The goal of shared knowledge could also be pursued through involvement of both developers and IT operations staff in professional education programmes with a focus on blending system development themes with operational concepts. Reich and Benbasat’s study showed that shared domain knowledge between business and IT executives was the strongest predictor of the social dimension of alignment (Reich and Benbasat 2000). The low rankings of the partnership issues among the IT operations personnel may be, in turn, understandable, as they are seldom in such positions in which they could actually establish new longer-term or binding cooperation agreements among different organisations. However, it remains as a managerial responsibility to consider, how much longer-term partnerships between developers and operations personnel would be beneficial, for example, in comparison to the market-based development strategy to seek for shorter-term development contracts with external firms. All in all, the partnership strategy likely has a profound impact on the three other alignment categories.

5.6 Implications for Research

After nearly a decade since the ITIL Application Management recommendations (Commerce, O.o.G. 2002), our study shows that IT operations still experience a wide set of alignment challenges in system development projects. While this research addresses the challenges, further efforts would be needed for improving the actual interplay practices in organisations. Practice-oriented research is needed to report experiences from contemporary practices and perhaps even to try out innovative practices and tools for enhancing IS development/IT operations alignment. This research deals with empirical data, but its findings are derived and discussed in a manner independent of specific system development methods. This is a limitation of the study. Further research could investigate our research question in specific methodological settings. Another stream of future research would include in-depth studies where the potential cause–effect relationships for particular alignment issues could be established. This would help to identify the root causes of the alignment challenges. Future research should also look into what the antecedents and outcomes of good IS development and IT operations alignment are and how they can be measured. Academics should also rethink their curriculums in order to ground the basis for a well-performed interplay already at the educational stage.

6 Conclusions

In this study, we have explored the interplay of IS development and IT operations in system development projects. Our approach has been to use the alignment theory perspective to analyse data from a ranking-based Delphi study. A well-performed interplay of the two areas is of uttermost importance, as argued by the authors of ITIL's Application Management (Commerce, O.o.G. 2002). It is therefore unfortunate to find that the two domains do not cooperate well and are not especially well aligned. Many issues and problems have been revealed that restrict their ability to cooperate efficient and effectively. The analysis provides several possible explanations for the situation. The experts agree that the interplay of IS development and IT operations in system development projects need to attention, and their comments give some issues to work on: Close the organisational divide, develop methodologies that coordinate the two groups' plans and activities, involve IT operations staff in the development project from the beginning, improve the process for handing over a new system from IS development to IT operations, and educate employees in the other team's profession.

References

Anandhi SB, Sambamurthy V, Robert WZ (1999) IT capabilities: theoretical perspectives and empirical operationalization. In: Proceedings of the 20th international conference on information systems, Charlotte, North Carolina, United States. Association for Information Systems

Bahli B, Rivard S (2005) Validating measures of information technology outsourcing risk factors. Omega 33(2):175–187

Beck K, Andres C (2004) Extreme programming explained: embrace change, 2nd edn. Addison-Wesley Professional, Boston

Boehm B (1988) A spiral model of software development and enhancement. IEEE Comput 21 (5):61–72

Cater-Steel A, Tan W-G, Toleman M (2009) itSMF Australia 2009 conference: summary report of ITSM standards and frameworks survey. University of Southern Queensland, Toowoomba, Australia

Chang YE et al (1997) Business strategy orientation, information systems strategic orientation, and strategic alignment. Inf Syst Res 8(2):125–150

Commerce, O.o.G. (2002) Application management. The Stationary Office, London

Commerce, O.o.G. (2007) The official introduction to the ITIL service lifecycle. The Stationary Office, London

Conradi R et al (2006) Software process improvement: results and experience from the field. Springer, Berlin

Fowler J, Horan P (2007) Are information systems' success and failure factors related? An exploratory study. J Organ End User Comput 19(2):1–22

Henderson JC (1990) Plugging into strategic partnerships: the critical IS connection. Sloan Manage Rev 31(3):7

Iden J (2010) The adoption of ITIL in the Nordic countries: a survey. In: Norsk konferanse for organiasjoners bruk av IT (NOKOBIT). Tapir Akademisk Forlag, Gjøvik

Iden J, Steindal F, Stokke B (2007) The implementation of IT infrastructure library (ITIL) in Norway: progress, success factors and benefits. In: Norsk konferanse for organisasjoners bruk av informasjonsteknologi. NOKOBIT 2007. Tapir Akademisk Forlag, Trondheim, pp 43–56

Jacobson I, Booch G, Rumbaugh J (1999) The unified software development process. Addison-Wesley Longman Publishing Co, Inc., Reading

Luftman J, Brier T (1999) Achieving and sustaining business-IT alignment. Calif Manage Rev 42 (1):109

Luftmann J, Papp R, Brier T (1999) Enablers and inhibitors of business-IT alignment. Commun Assoc Inf Syst 1(11):1–33

Papp R (1999) Business – IT alignment: productivity paradox payoff? Ind Manage Data Syst 99 (8):367–373

Peppard J, Ward J (2004) Beyond strategic information systems: towards an IS capability. J Strateg Inf Syst 13(2):167–194

Pollard C, Cater-Steel A (2009) Justifications, strategies, and critical success factors in successful ITIL implementations in U.S. and Australian companies: an exploratory study. Inf Syst Manage 26(2):164–175

Reich BH, Benbasat I (2000) Factors that influence the social dimension of alignment between business and information technology objectives. MIS Q 24(1):81–113

Schmidt R (1997) Managing Delphi surveys using nonparametric statistical techniques. Decis Sci 28(3):763–774

Schmidt R et al (2001) Identifying software project risks: an international Delphi study. J Manage Inf Syst 17(4):5–36

Schwaber K (2004) Agile project management with scrum. Microsoft Press, Redmond

Sommerville I (2004) Software engineering. Pearson Education Ltd., Essex

Subramani MR, Henderson JC, Cooprider J (1999) Linking IS-user partnerships to IS performance: a socio-cognitive perspective. MISRC working paper WP99-01, University of Minnesota, Minnesota

Teo TSH, King WR (1996) Assessing the impact of integrating business planning and IS planning. Inf Manage 30(6):309–321

Tessem B, Iden J (2007) Issues in the transfer of software from development to operations. In: NOKOBIT 2007 – Norsk konferanse for organisasjoners bruk av IT. Tapir akademisk forlag, Oslo

Tiwana A, Bharawaj A, Sambamurthy V (2003) The antecedents of information systems development capability in firms: a knowledge integrating perspective. In: 24th international conference on information systems (ICIS), Seattle, USA

Yeo KT (2002) Critical failure factors in information system projects. Int J Proj Manage 20 (3):241–246

Chapter 4
The Agile and the Disciplined Software Approaches: Combinable or Just Compatible?

Lise Tordrup Heeager

1 Introduction

In information systems development research, agile software development is a popular topic. This chapter uses the terminologies of agile and disciplined software development approaches adopted from Boehm and Turner (2003a, 2004). The agile and the traditional disciplined software methodologies are two approaches to describe processes for developing software (Dahlberg et al. 2006). The agile software development methodologies (e.g. XP and Scrum) promise a way to deliver software without excessive cost. On the contrary, disciplined software development processes (Boehm and Turner 2004) (e.g. CMMI and the ISO standards) are well defined and proven, but require a lot of effort (Nawrocki et al. 2002a).

Offhand, the two seem contradicting (Turner 2002; Turner and Jain 2002), but several researchers agree that a software project needs both agility and discipline (Boehm and Turner 2003a, b; Nawrocki et al. 2006). Neither the agile nor the disciplined approaches provide the ultimate approach, both have shortcomings and pitfalls (Boehm and Turner 2003a). The approach of the disciplined methodologies (plan everything and follow the plan) works well for stable and less complex projects, but they lack the ability of handling change and complexity. The agile methodologies embrace change (Beck and Andres 2004), but offhand they lack the ability of quality assurance.

Software development organizations are increasingly interested in the possibility of adopting agile approaches (Pikkarainen and Mäntyniemi 2006); even organizations that have been employing process standards are now increasingly interested in the possibility of adopting agile approaches (Marçal et al. 2008). Rapid change and increasing software criticality drive organizations to balance the agility and discipline (Boehm and Turner 2004).

L.T. Heeager (✉)
Aalborg University, Aalborg Ø, Denmark
e-mail: liseh@cs.aau.dk

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_4,

There are ongoing debates whether the quality of the products of the agile approaches are satisfactory (Hashmi and Baik 2007), and some projects, for example, safety-critical projects, require standards to be followed when developing software (Fritzsche and Keil 2007; Heeager and Nielsen 2009; Theunissen et al. 2003). Since the agile and the disciplined approaches are grounded in opposing concepts, balancing the two is not straightforward (Opelt and Beeson 2008; Reifer 2003; Vinekar et al. 2006). Some organizations attempt to use an agile software development approach and at the same time comply with a quality assurance standard (Vinekar et al. 2006; Nerur et al. 2005); however, it is not well understood how to do this in practice (Kähkönen and Abrahamsson 2004; Nawrocki et al. 2002b; Pikkarainen 2009). This has let research to focus on the compatibility and combinability of the agile and the disciplined approaches (Rönkkö et al. 2008; Zanatta and Vilain 2006).

With the research question 'are the agile and disciplined approaches combinable or just compatible?' in mind, the purpose of this chapter is to give an overview of the combinability and/or the compatibility of the agile and disciplined approaches and give a more detailed picture of the challenges facing an organization trying to use an agile software development approach in a disciplined setting. This is done through a review of 79 papers dealing with the subject of the combinability or compatibility of the agile and disciplined approaches. This review is relevant to researchers who wish to stay up to date with the state of research on the combinability and/or compatibility of agile and disciplined approaches and to practitioners who wish to implement an agile software development process in practice and at the same time comply with a quality assurance standard.

This chapter is structured as follows: The research approach is outlined in Sect. 2. Then the analysis on compatibility and combinability is presented in Sect. 3. In Sect. 4, the results of the analysis are discussed and also include a discussion of the guidelines and the challenges identified in the literature when trying to obtain compatibility between the agile and disciplined approaches. The analysis results in four propositions are also presented in Sect. 4. Finally, the results are concluded on in Sect. 5.

2 Research Approach

Previous research on the combinability and/or compatibility of the agile and the disciplined approaches was reviewed. The papers were found through a comprehensive search conducted in three steps (summarized in Fig. 4.1) (Webster and Watson 2002). First, the web was searched using relevant keywords. The keywords contained different combinations of names of process standards and agile methodologies and more general keywords as 'compatibility and agile' or 'combine and agile'. In parallel, ranked journals and relevant conference proceedings were searched. In the second step, a backward search (following references of identified papers) and a forward search (used Google Scholar to find relevant papers citing the

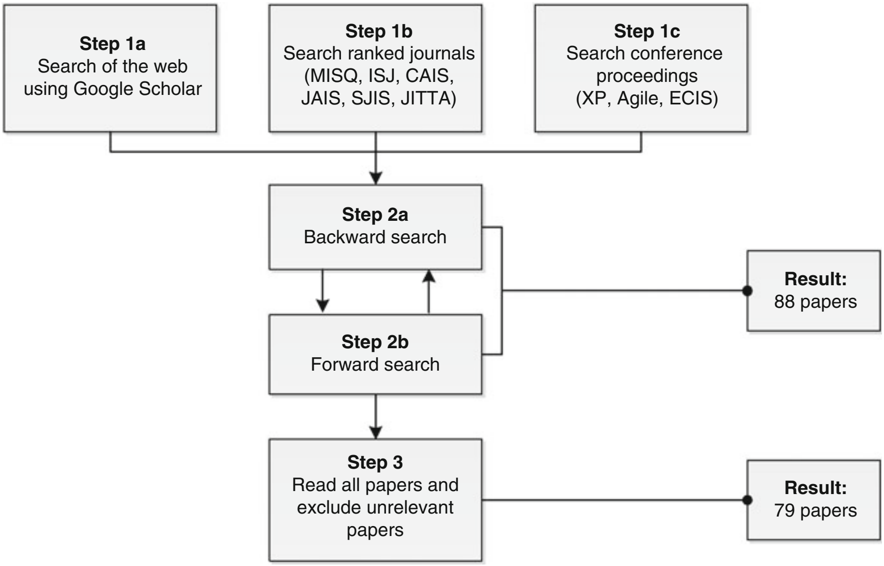


Fig. 4.1 The search strategy

identified papers) were done iteratively. At this point, the relevance of the papers was determined by reading the titles and abstract, so far the search had resulted in 88 papers. Each of these was read, and nine of them were discarded as they lacked relevance, either because the combinability and/or compatibility of the agile and the disciplined approaches was a very small and insignificant part of the paper or the fact that the paper only was dealing with either agile or disciplined approaches and not both, leaving 79 papers for further analysis (see reference list in Appendix A).

Conventional content analysis (Hsieh and Shannon 2005) was used as the primary analysis strategy (the analysis strategy is summarized in Fig. 4.2). This strategy was chosen as existing theory, and research literature on the combinability and/or compatibility of the agile and the disciplined approaches is limited. To structure the analysis, each of the papers was coded using Atlas.ti V6 (Muhr 1991). The process consisted of three linear steps – each with a separate result. First, the papers were coded using open coding (Strauss and Corbin 1990). The focus when reading and coding the papers was on the combinability/compatibility and non-combinability/incompatibility of the agile and the disciplined approaches, on guidelines to overcome the challenges when using an agile software development approach in a disciplined setting and on the strengths and weaknesses of such a practice. This was done in order to get an overview of the content of the previous research. The first step resulted in 59 different codes applied to 1,038 quotations. The second step was to categorize the codes, thus creating coding schemes. The resulting coding schemes can be seen in Appendix B. One table for each type of codes include the following: general codes, codes describing combinability /compatibility, codes describing non-combinability/incompatibility, codes

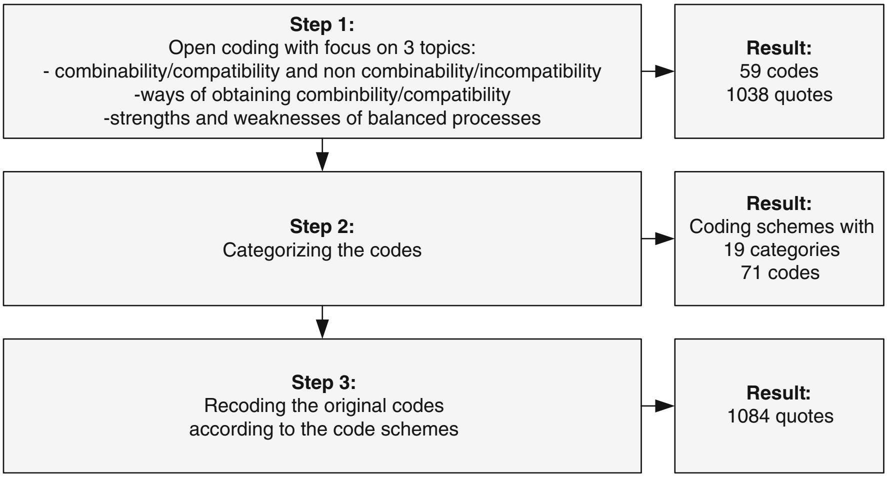


Fig. 4.2 The analysis strategy

describing guidelines of how to use an agile software development approach in a disciplined setting and codes describing strengths and weaknesses of the agile, the disciplined approaches attempting to balance agility and discipline. In the third step, the codes and quotes from step 1 were revisited and recoded according to the coding schemes containing 19 categories with 71 codes. This resulted in 1,084 quotations.

3 Analysis: Combinability and Compatibility

To be combinable or compatible are two different things. In order to determine the combinability and compatibility of the agile and the disciplined approaches, these two terms have to be defined. This is done with a starting point in the dictionary interpretation of these key terms.

To combine is 'to join two or more things or groups together to form a single one' (Wehmeier 2010), meaning that you are able to do two things at the same time without compromising either. To be compatible is to be 'able to exist together' (Wehmeier 2010), meaning that something is able to work at the same time as something else. Thus, combinability is a Boolean concept while compatibility can be graded.

The interpretations of the two key terms used in this chapter in the context of determining the compatibility and combinability of the agile and the disciplined approaches are as follows: (1) The agile and the disciplined software approaches are combinable if it is possible to use an agile software development approach and at the same time comply with a quality assurance standard without compromising the agility of the agile approach. (2) The agile and the disciplined software development approaches are compatible if it is possible to use some of the agile practices and principles and at the same time be able to comply with a quality assurance standard

The literature review shows that previous research does not distinguish between combinability and compatibility of the agile and the disciplined approaches. Several papers state that the agile and disciplined approaches are combinable; however, when reading into the analysis and results presented in the papers adopting the definitions of this chapter, the conclusion should instead be that the agile and the disciplined approaches are compatible, but not combinable (e.g. S10, S12, S26, S75).

3.1 The Compatibility of the Agile and the Disciplined Approaches

Offhand, the agile approaches seem in conflict with the disciplined approaches, but several researchers agree that it is possible to use (some of) the practices and principles of agile software development process and at the same time comply with a quality standard. The majority of the papers reviewed (e.g. S3, S10, S23, S24, S25, S36, S46, S61, S65, S69, S70, S79) state that the agile and the disciplined approaches (to some degree) are compatible. Most papers engage themselves in determining the overall compatibility or incompatibility of the agile and the disciplined approaches, and some of these engage in comparing specific methodologies, mainly using XP or Scrum from the agile methodologies and CMMI and ISO from the disciplined methodologies.

Thirty-nine of the papers reviewed outline which agile and disciplined approaches they compare. These are shown in Table 4.1. The relationship between XP and CMMI is the one most analysed. Scrum comes in second among the agile approaches, whereas ISO is the process model second most used. Five papers not only analyse the relationship of an agile and a disciplined approach but also mixed practices from two or more agile approaches. Vriens (S74) not only used a mix of XP and SCRUM but also a mix of CMMI and ISO.

Table 4.1 Comparison of the agile and the disciplined approaches in the reviewed papers

	XP	Scrum	RUP	XP/Scrum	Sum
ISO	(S42, S43, S48, S67, S77)	–	–	(S41)	6
CMMI	(S2, S3, S18, S28, S30, S40, S45, S47, S49, S51, S54, S55, S60, S62, S75)	(S16, S26, S27, S34, S35, S38, S39, S66, S78)	(S33, S37)	(S1, S12, S61)	29
CMMI/ISO	(S14)	–	–	(S74)	2
FDA	(S76)	(S24)	–	–	2
Sum	22	10	2	5	

In a point-counterpoint paper, Beck and Boehm agree that agility and discipline are not opposites (S4). In another paper, Boehm also states that even though advocates of the agile and the disciplined approaches consider them opposites, including parts of both in a project can be advantageous (S7), and Glass (S21) points out that 'one-size-fits-all' approaches do not work. Mahnic and Zabkar (S34, S35) conclude that it is possible to build software process that balances agility and discipline. Glazer (S22) concludes that the agile approaches and CMMI complete each other and can give fast, affordable, visible and long-term benefits. Through a comparison of the waterfall model and agile approaches, Huo et al. (S25) find that agile approaches contain QA practices and that these occur more frequently than in the waterfall model. Baker (S3) states that the agile approaches are focused and comprehensive, and according to Beck (S4), XP is disciplined, as it provides a clear picture of what activities to apply. DeMarco and Boehm (S15) support this statement by adding that XP involves more planning than CMM organizations.

According to the reviewed papers, using an agile approach in a disciplined setting makes it possible to take advantage of the strengths and compensate for the weaknesses of the two (S10, S19, S44). Turner and Jain (S70) state that the agile and the disciplined approaches have much in common and their strengths and weaknesses are complementary, while Boehm and Turner (S9, S11) present a risk-based method for developing balanced strategies.

However, only few papers deal with the strengths and weaknesses of processes balancing agility and discipline, and those who do only point out the strengths. The disciplined approaches support the agile approaches by providing a disciplined framework (S17), while agile methods ensure that processes are implemented efficiently while embracing change (S14, S26, S66), this balances adaptability and predictability in order to better serve customer needs (S27). The disciplined approaches are also helpful for the agile approaches, as they are able to help identify the shortcomings of the agile approaches (S18), support the decision of which processes to address (S26, 66), provide structure to help ensure your agile processes are followed (S41) and reduce the risk in agile development (S17).

The researchers disagree on how compatible the agile and the disciplined approaches are. The majority of the papers reviewed conclude that it is possible for a company using an extended agile approach to comply with process standards such as ISO and CMMI level 2 or 3. In the following subsections, the compatibility and/or combinability of the agile approaches and, respectively, the ISO standards and the CMM/CMMI will be analysed.

3.2 Agile Software Methodologies and ISO

The ISO standards ISO 9000 and ISO 9001 are those mostly used for analysis by researchers dealing with agile software development methodologies and ISO (S41, S42, S43, S48, S64, S77). Three papers do not specify which agile methodology they use for analysis, while the remaining five papers either use Scrum or XP (see Table 4.2).

Table 4.2 Comparison of agile methods and ISO in the reviewed papers

	XP	Scrum	Agile	Sum
ISO 9000	–	(S43, S48)	–	2
ISO 9001	(S42)	(S77)	(S41, S64)	4
ISO/IEC15504 (SPICE)	–	–	(S29)	1
ISO/IEC12207/1995	(S67)	–	–	1
Sum	2	3	3	

Agile approaches can be adapted to ensure compatibility with ISO standards (S67). Based on empirical evidence, Wright (S77) concludes that companies using XP can meet the requirements of ISO9001. McMichael and Lombardi (S41) conclude that ISO9001:2000 is able to help ensure the agile processes are followed. Melis (S42) concluded that several requirements of ISO9001:2000 are implemented by the existing tools in XP project management, while Stålhane and Hanssen (S64) suggest few changes to both the ISO standard and the agile approaches. Lami and Falcini (S29) showed that ISO/IEC15504 in principle is applicable to agile contexts, but that in practice, problems may occur, for example, in creating an agile process reference model and in order to find an assessor that has experience from software development in agile contexts.

3.3 Agile Software Methodologies and CMM/CMMI

Several researchers of the papers (18 of the review papers) compare the components of the CMM (S30, S37, S40, S55) or CMMI (S1, S12, S16, S17, S18, S30, S38, S39, S40, S45, S51, S66, S70, S78) process model with agile methods, such as XP and Scrum. Tables 4.3 and 4.4 give an overview of which agile methods are compared with the generic practices and process areas of, respectively, CMMI and CMM.

An analysis of the results on the comparison of agile methods with the process areas of CMMI shows that the researchers agree that CMMI level 2 is largely compatible with the agile approaches, CMMI level 3 are partially compatible with agile approaches while CMMI level 4 and 5 are less compatible with agile approaches. There does not seem to be a big difference in the compatibility of CMMI and, respectively, XP and Scrum.

Fifteen papers state that the agile approaches are compatible with CMMI level 2 or 3. Marcal et al. (S38, S39) say that it is possible to reach CMMI level 2 without compromising the agility. Alegria and Bastarrica (S1) state that each process area needs to be defined explicitly by the organization and complemented with elements obtained from other sources. Baker (S3), Bos and Vriens (S12) and Kähkönen and Abrahamsson (S28) conducted case studies of companies complying with CMMI level 2 using agile approaches. Santana et al. (S62) conducted studies of two companies complying with CMMI level 2 and 3, respectively, using XP and

Table 4.3 Comparison of agile methods and CMMI in the reviewed papers

	XP	Scrum	RUP	Agile	Sum
CMMI generic practices	(S18)	(S16, S18, S66)	–	(S1, S70)	6
CMMI process areas	(S12, S18, S30; S40, S45, S51)	(S12, S18, S38, S39, S78)	–	(S1, S17, S44, S70)	15
Sum	7	8	0	6	

Table 4.4 Comparison of agile methods and CMM in the reviewed papers

	XP	Scrum	RUP	Agile	Sum
CMM generic practices	–	–	–	–	0
CMM process areas	(S30, S40, S55)	–	(S37)	–	4
Sum	3	0	1	0	

Scrum, while Manzoni et al. (S37) conclude that reaching CMMI level 3 using RUP is possible. Rönkkö et al. (S61) found that the process areas of CMMI level 2 and 3 are adopted in parallel and can be supported by the use of agile approaches. Jakobsen and Johnson (S26) recommend extending the agile approaches using the mandatory goals and expected practices of CMMI level 2 and 3. Laurila (S30) concluded that CMM and agile methods are practice compatible, but not idea compatible.

Several researchers state that agile methods do not address the CMMI level 4 or 5 in their original state (e.g. S8, S30, S40, S55), but only one paper states that it is not possible to reach the highest maturity levels when being agile (S18). Few results show that it is possible to comply with CMMI level 4 or 5. Through an analysis of XP and CMM, Martinsson (S40) found it possible and advantageous to use XP to reach the highest CMM level. At Microsoft, they created an agile life cycle complying with CMMI level 5 (S2), and at Systematic, they used Scrum to reach the highest level (S66, S77).

The disciplined approaches tell what to do while the agile approaches tell how to do it (S16). CMMI is a maturity model while agile is a development philosophy (S31).

3.4 Agile Approaches for Developing Safety-Critical Software

Another ongoing discussion in the reviewed papers is whether or not agile approaches are useful when developing safety-critical software (S32). Safety-critical software is software in which failure can result in direct injury to humans or cause severe economic damage (Turk et al. 2002). In security engineering, the home grounds of the agile and disciplined approaches are considered far from each other as a high level of discipline is needed throughout the development. The process standards for safety-critical software are often more strict than, for example, ISO9001 (S77) as they require a robust development process that ensures the quality and safety of the product (S76).

Even though developers of safety-critical software are experimenting with agile approaches (S65, S73), only two case studies of the reviewed papers included a company developing a safety-critical product (S24, S76). They use elements of, respectively, Scrum and XP while complying with the US Food and Drug Administration's (FDA) standard for software development. These cases are proof that it is possible to implement agility in a safety-critical software development process. This proof is supported by other researchers. According to Beznosov and Kructhen (S6), pair programming naturally facilitates internal design and code review and motivates developers to follow coding standards. Wäyrynen et al. (S75) also conclude that XP is aligned with security engineering.

However, the review also shows consensus among the researchers on the fact that agile approaches in their pure form are not suited for developing safety-critical software. Beznosov (S5) introduces eXtreme Security Engineering (XSE), an application of XP practices to security engineering. Boström et al. (S13) also proposes extending XP practices, while Siponen et al. (S63) give an example of how to add security techniques to agile approaches in general.

The researchers have different advice on how to fit the agile methods. For example, Pohjola (S58) concludes that the test-driven development practice of XP is the key. Lindvall et al. (S32) gathered the experiences on agility at a workshop. They found that safety-critical projects can be conducted using agile approaches; the key is to make the performance requirements explicit and plan proper levels of testing early in the process. In order to introduce agility in the development processes of safety-critical software in general, a body of evidence that agile approaches provide secure software is needed (S73).

4 Discussion

Addressing the research question, this section discusses the combinability and compatibility of the agile and the disciplined approaches. It discusses the guidelines and challenges identified in the literature on obtaining compatibility between the two approaches. Furthermore, four propositions created based on the analysis and discussion is presented.

4.1 Compatibility, Not Combinability

The analysis showed that the agile and the disciplined approaches are highly compatible, that is, in practice it is possible to implement several agile practices and principles in a software development process and at the same time be able to comply with a quality assurance standard. The agile and the disciplined approaches are however too different to be combined, meaning, that the agility of a project and/or the practices and principles of an agile software development process will be harmed in some degree by the regulations of a disciplined quality standard or a process model.

Proposition 1: The agile and the disciplined approaches are compatible, but not combinable.

Taking into consideration that the agile software approaches were developed as a counterpoint to the heavyweighted traditional, disciplined software approaches, this proposition seem natural but however important for both practitioners and researchers. It is important that practitioners understand the consequences of introducing a quality standard in their agile process or that it may be possible to heighten the agility of a software development process complying with a quality standard. For further research, it is important to understand the difference between combinability and compatibility when comparing an agile and a disciplined approach.

4.2 *Obtaining Compatibility*

Several researchers focus on how to extend either the agile approaches, the disciplined approaches or both of them in order to make them compatible. Most researchers focus on how to extend the agile approaches, whereas only two see a need in extending the disciplined approaches. Researchers have different ways of extending the approaches.

Four papers make suggestions on extending XP. Beznosov (S5) focuses on extending XP for security engineering. Nawrocki (S45) proposed three modifications to XP: the written documentation and requirements are to be managed by a tester, the planning game is to be modified so that it allows having multiple customer representatives and a requirements engineering phase in the beginning of a project to provide wider perspective of the product being developed. Boström et al. (S13) focus on the planning game as a way of extending XP practices. Nawrocki et al. (S47) have built a maturity model for XP (XPMM), a 4-level maturity model with a structure that resembles CMMI. Their aim was to build a maturity model that is simple and lightweight. Two papers focus on extending Scrum. Zanatta (S78) has developed an extension, called xScrum, which consists of guidelines that allow Scrum to be compatible with the requirements management and requirements development process areas of CMMI. Marcal et al. (S38) suggest that few adaptations on Scrum will make it much more compatible or even fully combinable with CMMI project management process areas. Manzoni and Price (S37) focus on extending RUP and state that an organization must customize its own practice and develop its own procedures to satisfy key practices of CMMI not supported by RUP. Zuser et al. (S79) propose a standard for quality support in software process models. Port and Bui (S59) introduce two strategies to obtain compatibility of the agile and disciplined approaches. One adding cost-benefit to the agile approaches, the other modulates development iteration size to maximize the expected cost-benefit for each iteration. Visconti and Cook (S72) have developed an ideal process model for agile approaches.

Laurila (S30) suggests the creation of a CMMI-like framework that is more compatible with agile ideas. Instead of a plan-driven focus, the framework should embrace change. Stålhane and Hanssen (S64) suggest that the guidelines of ISO should include more specific guidelines, for example, a definition of acceptable reviews.

Risk management is another approach that can assist the process of obtaining compatibility between the agile and the disciplined approaches and help organizations answer the question of how much documentation is enough (S7). Boehm and Turner (S9, S10, S11) therefore suggest a risk-based approach when incorporating both agile and disciplined practices and principles. They have developed a five-step model. Galal-edeen (S19) discusses two approaches for obtaining compatibility, the ambidextrous organization approach and the risk-based approach. They found the risk-based approach of Boehm and Turner most useful, as the home grounds concept is practical when dealing with development projects. Geras et al. (S20) also concludes that the home grounds of Boehm and Turner represent a way to obtain compatibility between the agile and the disciplined approaches.

Proposition 2: Obtaining compatibility of the agile and the disciplined software development approaches requires an extension of either one or both approaches.

4.3 *The Challenges for Obtaining Compatibility*

The analysis revealed several challenges when using an agile software development approach and at the same time being compliant with a quality assurance standard. The two main challenges identified in the literature are on the issues of handling the documentation and the requirements. This section deals with these two issues and the advice presented by the literature to overcome these.

4.3.1 The Documentation

A difference between the disciplined approaches controlled by quality standards and the agile approaches is the amount of documentation required. Agile approaches do not support the degree of documentation demanded by disciplined approaches (S8, S53, S67, S78). The different focus on documentation is also reflected in the agile manifest which says ‘working software over comprehensive documentation’ (Beck et al. 2001). In the case study presented by Kähkönen and Abrahamsson (S28), they had to do some additional documentation to comply with the standard. According to Nawrocki et al. (S45), the lack of documentation is the main weakness of XP. Fortunately, both agile (S67) and disciplined (S3) approaches are highly flexible and adaptable. Some researchers state that documentation does not contradict the core principles of agility. Bos and Vriens (S12) argue that it is not necessary to write piles of documentation to comply with CMM level 2, and according to Diaz (S16), it is proven that the CMMI model can be applied in a light manner.

Research provides experiences and guidelines on how to deal with the amount of documentation. Stålhane and Hanssen (S64) suggest adding activities such as review meetings and writing design documents to the agile approaches, while still keeping the most agile ideas such as short iterations, building in increments and including the customer. The case study of Namioka and Bran (S43) showed how the short iterations and continuing updates to the documents enhanced the flexibility. Other guidelines are to treat the written documents as deliverables at the end of an iteration (S43) and to provide just enough documentation to help with enforcement of existing processes (S41). Process standards do not explicitly state how much documentation is required, and many companies end up doing more documentation than needed.

Previous research also suggests extending the agile approaches to have a stronger emphasis on documentation. In parallel, disciplined approaches need to keep the documentation to a minimum (S54). The key to making the agile and the disciplined approaches compatible on the issue of documentation seems to be a happy medium between focusing on working software and on writing documents.

Proposition 3: To obtain compatibility between the agile and disciplined approaches, the focus needs to be on both working software and on documentation.

4.3.2 The Requirements

The agile method XP bases the requirements on user stories created by the customer (Beck et al. 2001); these differences in how the agile and the disciplined approaches handle requirements can cause problems in obtaining compatibility (S8). According to Nawrocki et al. (S45), a weakness of XP is that this approach lacks documentation of requirements. User stories written in a plain business-like language cannot be used directly as requirements by a company wishing to comply with a process standard (S5, S77). Instead, each user story needs to be translated into functional test cases (S5). Nawrocki et al. (S45) propose a modification of the XP life cycle introducing a requirements engineering phase at the beginning of a project and to make the tester responsible for managing the requirements.

Several tools supporting the XP process has been developed (S77). Melis (S42) suggests using such tools to manage the gathering of requirements and planning activities.

Proposition 4: The different strategies for handling requirements proposed by the agile and the disciplined approaches can cause problems when trying to obtain compatibility.

5 Conclusions

A large number of software companies have a desire to adopt an agile software development approach and at the same time comply with a quality standard. The two approaches seem contradicting, but case studies which successfully balance agility and discipline are emerging. Therefore, the purpose of this chapter is to determine whether the agile and the disciplined software development approaches are combinable or just compatible. This is done by reviewing previous research comparing the agile and discipline approaches for analysis or through a case study. The review includes 79 papers which were analysed using conventional content analysis and in this process coded using Atlas.ti V6.

The analysis showed that the agile and the disciplined approaches are highly compatible, but not combinable. In practice, it is possible to implement several agile practices and principles in a software development process and at the same time be able to comply with a quality assurance standard; however, the agility of a project and/or the practices and principles of an agile software development process will be harmed in some degree by the regulations of a disciplined quality standard or process model.

The agile approaches are mostly compared with CMMI level 2 or 3 or ISO. Few cases prove that it is possible to use parts of an agile approach with CMMI level 4 or 5. Agile practices and principles can also be introduced in the software development process when developing safety-critical software in order to make the development process more flexible. The research however also agrees that obtaining compatibility between the agile and the disciplined approaches is not straightforward. The research provides guidelines and different models on how to extend the agile approaches in order to make them fit the disciplined approaches. The amount of documentation and the way the requirements are handled seems to be the biggest difference between the agile and the disciplined approaches which proposes some challenges. Previous research does not distinguish between combinability and compatibility of the agile and the disciplined approaches; hence, the researchers are advised to reconsider their terminology when comparing the agile and the disciplined approaches and adopt the definitions of compatible and combinable presented in this chapter.

Only few empirical studies have been conducted on software development teams attempting to use an agile software development process in a disciplined setting. The research within this area would therefore gain from further empirical data on the compatibility of the two approaches and on the strengths, weaknesses and pitfalls when trying to obtain compatibility between the agile and the disciplined approaches.

Appendix A. Studies Included in the Review

See http://heeager.com/files/combinability_or_compatibility_studies.pdf

Appendix B. Coding Schemes

See http://heeager.com/files/combinability_or_compatibility_coding_schemes.pdf

References

Beck K, Andres C (2004) Extreme programming explained: embrace change. Addison-Wesley Professional, Reading

Beck K, Beedle M, van Bennekum A, Cockburn A, Cunningham W, Fowler M, Grenning J, Highsmith J, Hunt A, Jeffries R, Kern J, Marick B, Martin RC, Mellor S, Schwaber K, Sutherland J, Thomas D (2001) Manifesto for agile software development. Available at: http://agilemanifesto.org/. 30 June 2011

Boehm B, Turner R (2003a) Observations on balancing discipline and agility. In: Proceedings of the agile development conference. IEEE Computer Society, Salt Lake City, pp 32–39

Boehm B, Turner R (2003b) Using risk to balance agile and plan-driven methods. Computer 36 (6):57–66

Boehm B, Turner R (2004) Balancing agility and discipline: evaluating and integrating agile and plan-driven methods. In: Proceedings of the 26th international conference on software engineering. IEEE Computer Society, Washington, DC, pp 718

Dahlberg H, Ruiz FS, Olsson CM (2006) The role of extreme programming in a plan-driven organization. In: The transfer and diffusion of information technology for organizational resilience: IFIP TC8 WG 8.6 international working conference. Springer, Galway, pp 291

Fritzsche M, Keil P (2007) Agile methods and CMMI: compatibility or conflict? e-Inform Softw Eng J 1(1):9–26

Hashmi SI, Baik J (2007) Software quality assurance in XP and spiral-A comparative study. In: International conference on computational science and its applications, 2007. ICCSA 2007. IEEE, Fukuoka, pp 367

Heeager LT, Nielsen PA (2009) Agile software development and its compatibility with a document-driven approach? A case study. In: Australasian conference on information systems, Melbourne, Australia, pp 205

Hsieh HF, Shannon SE (2005) Three approaches to qualitative content analysis. Qual Heal Res 15 (9):1277–1288

Kähkönen T, Abrahamsson P (2004) Achieving CMMI level 2 with enhanced extreme programming approach. Lect Notes Comput Sci 3009:378–392

Marçal ASC, de Freitas BCC, Soares FSF, Furtado MES, Maciel TM, Belchior AD (2008) Blending scrum practices and CMMI project management process areas. Innov Syst Softw Eng 4(1):17–29

Muhr T (1991) ATLAS/ti – a prototype for the support of text interpretation. Qual Sociol 14 (4):349–371

Nawrocki JR, Walter B, Wojciechowski A (2002a) Comparison of CMM level 2 and eXtreme programming. Lect Notes Comput Sci 2349:288–297

Nawrocki JR, Jasiñski M, Walter B, Wojciechowski A (2002b) Combining extreme programming with ISO 9000. Lect Notes Comput Sci:786–794
Nawrocki J, Olek L, Jasinski M, Paliswiat B, Walter B, Pietrzak B, Godek P (2006) Balancing agility and discipline with xprince. Lect Notes Comput Sci 3943:266–277
Nerur S, Mahapatra RK, Mangalaraj G (2005) Challenges of migrating to agile methodologies. Commun ACM 48(5):78
Opelt K, Beeson T (2008) Agile teams require agile QA: how to make it work. An experience report. In: Agile, 2008. AGILE'08. IEEE Computer Society, Toronto, pp 229
Pikkarainen M (2009) Towards a better understanding of CMMI and agile integration-multiple case study of four companies. In: Product-focused software process improvement: 10th international conference. PROFES 2009, Springer, Oulu, pp 401
Pikkarainen M, Mäntyniemi A (2006) An approach for using CMMI in agile software development assessments: experiences from three case studies. In: SPICE 2006 conference, Luxemburg, pp 4.
Reifer DJ (2003) XP and the CMM. IEEE Softw 20(3):14–15
Rönkkö M, Jarvi A, Makela MM (2008) Measuring and comparing the adoption of software process practices in the software product industry. Lect Notes Comput Sci 5007:407–419
Strauss AL, Corbin J (1990) Basics of qualitative research: grounded theory procedures and techniques. Sage, Newbury Park
Theunissen WH, Kourie DG, Watson BW (2003) Standards and agile software development. In: Proceedings of the 2003 annual research conference of the South African institute of computer scientists and information technologists on Enablement through technology. South African Institute for Computer Scientists and Information Technologists, Republic of South Africa, Johannesburg, pp 178
Turk D, France R, Rumpe B (2002) Limitations of agile software processes. In: Third international conference on eXtreme programming and agile processes in software engineering, Citeseer, pp 43
Turner R (2002) Agile development: good process or bad attitude? Lect Notes Comput Sci 2559:134–144
Turner R, Jain A (2002) Agile meets CMMI: culture clash or common cause? Lect Notes Comput Sci 2418:153–165
Vinekar V, Slinkman CW, Nerur S (2006) Can agile and traditional systems development approaches coexist? An ambidextrous view. Inf Syst Manage 23(3):31–42
Webster J, Watson RT (2002) Analyzing the past to prepare for the future: writing a literature review. MIS Q 26(2):13–23
Wehmeier S (2010) Oxford advanced learner's dictionary. Homepage of Oxford University Press. Available at: http://www.oxfordadvancedlearnersdictionary.com/. December 2010
Zanatta AL, Vilain P (2006) Extending an agile method to support requirements management and development in conformance to CMMI. HIFEN 30:58

Chapter 5
Expectations and Reality: Why an Enterprise Software System Did Not Work as Planned

David Greenwood and Ian Sommerville

1 Introduction

Information system development is the process of conceiving, analysing, designing and implementing an information system (Avison and Fitzgerald 2006). The process of implementation comprises the deployment, adoption and routinisation of a system within its environment. Implementation can be challenging because of problematic sociotechnical issues. In order to minimise these issues, methods of social analysis such as Coherence (Viller and Sommerville 1999, 2000), Ethics (Mumford 1995), Multiview (Avison et al. 1998) and I* (Yu 1997) have been developed to inform the analysis and design of systems.

There is recognition within the sociotechnical systems community that practitioners' needs are not being met by these analysis and design methods (Bygstad et al. 2010). This is motivated by their lack of use by industry and recognition of the 'design fallacy'. The design fallacy is the assumption that the primary solution to meeting users' needs is to develop ever more sophisticated social analyses to inform the design of systems (Stewart and Williams 2005). Qualitative studies of system development show that implementation outcome is not only influenced by a system's design but is also strongly influenced by groups of people ('technology mediators') shaping the familiarisation of a system and the ability of organisational and technical infrastructure to facilitate the adaptation of the system and associated work practices to users' circumstances during implementation (Anderson et al. 2005; Williams et al. 2005; Orlikowski et al. 1995; Doherty and King 2005).

This aspect of implementation is largely unexplored from an engineering perspective. As a result we (as a community) are devoid of engineering approaches to identify sociotechnical issues that may be inhibiting adaptation and familiarisation.

D. Greenwood (✉) • I. Sommerville
School of Computer Science, University of St Andrews, St Andrews, UK
e-mail: dsg22@cs.st-andrews.ac.uk; ifs@cs.st-andrews.ac.uk

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_5,

Similarly we have yet to develop knowledge of the most effective and efficient combinations of organisational and technical infrastructure (e.g. IS steering groups, user groups, wikis and so on) that facilitate adaptation and familiarisation.

In this chapter we present our work towards the development of an approach to identify sociotechnical issues that may be inhibiting the implementation of a system. The approach elicits issues that may be making the adoption of a system problematic as well as eliciting factors that may be inhibiting these issues from being resolved by familiarisation and adaptation.

In Sect. 2 we describe the types of factors that affect the adoption of a system, and we suggest that an approach based on activity systems theory provides a suitable analytical framework for analysing these types of factors. In Sect. 3 we present our research design describing the analytical framework that we adopted, the organisation in which the case study took place and our data collection and analysis approach. In Sect. 4 we demonstrate that the approach diagnoses problematic interactions between a system under investigation and structural-intentional elements in the situation. In Sect. 5 we discuss the approach within the context of existing social analysis approaches to inform systems development.

2 Social Analysis to Inform System Deployment and Adoption

Quantitative and qualitative studies of system adoption demonstrate a strong relationship between system adoption and structural-intentional elements. For instance the two best established quantitative models of system adoption or success suggest the importance of intentional factors, e.g. beliefs and intentions with respect to a system. The DeLone McLean model (Delone and McLean 2003) highlights the importance of the relationship between users' intentions and their beliefs about system quality, service quality and information quality. Similarly the UTAUT model (Venkatesh et al. 2003) highlights the importance of the relationship between system adoption and users' beliefs about 'performance expectancy', 'effort expectancy', 'social influence' as well as 'facilitating conditions' such as appropriate technical and organisational infrastructure to support the use of a system. Qualitative studies also indicate the importance of structural-intentional elements. For example, (Anderson et al. 2005; Williams et al. 2005; Orlikowski et al. 1995; Doherty and King 2005) illustrate the role of institutional rules shaping the adoption and use of technology as well as the important role of groups of people ('technology mediators') shaping users' perceptions and adoption of technology.

Since studies indicate a strong relationship between system adoption and structural-intentional factors, we believe it may be fruitful for approaches that analyse problematic implementations to analyse these factors. Whilst there has been little work specifically directed at developing engineering approaches that inform deployment and adoption, there is an existing body of work that addresses the *troubleshooting* of problematic sociotechnical systems. This work has largely come from related fields outside of sociotechnical systems engineering such as ergonomics, CSCW (Computer-Supported Collaborative Work) and Soft OR (Operations Research). Some of the most notable approaches include situated action (SA) (Suchman 1987), distributed cognition (DC) (Hutchins 1995), activity systems theory (AST) (Engestrom 2000) and soft systems methodology (SSM) (Checkland 1999). Of these approaches, SA and DC do not analyse intentional level issues (Halloran 2000). We adopted a variant of AST because AST enables a more granular analysis of structural-intentional issues than SSM which is limited in scope to norms, values and roles.

3 Research Design

A case study approach was selected because our aim was to test the hypothesis that 'structural-intentional analysis enables the timely analysis of problematic enterprise system deployments'. A case study approach was deemed appropriate since our aim was to test our hypothesis in a real-world corporation with a problematic system. We collected our data using 16-hour-long semi-structured interviews of the system's stakeholders comprising open- and close-ended questions. We interpreted our data using a variant of cultural-historical activity theory called the 'activity space' (Halloran 2000, 2002).

3.1 The 'Activity Space' Framework

The 'activity space' framework (Halloran 2000, 2002) is a framework for structuring data and identifying problematic intentional and structural aspects of a situation. The framework comprises three constructs consisting of subjects, objects and mediators. Subjects comprise the actors involved in the situation. Objects represent the desired outcomes of a situation from each actor's perspective. Mediators comprise internal and external representations that influence the activity of actors. External representations are things in the material realm such as *tools*. Internal mediators consist of intentions and beliefs such as those about the *responsibilities*, *rules* (formal/informal norms) and *division of labour* (e.g. how activities are managed and how they contribute to the overall objectives of an organisation). According to 'activity space' theory, the outcome of a situation (e.g. a system deployment) is brought about by interactions between subjects, objects and mediators. Problematic situations can arise when incongruence (tensions) exists within and between actors' objects and mediators. By understanding these tensions a situation can be modified to change the outcome.

3.2 The Organisation and the Sociotechnical System

The fieldwork was performed at three different sites of a multinational system-engineering corporation that we will call 'Company A'. Their main work activity comprises the design, manufacture and maintenance of specialist electro-optical components and systems, e.g. infrared search and track systems. The organisation is divided into a number of functional groups that come together under a project structure to produce customer deliverables, e.g. components, systems and documents. The design of components and systems is a collaborative activity, and the sharing of documents is considered to be an important aspect of this activity.

'Company A' deployed an enterprise document management (EDM) system in the early 2000s as it was perceived by the IT director that an EDM system would be more advantageous than using shared folders on a file server to exchange documents. There was a perception that the introduction of the system would bring about greater visibility and awareness of work rather than having different teams and functions working in information silos. Within projects it was envisioned that EDM would be an up-to-date repository of all project documentation. Teams would store their documents in personal working areas and upload them to standardised locations in standardised EDM project file structures.

When we visited the organisation in 2010, the EDM was perceived by engineering management to be problematic due to 'sociotechnical factors'. The use of the system was mandatory, so all projects had an EDM project area but the extent that documents were being uploaded from working areas to the EDM project areas varied between teams. In addition to this, the use of the EDM file structure varied between teams, as did the location of files within the file structure. As our investigation unfolded it became clear that engineering management perceived the system to be problematic because teams did not use it in a 'common way'.

3.3 Data Collection and Analysis

We collected our data using 16 one-hour semi-structured interviews of the document management system's stakeholders. Interview participants were selected on the basis of availability by a facilitator within the organisation. The participants were of varying gender, with ages ranging from late twenties to nearing retirement, and seniorities/functional groupings ranging from programme management, engineering management, IT management, change administration and engineering. All the participants had worked at the organisation for at least a year, and some had been there for more than 20. The interviews comprised a set of open-ended questions and a set of close-ended questions derived from (Mumford 1995; Venkatesh et al. 2003; Gable et al. 2003). Interview questions are available at (http://www.cs.st-andrews.ac.uk/~dsg22/P/EDM_Survey.pdf).

Interviews were digitally recorded and transcribed when permitted. The open-ended interview questions were designed to elicit the relationship between the participant's view of their work (role, responsibilities, their day-to-day activities, most serious work challenges) and the deployed system (their history with the system, which responsibilities/activities the system helps them accomplish, how it does so, what problems it introduces to their work, how the system impedes their responsibilities and activities). The close-ended interview questions elicited the relationship between the participant and the system by exploring aspects of IT systems that are associated with intention to use (performance expectancy, effort expectancy, information quality, system quality, support quality, system usage policy) and aspects of organisational change that can lead to conflict (interfering with roles, goals, values, resources, capabilities/skills, job satisfaction, status, procedural justice, distributive justice, importance, ownership).

Dialogue mapping was then used to organise interview data into more abstract units of information. Dialogue maps were compared to the participant's responses to close-ended questions to corroborate findings. Dialogue maps were compared across participants to identify themes. The 'activity space' framework was then used to structure the findings and provide a lens for identifying tensions between different elements within the situation.

4 Our Findings

We report our findings in three parts: the first part identifies *software* usability issues, such as user interface (UI) issues, that frustrate the use of enterprise document management (EDM) system regardless of the software's specific configuration; the second part identifies *system* usability issues to do with the deployed configuration fitting the work environment; and the third part reports on the *structural-intentional* issues that frustrate the use of EDM. In the third part, EDM is viewed as a resource that mediates (enables/constrains/transforms) work activity. In contrast to the first and second parts, the issues raised will highlight underlying tensions that result in issues or challenges that impede the use of EDM in a 'common way' as desired by engineering management.

4.1 Software Usability Issues

We found that the following aspects confounded the usability of the tool in both experienced (Table 5.1) and novice/infrequent users (Table 5.2). The consequences of these issues were typically frustration and/or perceptions of wasted time.

Table 5.1 Aspects detrimental to usability according to *experienced users*

#	Aspects detrimental to usability according to experienced users
1	Requires a login separate from workstation login
2	Web-based interface is slow to respond to user interaction Screen updates and file uploads are perceived to be slow or freeze
3	Files can only be uploaded individually using the Web-based interface
4	Files are rendered poorly when viewed in using the Web-based interface
5	Search feature does not return expected results
6	Web-based interface has screen-rendering issues when used with browsers other than Internet Explorer 6

Table 5.2 Additional aspects detrimental to usability according to *new/infrequent users*

#	Additional aspects detrimental to usability according to new/infrequent users
1	Menus are cluttered and there is no obvious feature prioritisation to guide novice/infrequent users
2	Search query presentation is difficult to understand e.g. use of MIME types
3	The 'look & feel' of the Web-based interface is dissimilar to the 'drag & drop' interfaces that end users are generally accustomed to

End users found the above issues to be slightly problematic, but in general they perceive them to have a minor effect on their overall productivity, job satisfaction, speed of accomplishing work activity and effort to use EDM.

- Most participants surveyed agreed or strongly agreed that EDM takes little effort to use on their part. They reported that EDM did not *significantly* improve nor worsen their individual productivity; responses were mainly distributed around no effect, or slight positive or negative effects.
- Most participants agreed that EDM did not *significantly* slow down or speed up their speed of accomplishing activities. Responses were distributed equally between no difference, slower and faster.
- Most participants reported that EDM neither favourably nor adversely affects their job satisfaction. However, participants did report that the user interface does not meet their needs and is slightly problematic. And that the search facility does not meet their needs and is slightly problematic.

These mixed responses indicate that although the system has a number of frustrating and/or time-wasting usability issues, the majority of users we interviewed found that it did not significantly interfere with either their overall productivity or job satisfaction. These findings are perhaps surprising as management perceive the system to be problematic. This difference is explained by the fact that the extent that each team uses EDM is in accordance with their own approach, and therefore, they use the system in a manner that is acceptable to them (as a team) but not necessarily in a manner that is desired by engineering management.

Table 5.3 Aspects detrimental to system usability

#	Aspects detrimental to system usability
1	EDM is perceived to be more time consuming to use for storing documents in comparison to shared drives, or personal areas, due to the software usability issues identified
2	EDM has been configured to offer standardised folder structures; however, users struggle to understand where to put their documents within these structures. They perceive that there are a variety of possible locations, which make remembering and sharing the location of a document problematic. This interpretative flexibility enables the use of EDM in contrasting and inconsistent ways
3	EDM project areas have no built-in document registers, making it difficult to establish what documents are within a project area and which are missing. Lack of a document register is seen be problematic because of inconsistent use of the standardised folder structures which makes finding files on the basis on their expected folder location impractical
4	EDM has practical limitations on the number of files in a single folder as this can cause freezing or degrade the performance of the search facilities. This has been mitigated on the most part by using a folder structure
5	EDM runs on servers within a 'restricted' network and so cannot be accessed by all parts of the organisation. In a number of situations, this results in end users having to use other IT resources for document sharing, undermining the purpose of the EDM

4.2 System Usability Issues

We found that the following aspects confounded the usability of the tool in both experienced and novice/infrequent users (Table 5.3).

Despite the issues identified above, the participants that we interviewed reported that, in general, the use of EDM does facilitate their working practices, that they are supportive of continuing investment and development of EDM and that EDM is considered to be slightly important, or important, to their interests and responsibilities. This indicates that despite the systems shortcomings it was recognised by those we interviewed as a valuable tool that supports work. Again these findings may be surprising, considering that engineering management perceive the system to be problematic. However, this again highlights that end users use the system in a manner that is acceptable to them but not necessarily in a 'common way' as desired by engineering management. These findings indicate the problematic implementation is not explainable solely in terms of usability issues, and so we explore the situation from a structural-intentional perspective.

4.3 Structural-Intentional Issues

Overall we found that the extent and nature of EDM use vary on a project-by-project basis and that the nature of use is dependent upon individual programme managers and engineering teams. Engineering management perceive this to be problematic and at the time were pursuing an improvement strategy that encourages standardisation of EDM use. We analysed this problematic situation as the outcome of the following interacting elements: responsibilities, objects, tools, division of labour and rules.

Responsibilities

- There is a potential for conflict between the roles of engineering management and programme management. Whilst it is the responsibility of engineering management to take a strategic (long-term) view and run improvement projects. This is at tension with the tactical (shorter-term) responsibilities of programme management. Introducing change, even when successful, can cause short-term productivity degradation as changes are being 'bedded in'. This can be at odds with programme managers' contractual obligations such as milestones.

Objects

- The objects of engineering management, programme management and engineers are aligned such that their overall objects are positively dependent. This means it is in all parties' interests to coordinate their activities as one party's success contributes to the success of the other parties.
- Whilst objects are positively dependent, there is however scope for process conflict, and our study suggests that it may be occurring. For example, engineer managements' objective of improving delivery time is compatible with programme managements' objective of meeting contractual obligations; however, the way in which the objective is pursued, such as modifying the EDM, may interfere with meeting a contractual obligation if not carefully coordinated.

Tools

- EDM is acknowledged by all communities to suffer from usability issues at both the software and system level that ultimately results in frustration.
- These usability issues provided a motive in some communities to use a shared drive rather than EDM.

Division of Labour

- A matrix structure is a conduit of tensions. Engineers are within the focal point of matrix, and so they resist change when incompatible demands are placed.
- The division of labour can result in disconnects of responsibility or ownership. This has occurred with respect to the domestication of EDM.
- Communities have emerged around the division of labour and thus roles/responsibilities, e.g. programme management and engineers. There is a divergence between the viewpoints of each of these communities with respect to the value of EDM and the salience of its capabilities and purpose.

Rules

- There is a strong practice culture rather than a process culture within programme management and engineering. This means that work is performed on the basis of

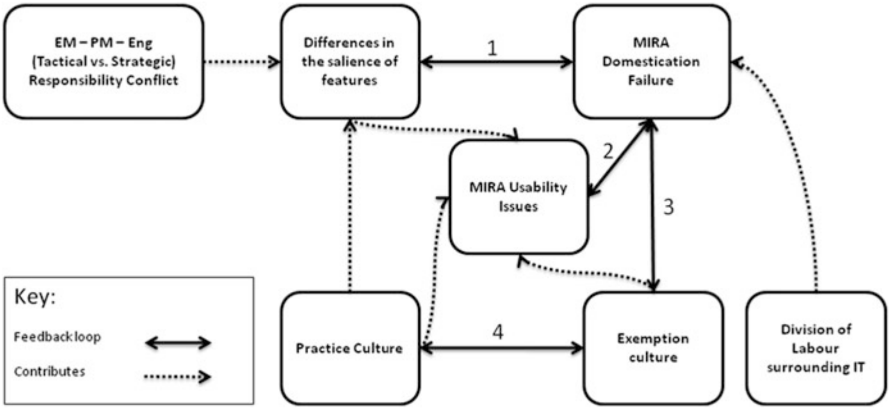


Fig. 5.1 Interactions of the sociotechnical issues identified

norms (e.g. individual and shared experience of what has happened in the past) rather than following explicit 'rules' (e.g. referring to process documentation).

We found that these tensions interacted to create four vicious circles (Fig. 5.1) that are contributing to sustaining the problematic situation (Forrester 1971).

The first vicious circle occurs between the differences in saliencies of features and the domestication of EDM. The programme managers and engineers do not value the visibility and control features of EDM as much as engineering management. This has resulted in the continued use of shared drives, rather than the use of EDM as intended, resulting in an absence of familiarity, adaptation and acceptance of EDM. Conversely as domestication has not occurred, programme managers and engineers did not have the opportunity to become familiar with the benefits and drawbacks of the visibility and control features.

The second vicious cycle occurs between domestication and usability issues. Because domestication did not occur, users experience usability issues due to lack of familiarity or because of lack of adaptation to the tool over time. Conversely, users experience usability issues because of lack of domestication.

The third vicious circle occurs between the exemption culture and domestication. The culture of allowing projects to decide on the extent and nature of EDM use (exemption culture) has resulted in a lack of familiarity with the full capabilities of EDM. Consequently EDM is not perceived as an acceptable substitute to shared drives. Conversely because EDM has not been domesticated, this has reinforced the culture of exemptions by permitting users not to use the tool.

The fourth vicious circle occurs between the practice culture and the exemption culture. As work is performed on the basis of norms (e.g. individual and shared experience of what has happened in the past) rather than explicit rule adherence, it has made it acceptable for projects to exempt themselves from standard ways of working such as EDM. Conversely since projects are permitted to exempt themselves from standard practices this reinforces, the 'practice culture' as enacting an exemption is in itself an exercise of the primacy of experience over standards.

5 Discussion

Our study was designed to evaluate the following hypothesis: ‘structural-intentional analysis enables the timely analysis of problematic enterprise system deployments’. Our study supports this hypothesis as we identified a number of tensions between structural and intentional elements and their interactions. Additionally we were able to make specific recommendations to ameliorate the deployment based on our findings. The recommendations we made comprised a set of steps to address the vicious circles identified. Each recommendation we made can be described codified commonsense. For example, address the exemption culture by identifying, implementing and enforcing a set of mandatory practices in coordination with programme management. For example, address lack of domestication by assigning the responsibility to a high-level manager and creating institutional infrastructures such as networks of champions and steering groups to guide familiarisation and adaptation of the work practices and the EDM. The value of the plan came from the fact that the *combination* of recommendations was tailored to the specific dynamics of the deployment environment.

We observed that structural-intentional analysis offers a number of useful trade-offs compared to task-centric analysis approaches such as SA and DC. Firstly, data collection can be of a shorter duration as a detailed understanding of tasks is not required. Secondly, the scale of the deployment under analysis can be much larger as data collection is rapid and data analysis can be supported through the use of off-the-shelf graph visualisation and analysis tools that support large datasets. This suggests that the approach fits well with the short time horizons and the size of system that industry works with. Thirdly, the approach focuses on system implementation, rather than analysis and design, which is typically when sociotechnical issues are observed and addressed in industry (Doherty and King 2005).

The disadvantage of a structural-intentional approach is that it will not deliver insights with respect to the subtleties of task-level interactions, e.g. distributed coordination, awareness, spatial-temporal organisation and so on. We believe that this trade-off is desirable as structural-intentional analysis is not intended to be an alternative to task-centric analyses such as SA and DC. In fact they identify different types of sociotechnical issue, so they may be considered complementary approaches to be used in combination when time permits.

6 Conclusion

This work illustrates that structural-intentional analysis is a promising candidate as a scalable engineering technique for analysing and troubleshooting problematic system implementations. Our case study indicates that structural-intentional analysis has a number of attractive characteristics with respect to timeliness and scalability. Data collection appears more rapid than task-centric analyses, and data analysis appears scalable as it can be supported through the use of off-the-shelf graph visualisation and analysis tools. Our fieldwork demonstrates that structural-intentional data is (1) sufficient to diagnose problematic interactions between a system, intentional elements and structural elements and (2) sufficient to suggest interventions to ameliorate a deployment. We therefore advocate a structural-intentional approach as a candidate engineering approach for analysing and troubleshooting enterprise system implementations. Our conclusion is limited by the usual limitations of case study research. Case study research may not be generalisable, and whilst every effort was taken to minimise investigator or participant bias, it may be reflected in our findings.

There are many opportunities to develop social analyses to inform/troubleshoot system implementation. We encourage more case studies or action research to develop a body of knowledge of typical problems and (un)successful attempts to resolve them by creating organisational and technical infrastructures to facilitate familiarisation and adaptation. We encourage attempts to explore the scalability of structural-intentional analysis and its ability to perform useful analysis in a variety of settings.

References

Anderson S, Hardstone G, Procter R, Williams R (2005) Down in the (data)base(ment): supporting configuration in organisational information systems. In: Ackerman MS, Halverson CA, Erickson T, Kellogg WA (eds) Resources, co-evolution and artifacts. Springer, London

Avison D, Fitzgerald G (2006) Information systems development: methodologies, techniques and tools. McGraw-Hill Higher Education, London

Avison D, Wood-Harper T, Vidgen R, Wood JRG (1998) A further exploration into information systems development: the evolution of Multiview2. Info Technol People 11(2):124–139

Bygstad B, Nielsen PA, Munkvold BE (2010) Four integration patterns: a socio-technical approach to integration in IS development projects. Info Syst J 20(1):53–80. doi:10.1111/j.1365-2575.2007.00280.x

Checkland P (1999) Systems thinking, systems practice includes a 30 year retrospective. Wiley, Chichester

Delone WH, McLean ER (2003) The DeLone and McLean model of information systems success: a ten-year update. J Manag Info Syst 19(4):9–30

Doherty N, King M (2005) From technical to socio-technical change: tackling the human and organizational aspects of systems development projects. Eur J Info Syst 14:1–5. doi:10.1057/palgrave.ejis.3000517

Engestrom Y (2000) Activity theory as a framework for analyzing and redesigning work. Ergonomics 43(7):960

Forrester JW (1971) Counterintuitive behavior of social systems. Theory Decis 2(2):109–140. doi:10.1007/bf00148991

Gable GG, Sedera D, Chan T (2003) Enterprise systems success: a measurement model. In: International conference on information systems, Seattle, Washington, DC, USA, pp 576–591. Association for Information Systems

Halloran J (2000) The activity space: analyzing intentionality in open cooperative work. Unpublished doctoral thesis, University of Sussex

Halloran J (2002) Taking the ‘No’ out of Lotus Notes: activity theory, groupware, and student groupwork. Paper presented at the CSCL ‘02, international conference on computer-supported collaborative learning

Hutchins E (1995) How a cockpit remembers its speeds. Cogn Sci 19(3):265–288

Mumford E (1995) Effective systems design and requirements analysis: the ethics approach. Palgrave, Basingstoke

Orlikowski WJ, Yates J, Okamura K, Fujimoto M (1995) Shaping electronic communication: the metastructuring of technology in the context of use. Organ Sci 6(4):423–444. doi:10.1287/orsc.6.4.423

Stewart J, Williams R (2005) The wrong trousers? Beyond the design fallacy: social learning and the user. In: Rohracher H (ed) User Involvement in innovation processes. Strategies and limitations from a socio-technical perspective. Profil-Verlag, Munich

Suchman L (1987) Plans and situated actions: the problem of human-machine communication. Learning in doing: social, cognitive and computational perspectives. Cambridge University Press, Cambridge

Venkatesh V, Morris M, Davis G, Davis F (2003) User acceptance of information technology: toward a unified view. MIS Q 27(3):425–478

Viller S, Sommerville I (1999) Coherence: an approach to representing ethnographic analyses in systems design. Hum Comput Interact 14:9–41

Viller S, Sommerville I (2000) Ethnographically informed analysis for software engineers. Int J Hum Comput Stud 53(1):169–196

Williams R, Stewart J, Slack R (2005) Social learning in technological innovation: experimenting with information and communication technologies. Edward Elgar Publishing, Cheltenham

Yu ESK (1997) Towards modeling and reasoning support for early-phase requirements engineering. Paper presented at the proceedings of the 3rd IEEE international symposium on requirements engineering, Washington, DC

Chapter 6
Supply and Demand of e-Government Services in Developing Countries: Cases from Tanzania

Bjørn Furuholt and Edmund Matotay

1 Introduction

Information technology (IT) has the potential to support development in developing countries, mainly through providing access to information and through building communication lines between people. In countries and regions with limited access to the Internet and other ITs, information for personal development, business start-up and growth or political participation is lacking, education is suffering and people cannot compete in the global economy. This leads to poor countries suffering greater economic downfall and thus stay in a vicious circle.

Therefore, one of the main contributions to create development in poor countries is to give people access to IT and the Internet, to information and communication. Information has become one of the primary inputs in economic processes, and information technology gradually becomes more crucial for the ability of enterprises, communities and individuals to participate successfully in the global economy (Hollifield and Donnermeyer 2003).

For most people in poor countries, it is out of the question to buy their own computer equipment, and access to this new technology is facilitated by arrangements for public use. These public Internet access points (PIAPs), variously referred to as information kiosks, telecentres, cybercafés, community technology learning centres and the like, have experimented to varying degrees of success with a variety of approaches (e.g. business models, service offerings, target populations, technologies employed) in service to their customers (Bell 2006). Public access points like telecentres and Internet cafés have been successfully spread to poor countries mainly because it combines a reasonably priced access to the Internet

B. Furuholt (✉)
University of Agder, Kristiansand, Norway
e-mail: Bjorn.Furuholt@uia.no

E. Matotay
Mzumbe University, Mbeya, Tanzania

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_6,

with the chance to socialise with fellow users and to pick up new knowledge and ideas in computer usage.

Telecentres operate mostly as not-for-profit organisations, relying on sources of external funding, including government institutions, multilateral agencies and nongovernmental organisations and with an explicit objective to support the community, which often mean to support development among underprivileged and marginalised populations. Internet cafés, on the other hand, are normally owned by local entrepreneurs and represent business opportunities for the owners. As private enterprises, their primary purpose is to generate profit for their owners. Both telecentres and Internet cafés vary considerably in size and variety of services. "Multipurpose community telecentres" (MCTs), for example, is an advanced concept promoted by the International Telecommunications Union (ITU), including facilities such as libraries, training workshops, seminar rooms and office space and providing services such as videoconferencing, distance education, training and e-commerce (Oestmann and Dymond 2001).

Today, Internet cafés and telecentres are the two main sources of public Internet access in developing countries, and these are the principal means of accessing the Internet in countries such as Tanzania, Indonesia and Peru, where as much as two thirds of Internet users get their access through public Internet access points (Wahid et al. 2006).

One area where information technology and public Internet access can be particularly useful for the development of poor countries is as a tool for promoting good governance through e-government systems. Kristiansen (2004:11) states that: "There seems to be a clear relationship between information asymmetry, corruption and bad governance". This view is supported by UNDP, which claims that:

> Access to information is a pivotal empowerment tool and underpins effective interventions in the area of democratic governance which is central to sustained poverty reduction and the achievement of the MDGs. It is vital for strengthening accountability, transparency, participation and rule of law. (UNDP 2010)

During the last 10–15 years, governments from all over the world have tried to take advantage of information technology in general and the Internet in particular to improve governmental administration and communication with their citizens and businesses.

Adoption of e-government has increased in most countries, but at the same time, the rate of adoption varies from country to country. Generally, developing countries are lagging behind in e-government adoption compared to developed countries, and among the developing countries, Tanzania and its East African neighbours are at the bottom of the United Nations' global e-government development rankings (United Nations 2010). Based on the 2010 Global E-government Survey, Tanzania was no 137 out of the 183 surveyed countries, Kenya no 124, Uganda no 142, Rwanda no 148, Malawi no 159, Mozambique no 161 and Burundi no 174.

It even shows that the Eastern African region showed little improvements and that the majority of the countries surveyed had a lower ranking in 2010 compared with 2005 and 2008. This lack of access for the millions of inhabitants of the region contributes to exclusion from the benefits of the information society.

If e-government services shall contribute to development in poor countries, as expected by scholars and policy makers, they must be accessed and utilised by ordinary citizens. This has to take part in public Internet access points, the only possibility for the majority of people in the least developed countries. In this chapter, we therefore investigate if public Internet access points are used for accessing e-government information and services in Tanzania today, to which degree there is a supply of and demand for these services, and we aim to improve our understanding of how the PIAPs can be used in order to contribute to good governance in developing countries in the future.

This chapter is organised as follows. After this introduction, we present an overview of relevant literature discussing e-government services and public Internet access in developing countries in general and in East Africa in particular. Section 3 presents the five cases studied, followed by a closer description of the findings from this work in Sect. 4, while Sect. 5 provide our conclusions, limitations and prospects for the next phases of this project.

2 Literature on e-Government and Public Internet Access

E-government can be defined broadly as the use of information and communication technologies in the public sector to improve its operations and delivery of services. Although e-government encompasses a wide range of activities and actors, three distinct sectors can be identified. These include government-to-government (G2G), government-to-business (G2B) and government-to-citizen (G2C) sectors (Furuholt and Wahid 2008:2).

The term e-governance has a wider meaning than e-government, and it is directly linked to the concept of good governance. UNESCO (2010) defines e-governance as:

> . . . the public sector's use of information and communication technologies with the aim of improving information and service delivery, encouraging citizen participation in the decision-making process and making government more accountable, transparent and effective.

According to Heeks (2001), e-governance lies at the heart of two global shifts: the information revolution and the governance revolution. Both shifts are changing the way society works and the way that society is governed. But it is only those who have access to IT and digital information and knowledge, who benefits from reforms in governance. We can thus talk of an "e-governance divide" that is increasingly separating developed and developing countries and elites and ordinary citizens within developing countries.

Some literature has dealt with the opportunities and challenges of e-government in developing countries, and it shows that the main opportunities for e-government in general, like cost reductions, improved efficiency and quality of services, also apply to projects in developing countries. Some motivation, however, seems a stronger and more important factor for transitional democracies and developing economies which may emphasise reforms such as transparency, increased citizen participation and attracting economic development (Seifert and Bonham 2003).

A common theme is the focus on transparency and fighting corruption. At a national level, Andersen (2009) claims that more e-government services online in a country result in less corruption. Grönlund et al. (2005) have studied a selection of handbooks for managing e-government projects in general, and for developing countries in particular, and found that, apart from a particular focus on corruption in the developing countries, these handbooks are strikingly similar. In another article, where Grönlund (2010) discusses how ICT is supposed to help in reducing corruption, he describes certain e-government systems designed with exactly the objective to combat corruption. The Bhoomi project in India, for example, was designed to facilitate online delivery of land records so that citizens could challenge arbitrary bureaucratic action if they deemed them to be unfair. From sub-Saharan Africa, Heacock and Sasaki (2010) report that almost none (three exceptions) of the countries in the region fall above the midway point of the 2009 Transparency International Corruption Perception Index, which measures how citizens perceive the level of government corruption. The corruption undermines political stability as well as the governments' capacity to provide effective basic services. On the other hand, a number of studies show how increased transparency leads to increased performance and responsiveness in both government and the private sector.

To be effective, e-government projects, like information systems in general, must focus on the social contexts into which they are introduced. This is even more important in developing countries, many of them African countries, with great cultural differences from the "western" world where the technology and systems normally are designed and developed. According to Heeks (2003), there is often a large design-reality gap when you try to introduce an e-government system designed in and for an industrialised nation into a developing country.

Schuppan (2009), as well, addresses the different institutional and cultural contexts which must be considered when implementing e-government in sub-Saharan Africa. Although e-government is a global phenomenon, simply transferring IT solutions and related organisational concepts from developed to developing countries seems inappropriate. More than in developed countries, the different initial institutional, cultural and wider administrative contexts must be considered to avoid unintended effects. Therefore, especially for African countries, a context-oriented approach seems to be a more promising route to the successful implementation of e-government.

Most of the literature covering public Internet access in developing countries has a supply-side focus. A successful distribution of e-government services depends, however, on both the supply and demand sides. Despite some initial success, e-government has yet to prove successful in developing countries. e-Government initiatives face serious challenges and broadly speaking, these challenges seem to fall into three categories: *management, infrastructure* and *human* factors (Furuholt and Wahid 2008). While the first category is solely linked to the supply side, the infrastructure and human factors belong to both the supply and demand side, that is, citizen issues. Kumar and Best (2006) point to the same challenges and failure factors and propose a sustainability failure model, where cultural and social sustainability is one of the principal modes. Few projects meet these challenges, and the failure rate is high. Heeks (2003) has analysed more than 40 e-government-for-development projects in developing countries and estimates that only 15 % were successes.

Also Sein (2009) points to critical developing country challenges, and he highlights the human factor. He states that the main hurdle has been that an overwhelming part of the citizens in these countries do not have the capability to either access government information physically or to use it effectively even when they can access it. He suggests that linking citizens to government in developing countries require an intermediary. Sein and Furuholt (2009) further develop this idea of intermediaries and suggest that this role could be played by telecentre and Internet café staff and thus that these public Internet access points have the potential to play an important role in the spread and use of e-government services.

Apart from the few articles mentioned above, very little research is published on e-government use in East Africa and Tanzania. The UN annual E-Government Development Survey (United Nations 2010) has been published at regular intervals since 2003 (see Sect. 1 above for East African details). The survey aims to identify and help address disparities among countries around the world, especially, in support of a move towards a more inclusive information society, as envisaged in the World Summit on the Information Society (WSIS). The survey tracks the progress of the member states in implementing e-government programmes. It compares their state of e-government readiness via a benchmarking tool and measures the use of the Internet and the World Wide Web (WWW) for the provision of information, products and services, plus the level of telecommunication and human capital infrastructure development.

E-Government services, in general, are normally offered through Internet portals hosted on a delivery Web server to provide access to the citizens through inexpensive Internet medium. From East Africa, Kaaya (2004) presents a survey, based on a content analysis to determine the status of government websites of three countries—Kenya, Tanzania and Uganda. The results were matched with a four-stage model of e-government growth based on the status of websites from simple to sophisticated features. The study identified 98 government websites including 33 for Kenya, 37 for Tanzania and 28 for Uganda. More than 83 % of the identified websites were established between 2000 and 2003, and their creators were still (in 2004) undergoing the learning experience. The study concluded that all of the East African websites are at the first and second stages of the website development and corresponding e-government services.

As mentioned above, improving transparency and fighting corruption has been main drivers when introducing e-government in developing countries. Schuppan (2009) describes implementation of a tax administration system in Tanzania, and he claims that this project has reduced corruption because the users now can go to a special dedicated one-stop office instead of having to contact back-office employees. Even if this is a governmental office, this public access point has increased transparency and tax revenue and reduced processing time and the possibility of fraud.

John Mahegere works for the Tanzanian government's Commission for Science and Technology (COSTECH) and is responsible for a governmental project aiming at creating a telecentre in each one of the country's 114 districts. In a study (Mahegere 2010), he presents a brief review of the technologies, the rural ICT projects and the associated impacts with the use of ICTs and telecentres in particular for rural e-governance applications. His report shows very few traces of actual e-government use, and he concludes that, so far, a large number of the rural ICT applications have slipped in performance and are facing acute problems of sustenance after their successful launch by the dynamic project champions.

A thematic report by Tan (2007) describes lessons learned from setting up and managing five rural communication access centres in Tanzania, including the Sengerema Telecentre (see below). From this study and subsequent discussions at a follow-up workshop, he presented a long list of lessons learned and derived, without even mentioning the role of these centres as e-government service providers, which clearly illustrates the need for making up for this shortage and study their potential as e-government intermediaries closer.

Our literature review thus reveals that there has been very little research which shed light on our research question and to build our work on. We, therefore, have to start with an explorative approach to our study of e-government use from public Internet access points.

3 Cases and Data Collection

In their study from Northern Colombia, Gomez and Baron-Porras (2010) found that public access venues such as telecentres, public libraries and cybercafés to a very small degree are used for government transactions. The same picture is described by Furuholt and Kristiansen (2007) from Tanzania, where they found very few traces of e-government use from Internet cafés. With a special focus on Tanzania, our next step was therefore to look into telecentres, the other main type of public Internet access points. The Sengerema Telecentre (STC) in Mwanza region was then chosen as our primary case because it is known as a model and a pilot telecentre in Tanzania. The other cases are the broadband networks in Bunda and Serengeti districts in the Mara region, and the Soma Book Café in Dar es Salaam, all in all three very different types of not-for-profit Internet access points. In addition to these, we also, recently, visited two ordinary Internet cafés in order to reveal a change towards more e-government use lately in this type of venues.

So far, with one exception, we have only interviewed the supply-side stakeholders, staff, management and support organisation. Data collection was conducted in September 2009–February 2011, using semi structured face-to-face interviews and informal focus group discussion with the actors. The interviews were carried out with the acting director of the Directorate of Information and documentation at the Tanzania Commission for Science and Technology (COSTECH), staff from the ICT for Rural Development (ICT4RD) research and development programme at the Dar es Salaam Institute of Technology, the STC manager, the Chalinze ICT owner/manager, the twins supervisor, and with the Soma Café manager and one of the users. The focus group discussions were made with the head of the operations and the IT manager at STC, in their premises.

The interviews and focus group discussions focused on the participants' experiences with e-government implementation and use and their opinion on the development of e-government services in the region and in Tanzania in general.

3.1 The Sengerema Experience

Sengerema Telecentre was initiated by (COSTECH) with support from the International Development Research Centre (IDRC) and other international donors back in 2001. At that time, it was the country's pilot rural telecentre.

Today, STC offers a wide variety of IT-related services, with many community development aspects, like an Internet café, where staff are instructed to help customers to operate the computers and to search for relevant information and services. They also run their own library and community radio station, in order to reach out to the rural villages with relevant information, training and distant learning courses, ISP services to various organisations (NGOs etc.), and they run ISP services for the local and district government. Their intention is to develop the governmental bodies to e-governments. They also offer a variety of IT consultant and secretarial services, like desktop publishing and web design.

The ISP services are their main source of income, together with the Internet café. An interesting feature of the centre in terms of staffing is that they to a high degree are relying on volunteers. Only five people have fixed position out of 18 employees. In addition to relying on volunteers and paid staff, the centre also gets assistance from local NGOs like women belonging to Sengerema Women Information and Communication Group, for dissemination of information and on local dance groups and musicians for creating awareness and promoting the telecentre.

3.2 The Bunda and Serengeti Networks

The starting point for these projects was the support provided by Swedish International Development Agency (SIDA) to the Tanzania Electrical Supply Company (TANESCO) to link up 12 villages in Bunda and Serengeti districts in Mara Region with the national electric grid (Mascarenhas and Kimasha 2007). A fibre optic cable was included in the electrical overhead cables provided for this project primarily to monitor the supply of electricity. However, the capacity was large enough to accommodate other forms of data transfer and Internet connectivity using broadband.

IC-Tanzania is a national committee of stakeholders in the IT sector with representatives from public and private institutions as well as development partners. One of its projects is ICT4RD, a 4-year programme with the objective to explore the possibilities of having affordable Internet connectivity in the rural areas in Tanzania. The essence of the project is to test different infrastructures, technical solutions and different business models and to investigate usage and impact of the connectivity.

The networks are connecting the headquarters of the two municipalities, the Serengeti district in Mugumu and Bunda district in Bunda. The primary focus of the network is to connect the two local government authorities, some education institutions and health facilities in the two municipalities. The network is hosting a website[1] where it's content management is being done in Bunda, a local e-mail server, telecommunication services via voice over IP and a wireless LAN across and between the connected centres. So far the two municipal councils are connected together with one primary health centre and one secondary school.

3.3 Soma Book Café

Soma Book Café in the Kinondoni area of Dar es Salam is run by Soma,[2] a non-profit organisation with a vision to fight poverty by transforming Tanzania into an informed society that values knowledge, creativity and independent thinking. The café premises house a bookshop and a mobile library, with a focus on African culture, café services based on East African cuisine, a number of computers with Internet connection and a wireless network for rent for visitors bringing their own equipment. They run various educational and networking forums and publish a quarterly literary magazine. The Soma organisation receives some external financial support from Swiss Development Cooperation and other donors.

3.4 Twins Internet Café and Secretarial Bureau

Twins Internet café is a solely private entity, localised in the Mwenge area of Dar es Salaam, surrounded by educational institutions (university, colleges and schools). The users are mostly students, especially from secondary schools, but also workers (public and private), doctors and lawyers are regular customers. Doctors, for example, are usually looking for medical information.

---

[1] www.bunda.go.tz

[2] http://www.soma.or.tz/soma/english/index.php?option=com_frontpage&Itemid=1

On average, Twins is visited by 50 customers per day, and the average time spent by the customers is 1 h. One-hour fee is Tsh 1,000 (approximately USD 0.65).

3.5 Chalinze ICT

Chalinze ICT is the only Internet café in Chalinze, a town at a road junction about 120 km west of Dar es Salaam. The owner, who opened the café in 2008, is also the manager and operator, and he is a student at a nearby university. He has hired two employees to help him run the business.

There are less than ten Internet access customers per day, paying Tsh 1,000/hour, but the café is also offering other services, like fax, copying, and typing.

The six computers were initially donated by COSTECH, and the rest of the start-up capital came from the owner's own savings.

4 Findings

Sengerema Telecentre (STC) is providing Internet services for the local governments, as an ISP, in order to make them capable of offering e-government services. This is their first step towards helping the local government to building up an effective e-government service. The second step has been to support the local and regional governments when developing their Web presence, and the third step has been to make these services available for citizens, by running the Internet café services and by supporting other Internet cafés in the region in their supply of Internet-based services. Today we can see some but very weak traces of use of e-government services from the demand side (G2C), limited to some access to static websites with governmental information. The STC management, however, has a clear vision of providing the local community with relevant content in collaboration with the local authorities and educational institutions.

In the Bunda and Serengeti districts, the use of the Internet is, in general, low. There is no local ISP and not a single Internet café. Within the local government offices, however, there is adequate equipment, and many of the district staff have been trained in the use of computers. There is also some use of the Internet. It is being used by departments in both district councils, by the district hospital in Bunda district and by businesses like Varrian Tanzania and the Bunda Oil Mills. Website information updates and changes are still poor. It took, for instance, 9 months to change or update the current information in the website. Moreover an informant has revealed that:

> ... still there is a problem of community awareness of this important community project, People don't own the project; they perceive it as a government project, for the government and not the community purpose, only students and youth see it as an opportunity.

The establishment and deployment of the technology in between and across these two municipalities had in mind the whole aspects of enhancing e-government in the municipalities. But so far, it is only government institutions that are connected (G2G). The main informants claimed, however, that:

> if you serve these municipal councils and their respective health and education institutions very well, then, then the community will benefit through improved services, and you have served the people. (G2C)

While the two first cases are mainly supply-side oriented in their e-government approach, Soma Café is focusing on the demand side (citizens). Their vision is to empowering people by giving them access and inspiring them to use it for knowledge development and e-participation. People are coming to the café both for pleasure and work, and the manager told us that they (the Soma organisation) look at the café as a "community knowledge centre", recognising that "information is power". They promote lifelong learning and aspire to take the role of "intermediary", in order to proactively supporting their clients in debating, networking and building up an active reading culture on the Internet.

Herry is a secondary school student and a native Maasai. He is a regular customer at Soma using the net mainly for surfing and e-mailing. He was not aware of the term or the concept of "e-government", but when he was asked what sort of information he was looking for, he mentioned secondary school results and colleges and universities applications:

> . . . once my sister, who are residing in the rural areas sent me exams numbers of her and neighbors children, so that I can help them in search for their secondary schools results. I usually come here at SOMA, and then send the results to my relatives. And: . . . I usually visit the website that secondary school students used this year to apply for their universities and colleges admissions.

In order to find this and other information, he uses Google:

> When we want to search for websites like Tanzania Commission for Universities (TCU), and the National Examination Council of Tanzania (NECTA), we usually Google which helps us a lot.

Even if all three telecentres had explicit development objectives and had high ambitions regarding use of e-government services and e-participation for their clients, we found, so far, very few traces of active use of such services when we visited them. What we found was more random use initiated by the users themselves when they were looking for various types of information, using search engines like Google.

The same picture was found in the recently visited Internet cafés. In Twins Internet café in Dar es Salaam, we met with Stellah, the café supervisor. She had only recently heard about "e-government" and had to look at Wikipedia to learn more about it when we asked her. After that, she told that the most popular e-government service among her customers is:

> . . . the download of forms for joining colleges and universities, application for loans from the higher education loan board, schools and universities admission results, midterm results

> for universities like University of Dar es Salaam, and application for international universities. ... Some people also come here to check for information in the ministry of tourism, industry and trade, local government, and ministry of land to check for some information I presume are business oriented though am not very much sure of it

John is the owner and manager of Chalinze ICT. He has been a teacher, and he is now a bachelor student at a nearby agriculture university. He was neither not aware of the concept of "e-government", but after a while, when we were discussing his customers' use of the Internet, he told us an interesting story about farmers who needed information about a chicken disease and how this initiative lead to information exchange between farmers, local agriculture agencies and central authorities, all communication over the Internet, in other words, relevant e-government services for rural farmers. Also Herri (above) mentioned the need for agricultural services, like information on contour farming, irrigation and new equipment, as one of the most important possible e-government services in Tanzania.

As far as we understand the reason for this lack of systematic use of e-government services is due to both infrastructure (lack of computers and lack of or slow and unstable connections) as well as management (lack of marketing, lack of relevant information and services and human factors and lack of awareness and knowledge among the citizens).

5 Conclusions and Recommendations

This chapter describes research in progress which started with asking if public Internet access points are used for accessing e-government information and services in Tanzania today and to which degree there is a supply of and demand for these services. As it appears from the cases described above, the use is highly limited, if any at all, and the reason for this has several explanations, from the supply side as well as from the demand side, based on management, infrastructure and human factors.

There is a pressure from international organisations and national NGOs to offer e-government services in developing countries. This pressure is forwarded from central governments to local government agencies. As a general experience, however, it seems like the G-side in the G2C context has focused very little on how the citizens may achieve access to the services. They offer it and that's it. It seems to be the citizen's own responsibility, or up to "intermediaries" like telecentre staff, to arrange for the access and to make efforts in order to utilise the information and services. In our opinion, the public sector has to take on responsibility also for the demand side. This can be done through a public/private partnership with these potential intermediaries.

Our first recommendation has, therefore, to do with the general attitude towards Internet access in general and e-government services in particular. If access to Internet services is as important an advance for the development of poor countries as the UNDP claims (see Sect. 1, above), then we contend that governments should consider Internet access as a “public good”, an essential part of the local, community infrastructure like electricity or schools. To this end, governments could collaborate with private companies or international organisations and could provide the infrastructure for access in all parts of the country, like we see some signs of in the cases, in particular in Sengerema.

The second recommendation deals with the public sector’s attitude towards Internet access points. Some authors suggest that public telecentres should be developed in the same manner as Internet cafés by using the same kinds of business properties. Our suggestion is broader, by including privately owned Internet cafés in this partnership as well. We therefore propose that the public and private sectors establish a partnership, where the public sector supplies the thousands of already existing privately owned Internet cafés with Internet-based e-government content and services. These services in turn would benefit both the private and public sectors. This would also give more benefits to the users and thus help build up a basis of sustainable Internet access points.

From the demand side, we see that people (both ordinary citizens and PIAP staff) do not know about e-government; they do not know the concept, and they are not aware of what services governmental agencies can offer. On the other hand, we have found a demand among people. Farmers find useful agricultural information and services through the Internet, and pupils and students as well as teachers, doctors and lawyers find relevant information and services when they need it. From the literature, we know that PIAPs have to adapt to this local demand in order to be a sustainable tool for development (Furuholt 2009; Masiero 2011). Also Prado (2010) shows that the longest continuous-running independent telecentre in the Dominican Republic (established in 1997) has been sustainable because of local adaption and access to relevant public information about health care, farming, etc. A public private partnership like we suggested above could facilitate this supply of local information.

One interesting example of local adaption, and an idea worthwhile trying out at a larger scale, is described by Elijah (2010). In order to increase information dissemination from federal agricultural government research institutes to smallholder farmers in a village in southeastern Nigeria, a special-purpose telecentre dedicated to agricultural knowledge and information was established in 2007. The knowledge centre operated in a convenient location in the village comprised of a customised website, computers with high-speed Internet access and a full-time on-site staff, and it was positioned as a social hot spot for people to exchange ideas, obtain information and assistance from one another.

Our next research step will address other stakeholders, namely, central and local government agencies, policy makers and users and potential users to improve the understanding of how the public Internet access points can be instrumental in order to increase the use of e-government services with an objective of contributing to good governance in developing countries in the future.

References

Andersen TB (2009) E-Government as an anti-corruption strategy. Inf Econ Policy 1(3):201–210

Elijah OA (2010) Effects of the emerging information communication technologies (ICTs) on agricultural knowledge transfer to smallholder farmers in Nigeria. IFIP WG 9.4 Newsl 20:2

Furuholt B (2009) Bridging the digital divide: sustainable supply and demand of internet access in developing countries. Publication no. 49. Department of Computer Science, Aalborg University, Denmark

Furuholt B, Kristiansen S (2007) A rural-urban digital divide? Regional aspects of internet use in Tanzania. Electron J Inf Syst Dev Ctries 31(6):1–15

Furuholt B, Wahid F (2008) E-government challenges and the role of political leadership in Indonesia: the case of Sragen. The 41st Hawaii international conference on system sciences (HICSS), Hawaii, US, 7–10 Jan 2008

Gomez R, Baron-Porras LF (2010) Public access to the internet and social change: an experience in Colombia, between silence and hope. J Community Inform 6(3):1–23

Gorla N (2009) A survey of rural e-government projects in India: status and benefits. Inf Technol Dev 15(1):52–58

Grönlund Å (2010) Using ICT to combat corruption – tools, methods and results. In: SPIDER: increasing transparency & fighting corruption through ICT. SPIDER ICT4D series no.3, Stockholm University

Grönlund A, Andersson A, Hedström K (2005) NextStep e-government in developing countries. Project report. Örebro University, Sweden

Heeks R (2001) Understanding e-governance for development. iGovernment working paper series. Institute for Development Policy and Management, University of Manchester, UK, Paper no. 11

Heeks R (2003) Most e-government-for-development projects fail: how can risks be reduced? iGovernment working paper series. Institute for Development Policy and Management, University of Manchester, UK

Hollifield CA, Donnermeyer JF (2003) Creating demand: influencing information technology diffusion in rural communities. Gov Inf Q 20(2):135–150

Kaaya J (2004) Implementing e-government services in east Africa: assessing status through content analysis of government websites. Electron J e-Gov 2(1):39–54

Kristiansen S (2004) Information asymmetry, corruption and governance. The Norwegian network on ICT and development annual workshop: e-governance, e-democracy and development, Kristiansand, Norway, 11–12 Oct 2004

Kumar R, Best ML (2006) Impact and sustainability of e-government services in developing countries: lessons learned from Tamil Nadu, India. Inf Soc 22(1):1–12

Mahegere JEW (2010) ICTs and e-governance for sustainable rural development in Tanzania: the case of rural multi-purpose community telecentres. Symposium on ICTs and development: an international workshop for theory, practice and policy, IIT Delhi, 11–12 Mar 2010

Mascarenhas O, Kimasha E (2007) A baseline study for assessing the impact of fiber optic broadband in Bunda and Serengeti Districts, March 2007. http://www.ict4rd.ne.tz/files/docs/ICT4RD_Baseline_MainDoc.pdf. Accessed Oct 2009

Masiero S (2011) Financial vs. social sustainability of telecentres: mutual exclusion or mutual reinforcement? Electron J Inf Syst Dev Ctries 45(3):1–23

Oestmann S, Dymond AC (2001) Telecentres – experiences, lessons and trends. In: Latchem CR, Walker D (eds) Telecentres: case studies and key issues: management, operations, applications, evaluation. The Commonwealth of Learning, Vancouver

Prado P (2010) Lighting up the dark: telecenter adoption in a Caribbean agricultural community. J Community Inform 6(3):1–14

Schuppan T (2009) E-government in developing countries: experiences from sub-Saharan Africa. Gov Inf Q 26:118–127

Seifert JW, Bonham GM (2003) The transformative potential of e-government in transitional democracies. The international conference on public administration in the 21st century: concepts, methods, technologies, School of Public Administration, Lomonosov Moscow State University, 26–29 May 2003

Sein MK (2009) E-government in developing countries: linking G to C through I. In: Proceedings of the 1st international conference on e-government & e-governance, Ankara, Turkey, 12–13 March

Sein MK, Furuholt B (2009) Intermediaries in ICT4D: the other "I". In: Proceedings of the association for information systems 2nd annual SIG GlobDev workshop (Pre-ICIS meeting), Phoenix, Arizona, 14 December

Sein MK, Ahmad I, Harindranath G (2008) Sustaining ICT for development projects: the case of Grameenphone CIC. Telektronikk 104(2):16–24

Slack F, Rowley JE (2004) Challenges in the delivery of e-government through kiosks. J Inf Sci 30 (4):369–377

Tan L (2007) Rural communication access centres in Tanzania. Thematic report. The International Institute for Communication and Development (IICD). http://www.iicd.org/files/Rural-Communication-Access-Centres-Tanzania-2007-Liang-Tan.pdf. Accessed Sept 2009

UNDP (2010) E-governance and access to information for citizens' participation. Practice notes. http://www.undp.org/governance/sl-egov.htm. Accessed Jan 2010

UNESCO (2010) Web world. Communication and information activities. http://portal.unesco.org/ci/en/ev.php-URL_ID=3038&URL_DO=DO_TOPIC&URL_SECTION=201.html. Accessed Jan 2010

United Nations (2010) United Nations e-government survey 2010: leveraging e-government at a time of financial and economic crisis. United Nations, New York, 2010. http://unpan1.un.org/intradoc/groups/public/documents/un/unpan038851.pdf. Accessed Mar 2011

Wahid F, Furuholt B, Kristiansen S (2006) Internet for development? Patterns of use among Internet café customers in Indonesia. Inf Dev 22(4):278–291

Chapter 7
Reflections on a Multimethodology Approach to Business Process Automation

Helena McCabe, David Bustard, and Patricia O'Sullivan

1 Introduction

In effect, all information systems are implemented to support business processes within organizations. A *business process management system* (BPMS) (Van der Aalst et al. 2003; Weske 2007) makes that relationship explicit by providing direct technology support for process enactment, creating what are known as *process-aware information systems* (PAIS) (Dumas et al. 2005; Russell 2007). The broad purpose of this chapter is to make progress towards the development of an integrated step-by-step methodology for such process automation.

The potential benefits of process automation through a BPMS have been appreciated for many years (Van der Aalst et al. 2003). So far, however, no general methodology has been offered to guide their introduction and use. Dumas et al. noted this deficiency in 2005 (Dumas et al. 2005) and although various ways of enhancing PAIS development have been examined since then (Weber et al. 2009), a proposal for all-encompassing methodology has still to emerge. Instead, organizations typically piece their own methodology together, taking what can be described as a *multimethodology* approach. This is defined by Mingers and Gill (1997) as "combining together more than one methodology (in whole or part) within a particular intervention," which means using a range of available paradigms, methodologies, techniques and tools, as considered appropriate in a specific problem situation (Mingers and Brockelsby 1997).

H. McCabe (✉)
KTL, Maynooth, Ireland
e-mail: Lena.McCabe@ktl.ie

D. Bustard
University of Ulster, Coleraine, Ireland

P. O'Sullivan
Trinity College, Dublin, Ireland

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_7,

This chapter considers the research question: *do we need an end-to-end methodology for PAIS development or is multimethodology use sufficient*? The first section explains the BPMS approach to information system development in more detail, presenting an 8-stage framework for process automation. This is followed by a case study of multimethodology use at KTL, a telecommunications company in Ireland. The general lessons from this experience are then presented, before looking specifically at the research question. Opportunities for further research are also identified.

2 Business Process Management

The field of BPM covers "... the methods, techniques and tools to support the design, enactment, management and analysis of operational business processes" (Van der Aalst et al. 2003). A BPMS is used to automate selected processes, which have the effect of making them explicit, thereby facilitating their enactment and ensuring conformance to their specification. Van der Aalst et al. (2003) define a BPMS succinctly as "a generic software system that is driven by explicit process designs to enact and manage operational business processes." This survey also describes the rise of BPM in terms of the evolution of information systems from a *data* to a *process* perspective in the 1990s, through to the current need to support organic growth in organizations. In practice, this means seeing BPM as an ongoing cycle of process design and refinement to achieve maximum organizational benefit. Indeed, Hammer and Champy (1993) argue that organizations should become "process enterprises," placing processes at the core of their business operations.

Janelle Hill, research vice president at Gartner, identifies the next stage of evolution as "pushing BPM beyond its traditional focus on routine, predictable, sequential processes towards broader, cross-boundary processes that include more unstructured work" (Pettey 2010). This trend moves development further away from traditional vertical function-led information systems towards the horizontal process-aware systems that are the focus of this chapter.

The activities involved in business process automation can be summarized as an 8-stage framework (Fig. 7.1), as follows:

1. *Understand the Business Context*: appreciate the aims and activities of the organization, its current concerns, and constraints imposed by its environment.
2. *Identify and Document Business Processes*: identify all significant business processes in the organization, describing them in reasonable detail, and determining opportunities for improvement. Relevant improvement techniques include soft systems methodology (Checkland and Scholes 1999; Wilson 2001) and Lean Thinking (Womack and Jones 2003).
3. *Select BPMS*: based on an appreciation of organizational and process requirements from the first two stages and an awareness of the commercial products on offer, select a suitable business process management system to

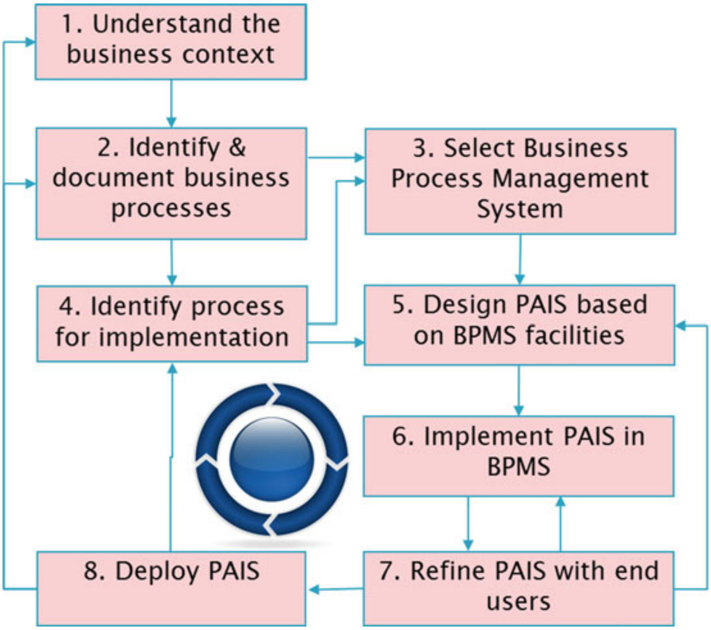


Fig. 7.1 Business process automation framework

implement specific (process-aware) information systems. Current leading BPMS vendors include BizAgi, ICCM Solutions, PNMSoft, Questetra, and Whitestein (Cool Vendors in Business Process Management 2010).
4. *Identify a Process for Implementation*: from an understanding of the business processes present or needed, select a process for implementation.
5. *Design PAIS Based on BPMS Facilities*: design the PAIS based on a detailed understanding of the infrastructure and features provided by the BPMS, including allowance for any limitations that may be present.
6. *Implement PAIS in BPMS*: implement the PAIS through the BPMS, testing it to ensure technical correctness.
7. *Refine PAIS with End Users*: review the PAIS implementation with those who will use it, adjusting the design as necessary (looping back to Stage 5 or 6).
8. *Deploy PAIS*: put the PAIS into operation, providing whatever training is needed by its end users, and further refine it as necessary.

Stages 4–8 in this framework correspond to the creation of individual information systems built around selected processes. This requires the use of standard production phases for information system development, covering requirements definition, design, implementation, integration and test, installation, and final acceptance (Avison and Fitzgerald 2003). Also, such projects would typically be managed within a specific management methodology, such as PRINCE2, as discussed by Roseman (2010). Further information systems can be created in the same way (from stage 4), with the business context, business processes, and choice of BPMS (stages 1–3) also reviewed periodically.

In principle, more than one BPMS could be used within an organization, selected according to the type of process involved. In practice, however, that is currently impractical in most circumstances, for reasons of cost, and the breadth of expertise needed. So, organizations typically adopt a single BPMS, assuming it to be sufficiently flexible to cover the full range of process activities that may have to be supported. According to Hill et al. at Gartner (2007), a general BPMS must, minimally, support the following ten areas of functionality:

1. *Process execution and state management engine*: a BPM engine which has the ability to execute end-to-end business processes
2. *Model-driven design/development environment*: a (drag-and-drop) modeling environment suitable for all aspects of process design
3. *Document and content management*: the ability to manage all types of documents and records, both inside and outside the context of a process flow
4. *User and group collaboration*: the provision of design time tools to enable collaboration among development teams and runtime tools to enable collaboration in work activities and processes
5. *Process component registry/repository*: the provision of a back-end administration repository to store and manage process components
6. *System management and administration*: the tools set up to maintain system and human access
7. *Business rule management*: the ability to trigger system actions and responses based on business logic
8. *In-line and offline simulation and optimization*: the ability to test and improve processes before they go live
9. *Business event management, business activity monitoring* (*BAM*)*, and business intelligence* (*BI*) *management*: employing reporting tools for governing and monitoring business operation behaviors
10. *System connectivity*: tools that enable system architects to set up services, enabling bidirectional connections to a variety of back-end business applications

At a higher level, Melenovsky and Sinur identify six critical success factors for business process management (Melenovsky and Sinur 2006): (1) strategic alignment (the linkage of processes to strategic organizational goals), (2) culture and leadership (the collective beliefs that mold process-related activities), (3) people (individuals who enhance and apply their process-related knowledge), (4) governance (transparency and accountability with regard to processes), (5) methods (approaches and techniques used to govern process outcomes), and (6) information technology (the technology utilized to support the management of business processes).

This section has presented a general framework for business process automation. The next section illustrates the framework through its application in practice. The particular research focus is on the use of different methodologies in this work.

3 Case Study

Killarney Telecommunications Ltd. (KTL) is one of Ireland's leading infrastructure solution providers in Mobile Telecoms and Power Networks. Their core business is to build masts, rigging, and base stations for mobile phone and broadband companies. Much of this work involves projects with similar well-defined stages. For example, building each telecommunications mast follows the same series of steps, covering the following: *design*, *construction*, *installation*, *commissioning*, *quality assurance*, and *client hand over*.

In 2009, KTL had the vision of using business process automation to drive all project-related processes in the company, including integration with existing Human Resource Management and Financial Management systems. This work was undertaken in collaboration with the University of Ulster, funded by the InterTradeIreland FUSION scheme (McCabe et al. 2011). The first named author was originally employed as an Information Systems Analyst on that project.

The main organizational objectives set by the KTL directors were general concerns about process automation:

1. Business process automation must support the business strategy and add business value.
2. The selected BPMS must be able to drive processes without the need to adapt them unduly to suit the technology.
3. Introduction of the BPMS and implementation of each PAIS should involve full consultation with operational staff.
4. The PAIS for projects should handle every stage of a project-related process, including the assignment of tasks to individuals and making available all necessary documentation.
5. The PAIS had to be flexible to handle the variations that occur in practice and allow for staff being based locally in the office or remotely on site.

Many other requirements were also general:

- Review individual processes to remove unnecessary activity, merge similar processes, and generally improve processes where possible.
- Consolidate information stored in dispersed locations into a single repository.
- Implement document management in a way that prevents multiple versions of documents circulating simultaneously.
- Improve project-related communication and collaboration within the company.
- Provide management and other decision makers with detailed real-time information on progress with projects.

Given the simplicity and generality of the business process automation framework outlined in Sect. 2, it seemed reasonable to assume that there would be a single overall guide to the activity involved somewhere in the literature. In practice, however, although there was general advice on the introduction and use of a BPMS, no suitable description was found at the level of detail needed to tackle the overall task systematically. Thus, instead, the project team followed common practice and used a multimethodology approach, pooling their expertise to tackle each stage of the work as effectively as possible. Having a single methodology implies integrated guidance, models and tools across every stage of the framework; a multimethodology approach means that the linkage between one or more phases is unspecified.

The activities associated with each stage of the process automation framework (Fig. 7.1), and the methodologies used in pursuing them were as follows:

1. *Understand the Business Context*: Through a workshop, all aspects of KTL's business were examined to build an understanding of its long term aspirations, current issues, basic structure, broad need for BPM, and current use of information systems. Soft systems methodology was used to facilitate the workshop, which led to the creation of a high-level rich picture and conceptual model (McCabe et al. 2011).
2. *Identify and Document Business Processes*: Workshops also drove the process discovery phase. Techniques used included marker boards and sticky notes. After identifying KTL's business functions, departments, and main activities, workshop participants were asked to map out the main business processes, which were subsequently modeled in MS Visio, using BPMN (business process model and notation) (OMG). Interviews were then held with individual business unit managers to assess the business requirements and expectations for PAIS development, documented as an SSADM Requirements Catalogue (Duncan et al. 1995). The analysis identified a particular need for automated document templates throughout the process path, accumulating both system-generated and user-supplied document content along the way—a feature that is not always present in BPMS platforms.
3. *Select BPMS*: The selection of the BPMS was approached very carefully (McCabe et al. 2011). Sage CRM was selected, strongly influenced by a decision to upgrade KTL's financial system with Sage software. This required a SQL Server infrastructure, on which the two systems were subsequently installed. This architecture would later facilitate bidirectional communication between the PAIS and the financial management system.
4. *Identify PAIS for Implementation*: KTL management selected the building of telecommunications masts as the first process to implement with the BPMS. This was expected to provide most business value, but the choice conflicted with general BPM advice to start with a "low-complexity" process. The mast-building process touched every department in the company, requiring input from project coordinators, account managers, project managers, field engineers, health and safety officers, and quality assurance personnel. To reduce complexity, it was agreed that only the core structure would be supported in Release 1.0, with subprocesses, such as "invoicing," following later. Requirements for the PAIS were also specified, again in the form of an SSADM Requirements Catalogue.

5. *Design PAIS Based on BPMS Facilities*: Meetings were held with individual business representatives to identify and document lower-level process detail. This revealed some small process variations between different client-project teams. Also, conflicts emerged. For example, the accounts department wanted tight control over spending capability, but project managers wanted more flexibility. Such issues had to be resolved as it was an essential requirement to use the same process for all projects. Functional specifications were then prepared as use cases (Stevens and Pooley 1999), chosen to support the creation of functional requirement documents. Unfortunately, these proved inaccessible to many of the end users who were expected to approve them. Instead, prototyping was used to explain what was planned. This meant that the PAIS design evolved incrementally after every meeting. One important consequence was a difficulty in keeping the use case documentation aligned with changes. This had a knock-on problem as, without supporting documentation, it was difficult to determine if an underlying workflow path was dependent on the change proposed. On reflection, at minimum, a Data Catalogue (indicating dependencies and specific workflows) and a requirements traceability matrix (ensuring change requests were aligned with original scope) should have been put in place to control the impact and scope of the requested changes.
6. *Implement PAIS in BPMS*: Implementing process support using the BPMS appeared relatively straightforward initially, aided by graphical modeling tools, online tutorials, and code samples. However, it emerged that customized BPMS concepts introduced for KTL, such as "projects" and "jobs," were treated differently to built-in concepts. For example, standard documents could be merged but not those developed for KTL. Such issues required regular consultation with the supplier. Some issues could be resolved directly, but others required separate development through a consultancy firm. This effectively extended the development team, thereby creating additional difficulties, particularly in managing distributed changes and in handling the security risks introduced by opening access to KTL's servers. Overall, these issues led to significant delays, pushing the project delivery timeline out by several months.
7. *Refine PAIS*: A user acceptance testing team was formed from selected staff members, representing each business function. With minimal guidance, the team followed steps equivalent to their usual manual process. Potential issues, bugs, observations, and suggestions for improvement were noted. Some staff strongly resisted considering an alternative to their current way of working, making it necessary to emphasize the benefits of the new approach and the management decision to implement it across the company. It was also found, however, that not all projects fitted the standard template that had been created, necessitating further development. Additionally, it emerged that finer access control to documents and activities was needed, which likewise required significant additional development.
8. *Deploy PAIS*: A final test team was created, involving input from the BPMS vendor, the consultancy firm that implemented additional functionality and KTL's external IT support providers, who maintained the servers involved.

Testing was performed on site over a 3-day period, covering ten areas, ranging from standard functionality and bespoke development down to hardware and system infrastructure. The main purpose of the test was to confirm that the system was ready for deployment so, as expected, very few issues emerged. User training was then conducted in-house, with each trainee simulating their daily activities within the new system. This approach proved very successful, judging by the positive feedback received and the relatively small number of queries raised subsequently. The PAIS went live in December 2010.

This section has summarized an experience of introducing and using a BPMS. It identified a range of methodologies, used as necessary, and selected according to the preferences of the development team and the needs of the end users. The next section reflects on some of the lessons learned from this experience, particularly in relation to the use of a mixed range of methodologies and the prospect of progress towards a simpler unified approach.

4 Lessons Learned

The general goal of the work described here was to find the best route between an initial vision of business process automation to having a BPMS in place that supported key processes in a way that realized significant business benefit. The study at KTL illustrated only one development route but is still sufficient to allow a number of general conclusions to be drawn:

1. *Having a generic framework for business process automation is helpful*. The 8-stage framework presented in this chapter seems general enough to argue that all of its stages need to be present and followed in the order shown. Having this model at the beginning of the KTL project would have been helpful, especially if populated with guidance on common issues that can occur at each stage. For example, in the KTL project the following advice would have been welcome: (1) all BPMS platforms have limitations so it is important to select a platform assuming further interaction and support with the supplier and/or associated BPMS consultants; (2) aim immediately for a fine-grained approach to data/document access control in process design (from server access down to individual permissions); and (3) appreciate that business change is always difficult even when supported strongly by higher level management, so make allowances for end-user resistance at every stage.
2. *The BPMS must be selected carefully*. In a BPM automation project, there is pressure to select the BPMS platform quickly as management is concerned about the costs involved and any training that is needed. However, the decision cannot be rushed, as there are many factors that need to be weighed up carefully. These include (1) the cultural fit with the BPMS vendor, as an ongoing relationship needs to be developed; (2) the technological fit with vendor as there may be potential for other projects; (3) the BPMS features and facilities available, as

these can vary significantly from one product to another; and (4) overall value for money.

3. *Start with the* “big picture”. Before considering which processes to automate, it is beneficial to appreciate their wider context, and SSM was very helpful in facilitating that analysis (McCabe et al. 2011). The construction of a rich picture at KTL, for example, helped bring out wider system integration issues that had a direct impact on the project, including the identification of the imminent need for replacement of the finance system.
4. *Start small*. The process selected for initial implementation was too ambitious. It would have been more productive overall to identify something much smaller to build knowledge of the development process and gain user confidence.
5. *Addressing end-user needs is essential*. This was partially successful in the KTL project in that management and office staff were actively involved in the early-stage workshops. Unfortunately site staff (approximately 80 % of the company) were largely overlooked at that stage, and it emerged later, during acceptance testing, that they felt the system was designed to suit head office preferences. Also, during the design phase, staff had difficulty interpreting BPMN diagrams and use case definitions. Techniques need to be used that achieve the necessary input from end users in their terms but also support the more precise descriptions needed to specify and construct each PAIS around the selected BPMS. Allowance for significant design iteration is also important as many end users can only comment effectively on a working system rather than abstract models. Users also needed to better understand that some constraint over previous behavior was inevitable.
6. *A single comprehensive methodology for business process automation is not essential but desirable*. A valuable PAIS, built on a cost-effective BPMS, has been implemented successfully at KTL, without guidance from a single methodology. On reflection, however, that can largely be attributed to the combined experience of the development team and the care that was taken at each stage of the work. This meant, for example, that while it was beneficial to use SSM in the first stage of the 8-stage framework, as a way of developing a shared understanding of the business context, it was not necessary to spell out exactly how SSM would be applied and how its models would be used in subsequent phases. Nevertheless, if a full default methodology had been available, it would have raised awareness of the linkage between stages and identified possible tool support to facilitate such connections, thereby further guiding the project.

   In both software engineering and information systems development, many have argued that it is impractical to tackle all projects in the same way (Fitzgerald et al. 2003). The alternate approach, often described as “method engineering” (Bergstra et al. 1985), means tailoring each project according to the range of relevant factors involved, including the novelty of the work, the application area, and the specific experience of the developers. This implies supporting options within the framework. These might be within a single stage but more commonly, would extend across several stages. For example, in the KTL study, SSM was used in the first two stages and the SSADM Requirements Catalogue technique

(Duncan et al. 1995) in stages 2 and 4. The availability of such options effectively turns the methodology into a "toolbox," which ideally would offer as wide a choice as possible. Selected techniques may have to be adapted, however, to ensure a reasonable fit with the wider methodology in which they are used.

7. *BPMN is a step towards standardization*. The Object Management Group (OMG) defines BPMN as a process modeling notation which "creates a standardized bridge for the gap between the business process design and process implementation" (OMG). If all BPMS platforms end up using this notation, there would be an opportunity to document processes directly within the BPMS.

5 Conclusion

Since all information systems support business processes, it is possible to envisage a future in which business process automation is part of every information system created. One implication of this scenario is that guidance on process automation would then be needed in whatever information systems methodology is used. This chapter has attempted to made progress towards the development of such guidance by proposing a general 8-stage framework to encompass the range of activities involved. This covered the identification of business processes, the selection of a BPMS, and the implementation and refinement of process-aware information systems using that BPMS. It was suggested that it was desirable to integrate the eight stages, where possible, to facilitate progress from one stage to another. Such linkage would also help handle the ripple effect of change across the models produced, as business processes and PAIS systems evolve.

Before integration can be considered across the business process automation framework, however, it is first necessary to have a good understanding of the requirements and issues in implementing each stage. This can be obtained through a multimethodology approach to process automation, which involves a selection of "paradigms, methodologies, techniques, and tools" considered appropriate in each stage (Mingers and Brockelsby 1997). This chapter has described the use of this approach in a study of business process automation in a telecommunications company in Ireland.

Perhaps, the main overall conclusion from the study is that the field is still maturing, as reflected in the variation in facilities across the available BPMS products. Such variation makes it difficult to define a general integrated methodology at this stage of the evolution of the field. The use of the BPMN notation is, however, a clear step towards standardization and gives some opportunity to think about other linkages that might be developed. One possibility, for example, would a tool-supported link between SSM conceptual models (Wilson 2001) and BPMN descriptions, if SSM were used in the first two stages of the framework.

In practical terms, PAIS development is likely to require a multimethodology approach until some BPMS product begins to dominate the market, or there is consensus on the core set of facilities provided by each product. Until then, there are significant opportunities for research into all aspects of the framework, from the technical management of process and other descriptions to the subtle issues in interacting with those affected by process change to ensure that they are supportive of the change and contribute appropriately to what should be an improved way of working for everyone involved.

Acknowledgements This work was completed through an IntertradeIreland FUSION project between KTL Ltd. and the University of Ulster. Thanks are extended to colleagues who were involved in the project: Kevin Curran at Ulster and Niall Byrne at KTL.

References

Avison D, Fitzgerald G (2003) Information systems development: methodologies, techniques and tools, 3rd edn. McGraw Hill, Maidenhead

Bergstra J, Jonkers H, Obbink J (1985) A software development model for method engineering. In: Roukens J, Renuart J (eds) ESPRIT: status report of ongoing work. Elsevier, North-Holland, pp 85–94

Checkland P, Scholes J (1999) Soft systems methodology in action. Wiley, Chichester

Cool Vendors in Business Process Management (2010) [internet]. http://www.gartner.com/technology/research/offer/cool-vendors.jsp. Accessed 9 Aug 2011

Dumas M, van der Aalst W, Ter Hofstede AHM (eds) (2005) Process-aware information systems: bridging people and software through process technology. Wiley, Hoboken

Duncan J, Rackley L, Walker A (1995) SSADM in practice: version 4 text. Macmillan, Basingstoke

Fitzgerald B, Russo NL, O'Kane T (2003) Software development method tailoring at Motorola. ACM 46(4):65–70

Hammer M, Champy J (1993) Re-engineering the corporation. Nicholson Brealey Publishing, London

Hill JB, Cantara M, Deitert E, Kerremans M (2007) Magic quadrant for business process management suites [internet]. www.parstasmim.com/Resources/Gartner%20Magic%20Quadrant%20for%20BPM%202007.pdf

McCabe H, Bustard DW, Curran K, Byrne N (2011) Managing step changes in information system support: lessons from an industrial study. In: Proceedings of ISD 2010, Prague. Springer, Berlin

Melenovsky MJ, Sinur J (2006) Maturity model identifies six phases for successful BPM adoption [internet]. http://zonecours.hec.ca/documents/H2010-1-2321794.BPMMaturityModelIdentifiesSixPhasesforSuccessfulBPMAdoption.pdf

Mingers J, Brockelsby J (1997) Multimethodology: towards a framework for mixing methodologies. Omega 25:489–509

Mingers J, Gill A (eds) (1997) Multimethodology: theory and practice of combining management science methodologies. Wiley, Chichester

OMG. Business process model and notation (BPMN) version 2.0 [internet]. http://www.omg.org/spec/BPMN/2.0/PDF

Pettey C (2010) Gartner reveals five business process management predictions for 2010 and beyond [internet]. http://www.gartner.com/it/page.jsp?id=1278415. Accessed 2 Apr 2011

Rosemann M (2010) The service portfolio of a BPM center of excellence. In: Bernus P, Błażewics J, Schmidt G, Shaw M (eds) Handbook on business process management 2: strategic alignment, governance, people and culture, International handbooks on information systems. Springer, Berlin, pp 267–284

Russell NC (2007) Foundations of process-aware information systems. Ph.D. thesis, Faculty of Information Technology, Queensland University of Technology, Brisbane, Australia
Stevens P, Pooley R (1999) Using UML: software engineering with objects and components. Addison Wesley, Harlow
Van der Aalst WMP, ter Hofstede AHM, Weske M (2003) Business process management: a survey. In: van der Aalst WMP, ter Hofstede AHM, Weske M (eds) Proceedings of business process management 2003, Eindhoven, The Netherlands. Lecture notes in computer science, vol 2678. Springer, Berlin, pp 1–12
Weber B, Reichert M, Wild W, Rinderle-Ma S (2009) Providing integrated life cycle support in process-aware information systems. Int J Cooper Inf Syst (IJCIS) 18(1):115–165
Weske M (2007) Business process management: concepts, languages, architectures. Springer, Berlin
Wilson B (2001) Soft systems methodology: conceptual model building and its contribution. Wiley, Chichester
Womack JP, Jones DT (2003) Lean thinking: banish waste and create wealth in your corporation. Simon & Schuster, New York

Chapter 8
NDT-Driver: A Java Tool to Support QVT Transformations for NDT

J.A. García-García, C.R. Cutilla, M.J. Escalona, M. Alba, and J. Torres

1 Introduction

The model-driven engineering paradigm (MDE) came up in order to tackle the complexity of platforms and the inability of third-generation languages to relieve this complexity and effectively express the domain concepts of the problem. This new paradigm, apart from raising the level of abstraction, intends to increase automation during the life cycle of software development.

MDE works, as the primary form of expression, with definitions of models and transformation rules among these models entailing the production of other models. One of the languages to describe the transformation rules is QVT (OMG 2008) or query/view/transformation language. QVT standard defines a declarative and imperative language proposed by the OMG (Object Management Group) for model transformation in the context of MDE.

However, QVT notations are not easy to be applied in practical environments because it does not result too friendly for development teams. Concepts such as models, metamodels, transformations, or QVT are not common notations in the enterprise environment, and they seem too abstract and complex. For this reason, this research paper presents how the NDT methodology, acronym for Navigational Development Techniques, addresses this challenge with the aim of involving the enterprise with the power of the model-driven paradigm.

The chapter is structured as follows. After this introduction, Sect. 2 briefly studies how to transform some methodologies that belong to model-driven engineering paradigm. Section 3 provides an overview of NDT methodology and their tools. Section 4 presents NDT-Driver, a tool that implements a set of automated

J.A. García-García (✉) • C.R. Cutilla • M.J. Escalona • M. Alba • J. Torres
IWT2 Group, University of Seville, Seville, Spain
e-mail: julian.garcia@iwt2.org; carmen.ruiz@iwt2.org; mjescalona@us.es; manuel.alba@iwt2.org; jtorres@us.es

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_8,

procedures to generate different models in NDT life cycle. Finally, in Sect. 5, final conclusions and lessons learned are detailed, and the chapter concludes with future work in Sect. 6.

2 Related Works

Currently, there is a wide range of Web methodologies that belong to the MDE paradigm and define its procedure and transformations from one model to other model. For this reason, only the most referenced ones will be briefly described:

- OOHDM (Rossi and Schwabe 2008) is a highly referenced methodology and one of the most accepted. It consists of four different design activities: conceptual model, navigational design, abstract interface design, and implementation. The conceptual and navigational transformations are completely independent, except from operation invocations of conceptual PSM objects from the navigational PSM, where the type of invocation may vary. Thus, the implementation technology and platform of the conceptual PSM and the navigational PSM may be selected and combine quite independently.
- UWE (Koch et al. 2008) is a methodology based on Unified Modeling Language (UML 2005) for Web application development. It covers the whole development life cycle of Web systems from the requirements specifications to code generation, focusing on personalized or adaptive applications. The process of development is based on three main phases: the phase of capture of requirements, the phase of analysis and design, and the phase of implementation.
  The transformation rules are defined to map metamodel WebRE to UWE and UWE to metamodels. The first model transformation step of the UWE process consists of mapping Web requirements models to UWE functional models. The design models are content, navigation, process, presentation, and adaptation. There is a set of dependencies among these functional models themselves that allow the creation of other models, as well as their amendments.
  ArgoUWE and MagicUWE have been developed for the computer-aided design of Web applications using UWE.
- WebML (Ceri et al. 2000) is a proposed methodology based on formal specification graphics into a complete design process. It is used on applications that highly interact with information.
  The model is divided into four phases: the process definition phase, which uses the notation BPMN; the data model phase, which is defined by different data tables and their relationships and uses graphs of entity-relation or UML class diagrams; the hypertext model phase, which consists in the composition model, representing a hypertext sites where every site has content elements, and the navigation model, which defines the links among different sites; and finally, the composition model, which details content items of each site.

WebML is associated with a development tool called WebRatio, which automatically generates fully functional applications from WebML diagrams.

- OOWS (Fons et al. 2003) is an extension of OO-Method. The process to define a software system for a Web environment is based on two steps: the conceptual model and the navigational model. The conceptual model is the specification of user requirements. The navigation model is based on an object model and navigation requirements using UML notation. The navigation model is composed of a set of navigation maps (one for each agent) to represent and structure the global vision system by defining allowable navigation. This is directly represented by means of a directed graph where nodes are navigational contexts and arches represent the navigation links.

  Once the navigational model is defined, the presentation characteristics are associated with the system. Presentation requirements will be based on the use of simple presentation patterns related to the different elements that conform a navigation node. The generator (compiler) will use this information stored to create the diverse interfaces for each user within the architecture of Web application that the OOWS method proposes.

3 An Overview of NDT and NDT-Suite

The proposed methodology NDT (Escalona and Aragón 2008), acronym for Navigational Development Techniques, belongs to the MDE paradigm.

Initially, NDT dealt with the definition of a set of formal metamodels for the requirements and analysis phases. In addition, NDT defined a set of derivation rules, stated with the standard QVT, which generated the analysis models from requirements model.

Nowadays, NDT defines a set of metamodels for every phase of the life cycle of software development: the feasibility study phase, the requirements phase, the analysis phase, the design phase, the implementation phase, the testing phase, and finally, the maintenance phase. Besides, it states new transformation rules to systematically generate models. Figure 8.1 shows the first part of the NDT life cycle[1].

The main goal of the requirements phase is to build the catalogue of requirements containing the needs of the system to be developed. It is divided into a series of activities: capture, definition, and validation of requirements.

NDT classifies project requirements according to their nature: information storage requirements, functional requirements, actor requirements, interaction requirements, and nonfunctional requirements. In order to define them, NDT provides special patterns and UML techniques, such as the use case technique for functional requirements specification.

[1] More information about the NDT full life cycle can be found in www.iwt2.org

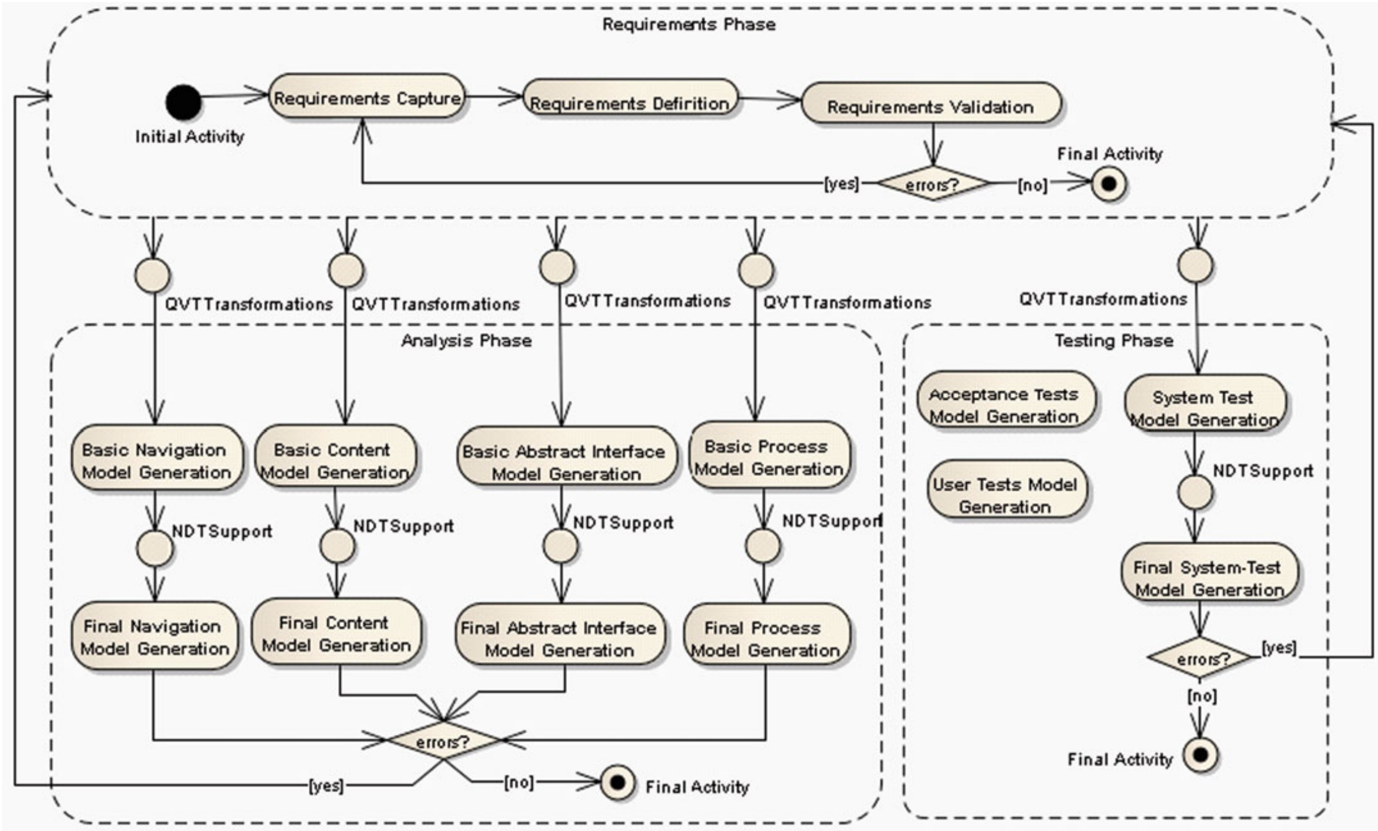


Fig. 8.1 Transformations from requirements to analysis and from requirements to testing

Once the requirements specification phase has been completed and the catalogue of system requirements has been drafted and validated, NDT defines derivation rules to generate the system test model and the analysis phase models. Figure 8.1 shows all these transformations through the stereotype "QVTTransformation."

NDT conceives the testing phase as an early phase of the software life cycle and proposes to carry it out together with the remaining phases. NDT proposes three models in this phase: the implementation tests model, the system tests model, and the acceptance tests model. The system tests model is the only one that can be generated systematically. NDT proposes derivation rules to generate the basic system tests model from the functional requirements defined in the requirements phase. The team of analysts can perform transformations in order to enrich and complete this basic model. Transformations are represented in Fig. 8.1 through the stereotype "NDTSupport."

The analysis phase will include the resulting products from the analysis, definition, and organization of requirements in the previous phase. At this phase, NDT proposes four models: the conceptual model, which represents the static structure of the system; the process model, which represents the functional structure of the system; the navigation model, which shows how users can navigate through the system; and the abstract interface model, a set of prototypes of the system.

The transition between the requirements and the analysis model is standardized and automated, and it is based on QVT transformations, which translate the concepts of requirements metamodels to the first versions of the analysis models. These models are known in NDT as basic models of analysis. For example, the basic conceptual model of analysis is obtained from the storage requirements defined during the requirements phase.

Thereafter, the team of analysts can transform these basic models to enrich and complete the final model of analysis. As this process is not automatic, the expertise of an analyst is required. Transformations are represented in Fig. 8.1 through the stereotype "NDTSupport." To ensure consistency between requirements and analysis models, NDT controls these transformations by means of a set of defined rules and heuristics.

To sum up, NDT offers an environment conducive to the development of Web systems, completely covering the life cycle of software development. NDT has been applied in many practical environments and has succeeded due to the application of transformations among models, which has reduced development time (Escalona and Aragón 2008).

3.1 NDT-Suite

The application of MDE and, particularly, the application of transformations among models may become monotonous and very expensive if there are no software tools automating the process. To meet this need, NDT has defined a set of supporting tools called NDT-Suite. Currently, the suite of NDT comprises the following free Java tools:

- NDT-Profile is a specific profile for NDT developed using Enterprise Architect (EA 2010). NDT-Profile offers the chance of gathering all the artifacts that define NDT easily and quickly, as they are integrated within the tool Enterprise Architect.
- NDT-Quality is a tool that automates most of the methodological review of a project developed with NDT-Profile. It checks both the quality of NDT methodology in each phase of software life cycle and the quality of traceability of the MDE rules of NDT.
- NDT-Driver is presented in Sect. 4 of this chapter.
- NDT-Prototype is a tool designed to automatically generate a set of XHTML prototypes from the navigation models of a project, described in the analysis phase, developed with NDT-Profile.
- NDT-Glossary implements an automated procedure that generates the first instance of the glossary of terms of a project developed by means of NDT-Profile tool.
- NDT-Checker is the only tool in NDT-Suite that it is not based on the MDE paradigm. This tool includes a set of sheets different for each product of NDT. These sheets give a set of checklists that should be manually reviewed with users in requirements reviews.

4 NDT-Driver

4.1 Overall View

NDT-Driver is one of the main tools of NDT methodology. It is completely based on NDT-Profile.

NDT-Driver is developed in Java and implements a set of automatic procedures for carrying out each of the QVT transformations defined in NDT. It generates the analysis models from requirements, the design models from the analysis, and the tests models from requirements. In addition, NDT-Driver allows obtaining the model requirements from the requirements gathered in the feasibility study phase of the project.

Furthermore, NDT-Driver can be used in projects using both a sequential life cycle and an evolutionary life cycle. Once transformations to perform have been selected, models to generate can be chosen.

To support projects with evolutionary development cycles, NDT-Driver allows the models previously selected to be rebuilt and updated. For example, if all storage requirements have already been defined in the requirements phase, the conceptual model of the analysis phase can be generated. Then, if it is noticed that any storage requirement is not defined according to the user's needs, it is not completely necessary to rebuild the content model because the tool will update it.

NDT-Driver also provides support for projects that develop business systems with service-oriented architecture (SOA).

4.2 QVT Transformations

QVT Transformations offered by NDT are grouped into three categories:

- Requirements to analysis:
  - *Requirements2Content*: this transformation allows the generation of the basic content model from storage requirements.
  - *Requirements2Process*: this transformation allows the generation of the basic model of process classes from the functional requirements.
  - *Requirements2Navigational*: this transformation allows the generation of the basic navigational model from interaction requirements.
  - *Requirements2Prototypes*: this transformation allows the generation of the basic abstract interface model from the requirements definition.
  - *ServicesR2ServicesA*: this transformation allows the generation of the basic service model of analysis phase from the services included in the requirements phase.
- Analysis to design:
  - *Content2DataAccess*: this transformation allows the generation of the basic model of data access classes from the content analysis model.
  - *Content2PhysicalDataModel*: this transformation allows the generation of physical data model from the content model.
  - *Process2Bussines*: this transformation allows the generation of the basic model of business classes from the model of process classes.
  - *Navigational2Presentation*: this transformation allows the generation of the basic model classes presentation from the navigation model.
  - *ServicesA2ServicesD*: this transformation allows the generation of the basic service model of design phase from the services included in the analysis phase.
- Requirements to tests:
  - *Requirements2Test*: this transformation allows the generation of the basic model of system tests from functional requirements.

Table 8.1 *Requirements2Content* transformation (QVT-Java)

QVT transformation

```
transformation Requirements2Content
             (in msr:StorageRequirementsModel, out cm:ContentModel)
{
   main (){
     msr.ObjectsOfType (StorageRequirement)- > map SRtoCL ();
     msr.ObjectsOfType (NewNature)- > map NNtoCLn ();
   }
   mapping StorageRequirement::SRtoCL () : CL {
    name:= 'CL' + self.name ();
    attributes:=self. SpecificField- > map SFtoAttr ();
    links:= self. SpecificField- > map SFtoLinks ();
   }
   mapping SpecificField sf::SFtoAttr (): Attribute
               where {sf.nature- > size () = 0;} {
    name:=self.name ();
     //...
   }
   mapping SpecificField sf, CL cl::SFtoLinks () : Association
                 where {sf.nature = conc:Concept{}}{
    name:=self.name ();
    connection = {sf.nature- > map SRtoCL (), cl};   //...
  }  //...
}
```

Java transformation

```
public void requirements2content (StorageRequirementsModel msr){
  Collection Relations col = new ArrayList < Relations > ();
  for (StorageRequirements sr : msr.getByStereotype
   ("StorageRequirement")){
    CL cl = this.createCL (sr);
    for (Attribute a : sr.getAttribute (sr){
        if (a.isBasic ()) this.createAttribute (cl, a);
          else col.add (new Relations (cl, a.getType ()));
    }
  }
  for (NewNature nn : msr.getByStereotype ("NewNature")){/* idem */}
  this.createAssociationLinks (col);//...
}
```

Presenting all these transformations in this chapter is very complex. For this reason, the first one is only presented (see Table 8.1[2]). It states how each storage information requirements has to be translated into a content class in the analysis

[2] Complete information on NDT, their metamodels and transformations can be found in www.iwt2.org

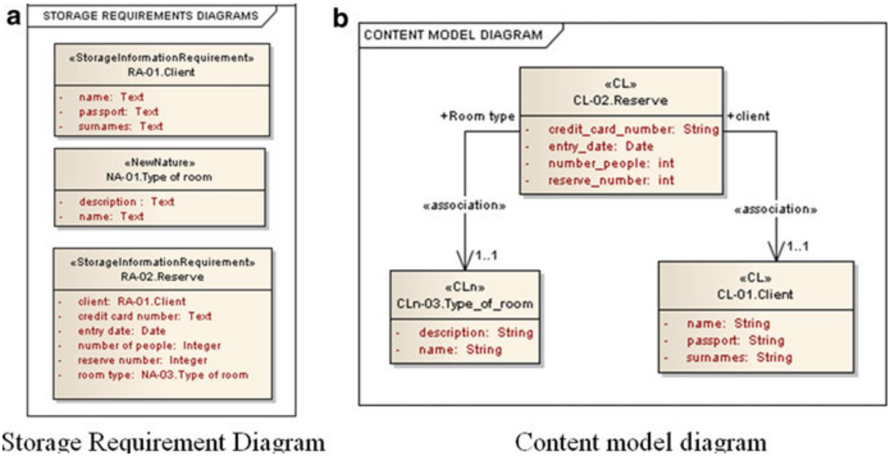


Fig. 8.2 Diagrams of the example (**a**) Storage requirement diagram and (**b**) Content model diagram

phase. Each storage information system generates a class, and each specific field generates an attribute.

In the *SRtoCL method*, each storage requirement is translated into a class (CL) in the content model. In methods such as *SFtoAttr* and *SFtoLinks*, specific fields are treated. If specific fields have a basic nature, that is, when its relation with content has cardinality 0, they are translated into attributes. If its nature is content, it is translated into association.

4.2.1 A Basic Example

This section provides a representative example of the application of the QVT transformation rule explained in the previous section. The input of this rule is the requirements model.

Figure 8.2a shows the requirements model of a small Web application that manages hotel reservations. In this basic example, important concepts are *reserve*, *client*, and *type of room*. Each concept has its own specific data: name, surname, and passport number of every hotel guest and name and description of each room type. Likewise, each reservation requires the name of the client who books the room, credit card number, date of arrival, number of people per room, and type of room chosen.

After applying the transformation rule shown in Table 8.1, three content classes of analysis will be generated, one for every storage requirement. Figure 8.2b shows a class diagram with the generated content classes, their attributes, and links.

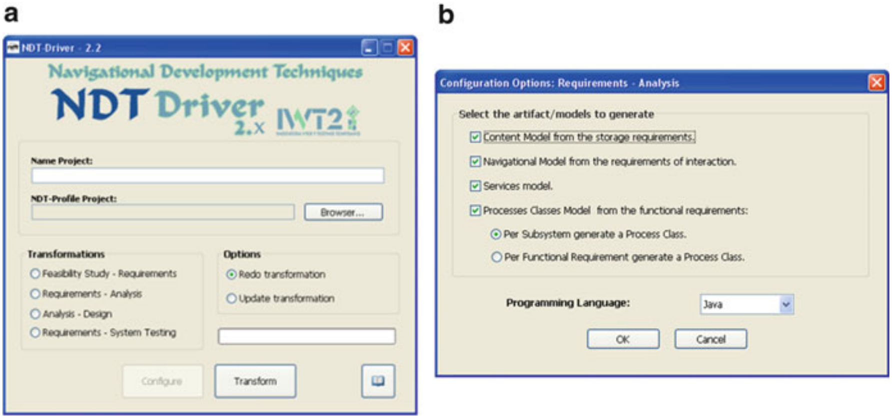


Fig. 8.3 NDT-Driver tool (**a**) Main interface (**b**) Configuration interface

4.3 Interface

The graphical interface of NDT-Driver is very simple and intuitive. Figure 8.3a shows the main interface of the tool. NDT-Driver, as the remaining NDT tools, is available in both English and Spanish. Before carrying out any transformation, some information must be given to the tool.

In the section “Name Project,” the name of the project must be entered. This will be useful for the reports generated by the tool. With the “Browser” button, the Enterprise Architect file of the project must be selected. “Transformations” button lets the user select the transformations to be carried out, and “Options” lets the user choose the way the transformation chosen will be carried out.

The “Configure” button selects the models you want to transform. By default, all the models are selected. Figure 8.3b shows the modal window associated with the transformation requirements to analysis. Finally, if clicking on the “Transform” button, NDT-Driver will begin to carry out the transformations selected. At the end of the process, a modal window displays the report of the transformations carried out.

4.4 Practical References

In the last 10 years, NDT and NDT-Suite, and particularly NDT-Driver, were used in a high number of real projects. In fact, NDT-Driver is currently being used in several projects by different companies, either public or privates, big or small. A set of projects developed with NDT was selected, and its tools were used during its life cycle.

4.4.1 AQUA-WS

EMASESA[3] is a company dealing with the general management of the urban water cycle, providing and ensuring water supply to all the citizens in Seville, Spain. AQUA-WS (AQUA-WebServices) project consists in the development and implementation of an integrated business system for customer management, interventions in water distribution and cleaning up, and projects and work management. The development time of this project is estimated in 3 years, and it will finish in 2012. AQUA-WS is very relevant for the use of NDT-Driver in the test phase.

The existing systems are the customer management system (AQUA-SiC), network management system (AQUA-ReD), and the work and projects management system (AQUA-SigO). AQUA-WS arises from the need of unifying all the existing systems into a single one, the core AQUA, by means of the same technology platform.

The project follows an iterative life cycle. Each iteration allows the development team, composed by more than 20 analysts from two companies, to define requirements, studying the previous systems, and introduce them into NDT-Profile. They are checked with NDT-Quality and NDT-Checker and, later, NDT-Driver generates functional test cases. The systematic generation of test cases from functional requirements in the project is providing good support by expediting the validation of their own functional requirements with users. In addition, NDT-Driver can reduce the time analysts spent on specifying system tests and test plans.

4.4.2 Projects for e-Health Systems

NDT was also widely applied in the e-health environment. An example of this is the Diraya project, an information system applied in the Andalusian Health Service, which allows consulting the clinical record of a patient belonging to any hospital center in Andalusia. Its requirements phase was developed by a group of six companies and a high number of analysts. Every company was expert in a specific module of Diraya.

For this project, the development team used the main tools of NDT-Suite. The use of NDT-Profile and NDT-Glossary was essential to guarantee the unification criteria in this multidisciplinary development team. NDT-Quality was also important to assure quality, and NDT-Driver was crucial to reduce the time spent on the project development.

[3] http://www.aguasdesevilla.com

5 Conclusions and Learned Lesson

In this chapter, we present NDT-Driver: a simple and intuitive tool useful for any company which develops a project using NDT methodology in order to take advantage of MDE power, even without knowing this paradigm. Moreover, we describe how the most referenced methodologies perform their transformations, and finally, we briefly introduce NDT and its supporting tools.

As a learned lesson of our experience in the use of MDE in Web treatment, we could conclude that the use of this paradigm in this environment can quantitatively improve the results of the project.

However, MDE in practical environment does not result too friendly for development teams. The concepts of models, metamodels, transformations, and among others, are not common notations for enterprise environment, and they seem too abstract and complex. However, we conclude that the use of UML profiles and UML-based tools offers an interface to deal with instances of metamodels quite suitable for analysts and designers and even for expert users.

Likewise, since transformations in QVT do not result easy to understand, our users do not work with it. They prefer an easier and more intuitive interface, as the NDT-Driver interface presented in Fig. 8.3, in order to benefit from its transformation power.

6 Future Work

NDT-Driver is a very powerful and useful tool to develop Web-oriented systems. However, we know that NDT-Driver is not functional enough. In this regard, we propose several improvements as future research.

Firstly, we suggest doing a research on how to improve test generation from functional requirements, particularly when functional requirements are described using activity diagrams. Currently, NDT-Driver generates all possible paths from the initial and final activities of the activity diagram of the functional requirement. As far as we know that the method is not suitable enough, we propose a research on how to enrich the transformation method with new techniques for the selection and reduction of redundant testing paths.

Secondly, another aspect that we must continue working on is considering other programming languages since today NDT-Driver is only available for Java.

Finally, we propose to investigate on how to incorporate heuristics in the generation of some models. For instance, in the generation of the navigation model of the analysis phase, NDT-Driver should identify aspects as that the graph is not connected. To achieve this, NDT-Driver could use the Warshall algorithm.

Acknowledgments This research has been supported by the project QSimTest (TIN2007-67843-C06_03), by the Tempros project (TIN2010-20057-C03-02), by the National Network Quality Assurance in Practise, and by the CaSA (TIN2010-12312-E) of the Ministry of Education and Science, Spain.

References

Ceri S, Fraternali P, Bongio A (2000) Web modelling language (WebML): a modelling language for designing web sites. Comput Netw 33(1–6):137–157

EA (Enterprise Architect) (2010). http://www.sparxsystems.com

Escalona MJ, Aragón G (2008) NDT: a model-driven approach for web requirements. IEEE Trans Softw Eng 34(3): 377–394

Escalona MJ, Gutierrez JJ, Villadiego D, León A, Torres AH (2007) Practical experience in web engineering. In: Magyar G et al (eds) Advances in information system development. New methods and practice for the networked society. Springer, New York

Fons J, Pelechano V, Albert M, Pastor O (2003) Development of web applications from web enhanced conceptual schemas. In: Song I-Y, Liddle SW, Ling TW, Scheuermann P (eds) Conceptual modeling - ER 2003. 22nd international conference on conceptual modeling, Chicago, IL, USA, 13-16 Oct 2003. Proceedings (LNCS 2813). Springer, Berlin/Heidelberg, pp 232–245

Koch N, Knapp A, Zhang G, Baumeister H (2008) UML-based web engineering. In: Rossi G, Pastor O, Schwabe D, Olsina L (eds) Web engineering: modelling and implementing web applications. Springer, London, pp 157–191

OMG (2008) Documents associated with meta object facility (MOF) 2.0 query/view/transformation. http://www.omg.org/spec/QVT/1.0/

Rossi G, Schwabe D (2008) Modelling and implementing web applications with OOHDM. Web engineering: modelling and implementing web applications. In: Rossi G, Pastor O, Schwabe D, Olsina L (eds) Web engineering: modelling and implementing web applications, Human-computer interaction series. Springer, London, pp 109–155

UML (2005) Unified modeling language: superstructure. Specification, OMG, 2005. http://www.omg.org/cgi-bin/doc?formal/05-07

Chapter 9
Designing Communities for Large-Scale Sustainable Collaboration

I.T. Hawryszkiewycz

1 Introduction

The chapter focuses on ways to design systems that support sustainable communities now increasingly found in the emerging collaborative business and government environments. The increased focus on communities results from greater emphasis on collaboration, which goes beyond small group working in well-defined problems, but involves groups, each working in their specialized areas, that need to collaborate with other groups to achieve a wider business objective by combining their specialized knowledge in large-scale projects such as transportation systems and defence procurement systems and large projects such as designing new car models or aircraft or in evolving supply chains (Sen 2008; Ming et al. 2008).

The challenge posed by complexity is not because systems are composed of many strongly connected components that behave in predictable ways but that they must respond to changes in external environment in innovative ways. This is best exemplified in government systems as described by a report by the Australian government titled 'Tackling Wicked Problems: A Public Policy Perspective' which describes the trend to systems that are difficult to specify, have no clear solutions and many multicausal interdependencies that require changes of behaviour of system stakeholders. McAfee (2006) describes a similar trend in business systems that are increasingly networked with greater emphasis on social relationships (Prahalad and Krishnan 2008) and how social relationships emerge to deal with environmental change. The ability to manage evolving systems in complex environments places new criteria on system design especially:

- Coordination between communities rather than on predefined workflows
- Support for social structures for self-organizing communities

I.T. Hawryszkiewycz (✉)
School of Systems, Management and Leadership, University of Technology, Sydney, NSW, Australia

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_9,

- Ability to change given continually changing requirements with greater emphasis on more descriptive methods guided by architectures (Matthias et al. 2010)

The top-down approach commonly found in system design where requirements are systematically reduced to a design is not relevant here. Matthias et al. (2010) suggest that design approaches, which place greater emphasis on more descriptive methods guided by architectures, are needed. This chapter defines a high-level community architecture to maintain sustainability in community collaboration. It then provides a more detailed set of concepts to model evolving relationships within this architecture and a systematic method based on an open modelling approach to support modelling the myriad of relationships from different perspectives.

2 Managing Design Through a Flexible Architecture

System evolution in complex environments has been described systematically using complexity theory as interpreted in Merali and McKelvey (2006) and Maguire et al. (2006) for information systems and expressed in terms of changes to relationships described as:

- The ability to self-organize at local levels in response to a wide variety of external changes which implies changes to network operations to meet newly negotiated requirements often by reassigning responsibilities and rearranging relationships or adding new artefacts
- Ways to quickly set up new self-contained business activities that address well-defined new activities
- The aggregation of smaller units into larger components to support interconnection between the smaller units and coordinate their activities

Complexity arises from closely linked components that self-adapt to changes to the environment. A design team can be faced with an environment like that shown in Fig. 9.1 for a large-scale tendering process. Relationships here include:

Person to person to exchange knowledge and collaborate and decide on future actions, often requiring creating new relationships with experts or between managers and clients and specification teams.

Group of people to activity in arranging ways to develop and provide a service or develop a product with the group activities changing as new requirements emerge.

Person to knowledge where people seek knowledge to accomplish their task as, for example, getting expert advice and knowing the location of experts.

Person to task to identify define person's role in task as, for example, group of people to task need to define how group relates to task. This depends on the kind of task.

Task to business unit to decide how a task will be used to reach an organizational goal.

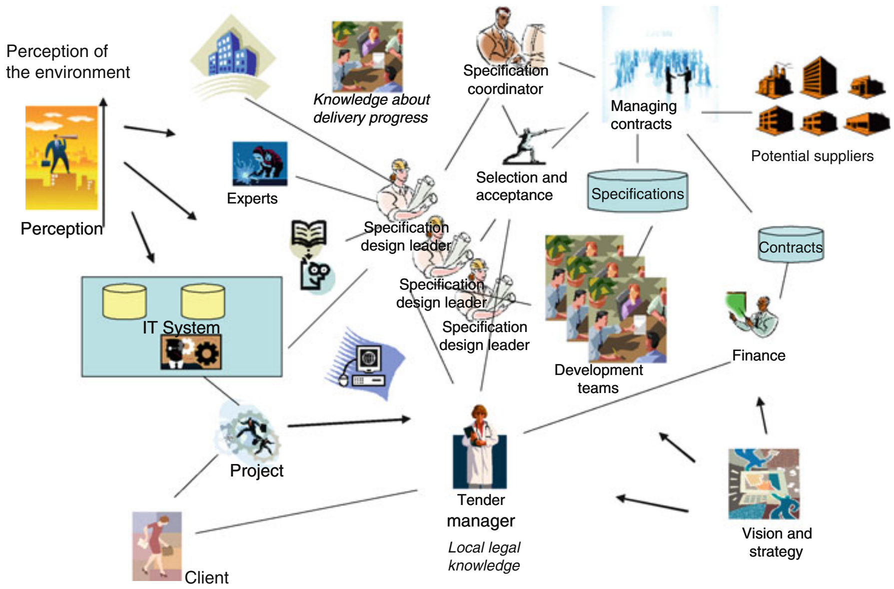


Fig. 9.1 Complexity arises from a range of continually changing relationships

The driving factor is new success factors for complex environments. Traditionally success factors in design focused on criteria such as technical performance such as response time, efficiency of the use of resources or correctness in representation. The emerging systems present new criteria, such as:

Responding quickly to change in system relationships
Flexible reorganization to deal with unexpected events
Improving knowledge sharing through better knowledge flows
Manage creativity and idea selection
Enabling collaboration through adaptable social structures

The goal is to develop models that allow change to be easily expressed. The chapter proposes that change can be proposed from different perspectives and should allow designers to view the system from these different perspectives. Correspondingly models should reflect these perspectives.

3 The Flexible Architecture

Following the suggestion of Matthias, the chapter proposes an architecture that enables flexibility and change. It is shown in Fig. 9.1. It sees progress through a number of knowledge communities each providing its expertise but coordinated through a coordination group. It includes a feedback loop, which is shown by the

Table 9.1 Perspectives

Perspective	Description
Organizational structure	The organizational structure exists to set the vision and provide leadership associated with reward system and create the business structure. Includes the ability to create new high-level communities and link them to existing communities
Business activities	Defines tasks in terms of collaborative concepts (Hawryszkiewycz 2005)
Social structure	The roles and their responsibilities to define the collaboration (Pisano and Verganti 2008) needed in the organization
Knowledge	The products and services produced by the organization and the explicit and tacit knowledge needed to create them
Technology	Provides support for the relationships

dotted line, with the coordinating computing agreeing on goals with the knowledge communities. Thus, for example, one community may focus on population movements, another on housing development and still a third on engineering aspects.

The chapter proposes a systematic approach to model evolving systems in complex environments characterized by continuous emergence to address emerging needs. The chapter proposes that such systems be modelled from different perspectives. A change to be viewed from one perspectives followed by its impact on other perspectives. The major perspectives are shown in Table 9.1. The relationships between the perspectives are shown by the oval shapes in Fig. 9.3.

Table 9.2 describes the relationships especially on their importance on satisfying design criteria.

There are other perspectives include economics and human factor issues.

4 An Example of Modelling: Tendering

The remainder of the chapter illustrates a model for representing the relationships. The modelling method described here is implemented on the open modelling platform at the University of Vienna. The method known as MelCa allows models to be set up from different perspectives and maintains cross references between models as allowed by the open modelling platforms. New objects can be easily added to each perspective. The concepts for each perspective are shown in the centre and can be easily selected and plugged into the model.

4.1 Modelling Organizational Relationships: r1, r2 and r5

Figure 9.4 illustrates the application of the architecture in the design of tendering processes. It shows that the organization issues tenders to provide systems to

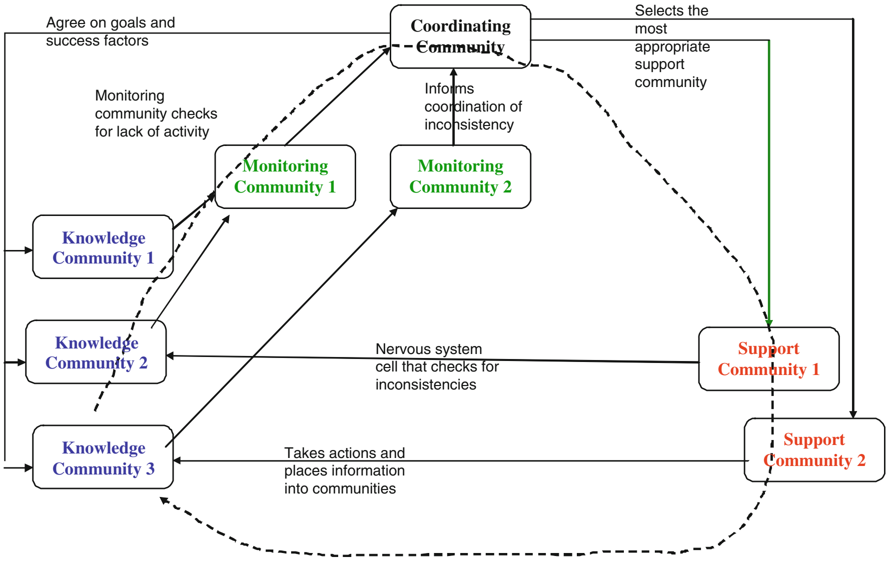


Fig. 9.2 The community architecture

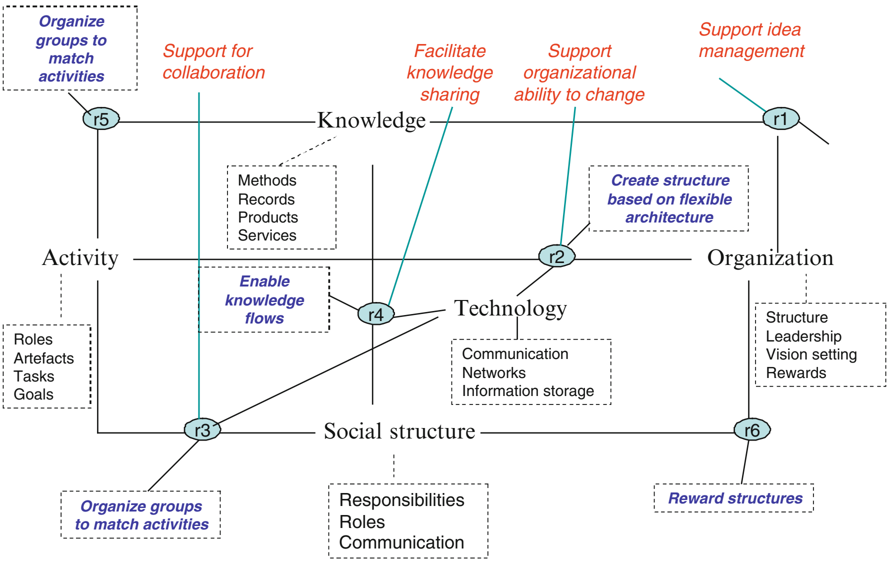


Fig. 9.3 A structure for relationships

support its client needs. The structure is closely related to the architecture in Fig. 9.2. The coordinating community here is the client and tender manager together with leaders responsible for developing the tender specification. The tender in this case is divided into two parts and two tenders to different classes

Table 9.2 Relationships

Relationship	Description and design check list
r1	Expresses the organizational vision as the knowledge needed to realize vision. Define expected outcomes and success factors
r2	Create an activity structure using the architectural model in Fig. 9.3 as a guideline. Provide the ability to reorganize the structure
r3	Define the roles with the right responsibilities to carry out the activity with focus on *facilitating collaboration* and *supporting evolution within the communities* supported by communication technologies
r4	Support knowledge sharing and creation in communities using technology with its main emphasis to meet *knowledge-sharing needs*. Distinguish between knowledge requirements of the different communities
r5	Define the knowledge needs and created by an activity
r6	Provide the structure and incentives for people to align themselves to organizational goals

of suppliers. These are 'tender specification-system-A' and 'tender specification-system-B'.

They are for two different systems that must be combined to form a total system. This is a common procedure in many organizations, where development of each system is carried out by a 'knowledge community'. The model shows roles created to manage the tendering process and link the communities. There is a 'tender manager' two coordinators for developing the specification and two for keeping track of responses. These are key roles in the coordinating community.

Monitoring is the responsibility of roles called response coordinators. These roles report any problems to the coordinating group who takes any corrective action.

The model also defines what information is exchanged in the relationships. Some is shown on Fig. 9.4. For example, response coordinator has the responsibility to maintain liaison with the tenderers. The 'tender manager' primarily arranges the work schedule, and the system coordinators monitor. The emphasis on collaboration here is to maintain awareness of progress of the different activities. The other requirement is to keep track of issues raised with tenderers and ensure that they are adequately answered. Primarily a blog can be considered as a potential technology.

4.2 Modelling Activities: r3

The goal here is to model the different communities in more detail. Figure 9.5 illustrates the activities in the tendering process. It illustrates the way roles are grouped into activities. This now clearly shows the role responsibilities. Figure 9.5 identifies a number of high-level activities including 'developing the specification',

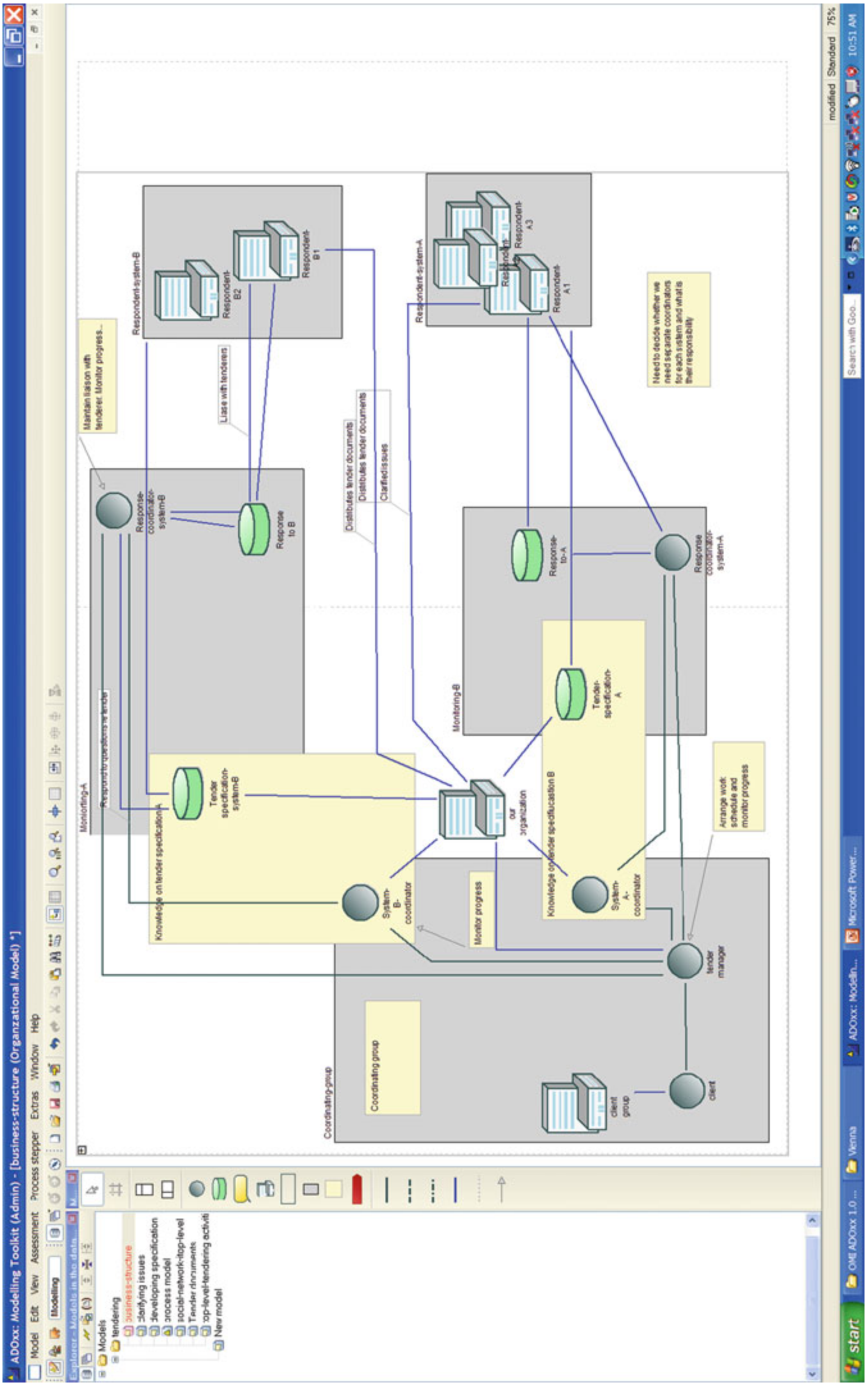


Fig. 9.4 Organizational level model

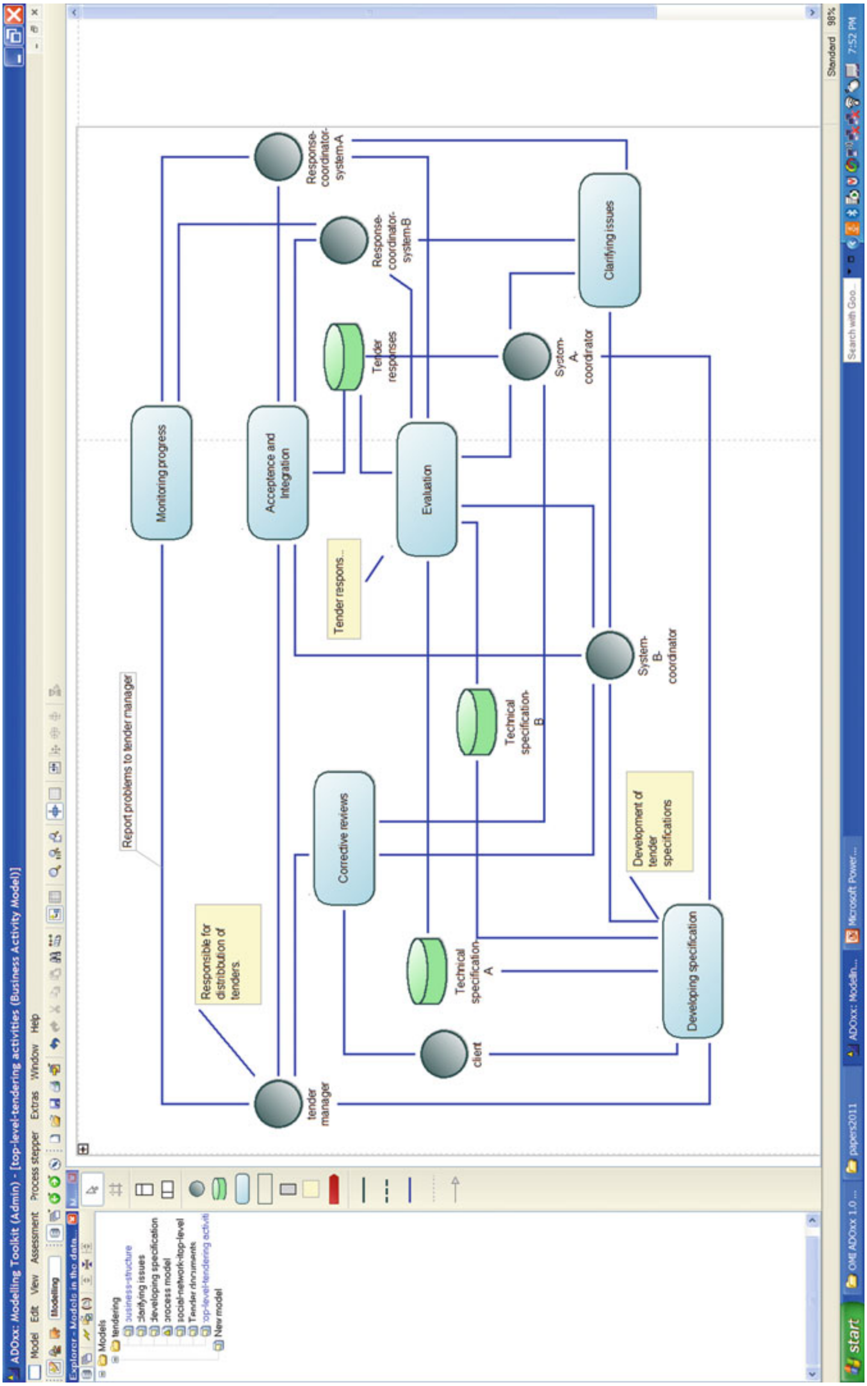


Fig. 9.5 Organizing activities

'distribution and collection of tenders', 'acceptance and integration of responses', 'evaluation' and 'clarifying issues'.

The collaboration in this cases focuses on keeping track of what is happening in each activity. In fact what is needed is a knowledge hub (Rye et al. 2008) for each activity. This will keep records of what processes were followed during the activity and the outcomes achieved by these processes. This will provide a knowledge base to guide future activities.

This perspective is the major perspective for designing the collaborative architecture. It identifies the roles in the business activities and their interactions. These interactions form the basis for the design of the collaborative architecture.

4.3 Modelling the Social Relationships: r5 and r4

Figure 9.6 describes the enterprise social network. It shows roles in the different communities. It describes the interaction between the different roles in the activities. Figure 9.4 is only a broad outline of the social structure. It shows designated roles for each specification that include a project manager that coordinates specification development with team leaders in the technical and environmental areas. The interactions here focus on capturing details of issues raised and their resolution. This perspective shows the interactions between the different roles and the context in which they interact. These interactions can be described in detail and later supported with Web 2.0 technologies. One detailed interaction is shown in Fig. 9.4. This concerns developing guidelines for tender specification. Here the project specification leader works together with the technical and environmental leaders to develop guidelines for developing specifications. These are then presented to the project leaders who may comment on them and suggest revisions.

5 Summary

The chapter described a systematic approach to model dynamically evolving systems. The chapter introduced a community-based approach as a model for large-scale collaboration and proposed a governance structure to coordinate the communities to an agreed goal. It focused on an open modelling method where a flexible architecture is supported by concepts to develop models from different perspectives to systematically manage the variety of relationships in the system.

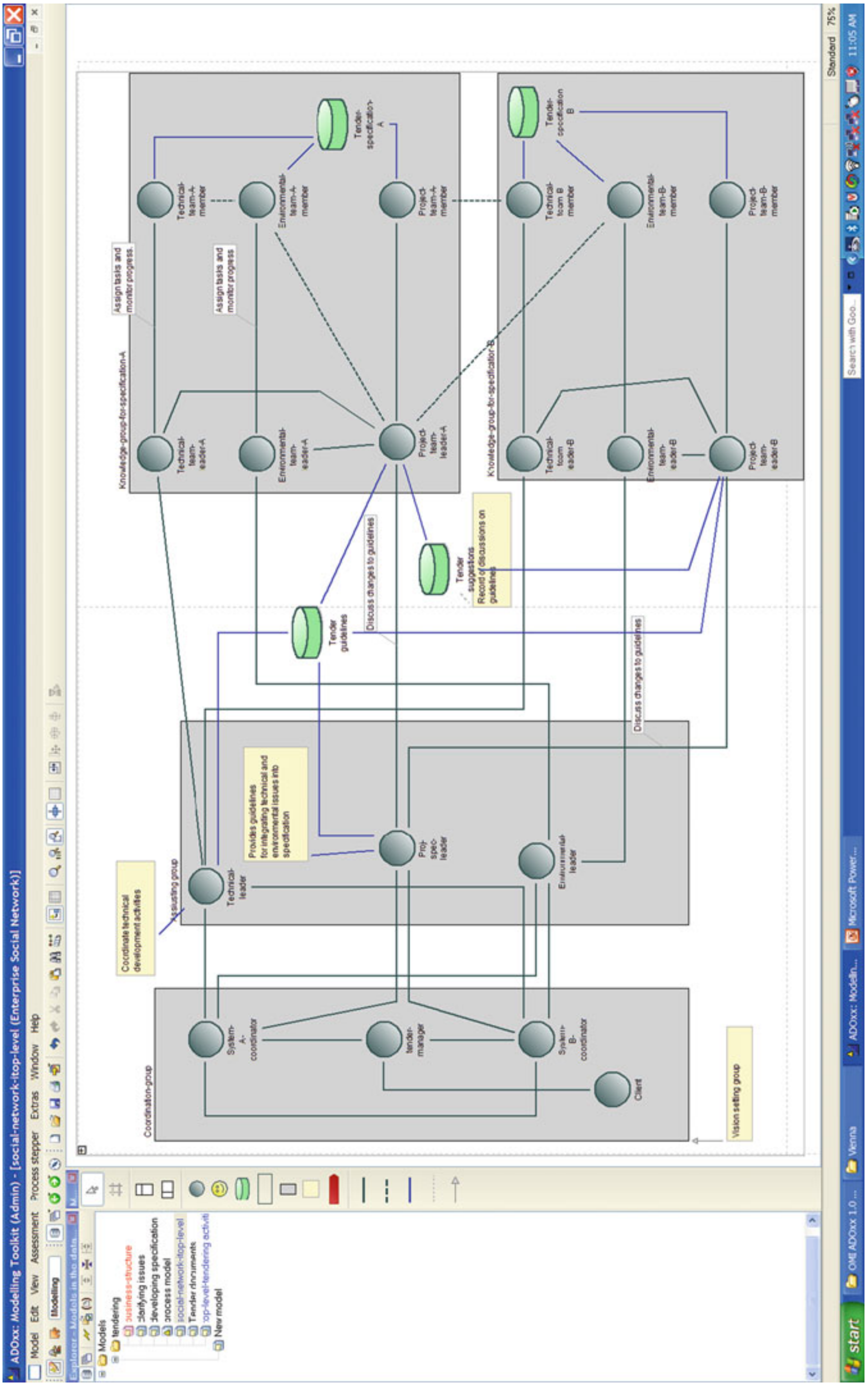


Fig. 9.6 Enterprise social network

References

Australian Government publication. Tackling wicked problems. www.apsc.gov.au/publications07/wickedproblems.pdf

Hawryszkiewycz IT (2005) A metamodel for modeling collaborative systems. J Comput Info Syst XLV(3):63–72

Maguire S, McKelvey B, Mirabeau L, Oztas N (2006) Complexity science and organizational studies. In: Clegg SR, Hardy C, Lawrence TB, Nord WR (eds) The Sage handbook of organizational studies, 2nd edn. Sage, London

Matthias J, Loucopoulus P, Lyytined K, Mylopoulus J, Robinson W (2010) The brave new world of design requirements: four key principles. In: 22nd international conference on advanced systems engineering, CAiSE 2010, Hammamet, Tunisia, pp 470–482

McAfee AP (2006) Enterprise 2.0: the dawn of emergent collaboration. MIT Sloan Manage Rev 47(3):21–28

Merali Y, McKelvey B (2006) Using complexity science to effect a paradigm shift in Information systems for the 21st century. J Info Technol 21:211–215

Ming XG, Yan JQ, Wang XH, Li SN, Lu WF, Peng QJ, Ma YS (2008) Collaborative process planning and manufacturing in product lifecycle management. Comput Ind 59:154–166

Mintzberg H (2009) Rebuilding companies as communities. Harv Bus Rev 84(4):140–143

Pisano GP, Verganti R (2008) What kind of collaboration is right for you. Harv Bus Rev 83 (8):80–86

Prahalad CK, Krishnan MS (2008) The new age of innovation. McGraw-Hill, New York

Rye K, Lee S, Choi H (2008) Modularization of web-based collaboration systems for manufacturing innovation. In: Proceedings of the tenth international conference on enterprise information systems, Barcelona, 12–16 June, pp 174–177

Sen A (2008) The US fashion industry: a supply chain review. Int J Prod Econ 114:571–593

Chapter 10
Cognitive Processes in Object-Oriented Requirements Engineering Practice: Analogical Reasoning and Mental Modelling

Linda Dawson

1 Introduction

Information systems development (ISD) and requirements specification for ISD can be considered as socio-technical activities (Pohl 1994; Urquhart 1998; Kautz et al. 2004). Underlying the technical and social aspects of the requirements engineering (RE) process are skills and concepts from other disciplines such as sociology, organisational science and cognitive science (Hirschheim and Klein 1989; Checkland and Scholes 1990; Urquhart 1998). Some empirical research has addressed abductive reasoning in RE (Menzies et al. 1999) and the importance of perceptions of processes by stakeholders in the RE process (Napier et al. 2009). Although there have been some studies of programmers and the role of experience and knowledge in software (rather than systems) quality (Steen 2007; Gendreau and Robillard 2009), an understanding of how successful systems analysts think and produce requirements models needs to be investigated (Burton-Jones and Meso 2006; Dobing and Parsons 2006; Grant and Reza 2007).

This chapter presents the findings of a multiple case study which examined cognitive modelling for object-oriented (OO) RE based on qualitative data collection and analysis methods. The data and findings presented here are part of a larger study of OO RE practice. In the four cases reported in this chapter, requirements specification involved mental modelling during the transformation from elicitation to concrete models for design and implementation. This mental modelling process involved abstraction and analogous reasoning as well as problem-solving activity.

Section 2 of this chapter provides background in cognitive modelling. Section 3 provides background in OO RE. The research approach adopted is presented in Sect. 4. Section 5 presents the four case studies and findings, and a discussion of the findings is presented in Sect. 6.

L. Dawson (✉)
Faculty of Informatics, University of Wollongong, Wollongong, NSW, Australia

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_10,

2 Cognitive Processes in the Requirements Engineering Process

Most studies of cognition and system modelling and design have been based on positivist-style surveys (Lang and Fitzgerald 2007; Thomas and Bostrom 2007; Shaft et al. 2008) or formal models (Wang and Fu 2004; Overbeek et al. 2007) or normative theoretical studies (Gasson 2004; Nelson and Monarchi 2007; Siau and Wang 2007).

Loucopoulos and Karakostas (1995) identified some factors affecting requirements specification as a problem-solving activity as follows:

- Analysis problems are ill-defined and constantly changing as the organisational context changes and more information about user requirements is gathered.
- Requirements exist in organisational contexts and may be conflicting from differing viewpoints.
- The process of analysis is a cognitive activity, requiring understanding of an abstract problem and development of a logical and internally consistent set of specifications.

Successful requirements specification can be considered to be an art or a skill honed from a great deal of experience (Loucopoulos and Karakostas 1995; Macaulay 1996; Sommerville and Sawyer 1997).

2.1 Thinking and Problem-Solving

Problem modelling for simulation or systems design is an important aspect of many design activities (Schön 1983; Khushalani et al. 1994; Galal and McDonnell 1998). All of these design processes draw on the discipline of cognitive science or the study of human thinking and problem-solving. Problem modelling in systems analysis is not necessarily aimed at modelling "problems" in the general sense (although the term problem is an often-used general term as it is in cognitive science) but is usually aimed at achieving some task. That is, if there is a perception of a task to be accomplished or a problem to be solved, a systems designer must first describe or model the problem or task in a way that facilitates progress towards accomplishing that task or solving that problem.

Problem-solving can be defined as having three phases (Polya 1957; Mayer 1992):

Understanding the problem in terms of what is known (givens) and what needs to be achieved (goal), that is, describing what the task is

Planning a solution (using past experience where appropriate), that is, reusing previous knowledge to describe how to achieve the task

Testing the result, that is, validating and verifying the solution

The investigation of thinking, problem-solving and cognition has long been a major concern of cognitive psychologists, and now, cognitive scientists are seeking to apply cognitive theory to information technology disciplines. General definitions of thinking and problems are a useful starting point for discussing problem modelling for information systems development.

Mayer (1992) defines human cognitive processes in terms of problems and thinking. He defines a problem as having the characteristics of givens and goals and that "... any definition of 'problem' should consist of the three ideas that (1) the problem is presently in some [given] state, but (2) it is desired that it be in another [goal] state, and (3) there is no direct, obvious way to accomplish the change".

Mayer (1992) also considers the terms *thinking, problem-solving* and *cognition* to be equivalent, and although there is a serious lack of agreement among psychologists about the definition of thinking, he suggests a compromise general definition that most psychologists might accept:

- Thinking is *cognitive* but is inferred from behaviour. It occurs internally, in the mind or cognitive system, and must be inferred indirectly.
- Thinking is a *process* that involves some manipulation of a set of operations on knowledge in the cognitive system.
- Thinking is *directed* and results in behaviour that "solves" a problem or is directed towards a solution.

In other words, thinking is what happens when a person solves a problem, that is, produces behaviour that moves the individual from a given state to a goal state.

2.2 Analogical Reasoning

Analogical reasoning is based on analogues, models and examples and "... pervades all our thinking" (Polya 1957). Mayer (1992) defines analogical reasoning as abstracting a solution strategy from one problem and relating that information to a new problem where the original domain is called the *base* domain and the domain to be explained is called the *target* domain. A critical skill in the cognitive processes required in conceptual modelling is the ability to apply the principle of abstraction. Abstraction is a key aspect of analogical reasoning.

Two analogical problems may have a *surface similarity* where the two problems share common characteristics which may not be related to the solution and/or a *structural similarity* where the relations of objects in one problem correspond to the relations of objects in the other problem. If solving one problem helps to solve another problem, we can say there has been an *analogical transfer* of problem-solving strategy. Mayer (1992) suggests that it is *how* the analogies are presented which influences the positive or negative outcome of analogical transfer.

Mayer describes three conditions for successful analogical transfer:

- *Recognition* – in which a problem-solver identifies a potential analogue (or base) from which to reason

- *Abstraction* – in which a problem-solver abstracts a general structure or principle or procedure from the base
- *Mapping* – in which a problem-solver applies that knowledge to the target

Mayer divides analogical reasoning into thinking using analogues, thinking using models and thinking using examples. An analogue has structural similarity but not surface similarity with a target problem.

Knowing a solution plan for an analogous problem is not very useful unless you realise that the problem is analogous to the one you are working on. Studies in this area (Gick and Holyoak 1980; Gentner 1983, 1989; Holyoak and Koh 1987) show that experience and structural similarity of problems are critical in analogical transfer.

Encouraging subjects to abstract solutions from analogues has been the subject of several studies. Gick and Holyoak (1980) propose a *schema induction theory* which suggests that it is easier to induce a general schema from experiences with structurally similar problems in different domains than from a single problem.

Mapping knowledge from a base analogy to a target analogy depends on the overall analogy transfer, that is, recognition of an analogous problem, then the abstraction of useful information that can be used to solve another problem.

2.3 Cognitive Modelling

Mayer describes a *model* of a system as a system "[which] includes the essential parts of the system as well as the cause-and-effect relations between a change in status of one part and a change in status of another part". He uses Gentner's (1983, 1989) *structure-mapping theory* where knowledge about one system (the base) is used to reason about another system (the target). Gentner's system consists of *objects* with *attributes* and *relations* between objects. Studies based on using a water-flow model and a moving-crowd model to understand and reason about an electrical system showed that subjects could reason about electrical circuits based on knowledge of a water-flow model or a moving-crowd model. Mayer and Gallini (1990) also showed in a series of experiments concerning reasoning about models for explaining how radar and pumping systems work, that adding pictorial models to textual descriptions improved problem-solving performance by an average of over 60 %.

The term *mental model* is defined in cognitive psychology (Craik 1943; Johnson-Laird 1983; Norman 1983) as either analogical representation or a combination of analogical and propositional representations. Preece (1994) suggests that "a mental model represents the relative position of a set of objects in an analogical manner that parallels the structure of the state of objects in the world". In the field of human-computer interaction (HCI), a mental model has been defined as "... the model people have of themselves, others, the environment, and the things with which they interact. People form mental models through experience, training and instruction" (Norman 1988, p. 17). Mental models are used extensively in HCI to explain dynamic aspects of cognitive activity, and it is suggested that people build mental models of the world in order to make predictions about an external event before carrying out an action (Preece 1994).

3 Object-Oriented Requirements Engineering

OO models and approaches (Budd 1997; Henderson-Sellers 1997; Jacobson et al. 1999; Booch et al. 2007) are claimed to provide a more natural way of specifying, designing and implementing information systems based on features which include:

- A consistent underlying representation of an identified object throughout the development process
- The encapsulation of static or descriptive characteristics together with the dynamic or behavioural characteristics of an object
- The ability to model complex systems by reusing objects and object components from previously designed systems
- The incorporation of high-level data abstraction facilities including inheritance and polymorphism

In this study, the interest is in how experienced analysts and developers actually use objects in requirements specification.

4 Research Approach

The objective of this project was to investigate whether the cognitive processes described in Sect. 2 above, in particular analogical reasoning and mental models, are used in OO RE and, if so, how these processes are used during the requirements definition process. The research questions for this project were:

- How do analysts use *analogical reasoning and analogical transfer* when producing requirements for a systems development project?
- How do analysts develop *mental models* when producing requirements for a systems development project?

The research approach was based on cases involving taped semi-structured interviews with individual requirements engineers. Each participant was interviewed several times, providing descriptive empirical data.

5 The Case Studies

The following sections describe the four cases with particular reference to the system development experience of the analyst, the development philosophy of the consultant/analyst and the evidence of mental modelling.

Table 10.1 Background information for each consultant

Case	Title	Years in RE	Client	Project
1	Senior consultant	12	Telecommunications	Fault mgt system
2	Consultant	14	Software developer	Life insurance
3	Technical manager	12	Software developer	Stockbroking
4	Principal consultant	15	State govt	Web-based transactions

Participants were recruited through industry. Some participants provided contacts for subsequent participants. There has been no attempt to select participants based on specific background characteristics. The common factor is that all the participants were currently working in the field of OO requirements specification. The contextual information for each consultant interviewed is summarised in Table 10.1.

5.1 *Findings*

The presentation of findings is based on illustrated narrative style, or an oral narrative told in the first person, as described by Miles and Huberman (1994) and Myers (2010) and as used in Fitzgerald (1997) and Urquhart (1998). This approach is described as (Miles and Huberman 1994) "...each part of the sequence is followed by a series of illustrative excerpts [quotes from the transcripts]" which does not resort to explicit coding but looks for "... key words, themes, and sequences to find the most characteristic accounts". Where transcript data is quoted directly, the researcher's questions or interactions are shown as bold italic and the participant's as plain italic.

5.1.1 Case 1

The consultant considers himself to be a very experienced developer. When asked to comment on the main advantages of an OO approach to system development, he replied: "The ability to evolve code in isolation behind interfaces just seems to be the key benefit ... the ability to make significant changes to significant amounts of code and not destabilise parts of the system which you are not directly working in".

The client was a large telecommunications organisation. The project was a fault management system for managing planned and unplanned outages in a transmission network. It was a 5-year project and had involved two and a half years of development work for the analyst. The first-stage deliverables were a suite of requirements and analysis specifications. The requirements model was a use-case model, and there was also a prototype.

Analogy Transfer and Abstraction

The methodology used for system development was an in-house OO method. It was based on other methodologies that members of the team were familiar with. The development process was influenced by the people who were available – and the fact that they had come with considerable industry experience in this kind of system development. "We sampled from methodologies that we were familiar with. We used bits of other methodologies as appropriate.... five of the developers had significant experience of building similar systems elsewhere… What that meant was there were three or four people who were able to contribute to a methodology that picked up bits and pieces from a number of influences… They just all brought their biases and their interests and thoughts".

Mental Modelling

The specifications were drawn up after the consulting analyst joined the team and a user group was part of that process.

> I would say that [the class model] was drawn up quickly within the space of probably a week to two weeks. But I would say that there were fragments of that model getting developed in a couple of people's heads for probably three months beforehand. The development of that model was not done publicly, or the first cut of it, so after that it was tossed to the team and it just diverged … As the requirements were being collected as the requirements modelling was being done. But I would say that it was being done largely privately and it was not written down until the last minute when it was just a dump.

Further questioning on mental modelling followed: "How and when do these mental models start forming inside your mind. You have said that this is what happens and at some point it gets turned into hard copy … do you think in general people doing this kind of work are mulling around mental models in their head?"

> Firstly, when we talked about collecting requirements and putting them into the requirements model, we talked about thinking of objects, and … there will be someone whose responsibility it will be … to start casting bits of models together. So as they go through they will be listening to discussions and working on the requirements model and they may or may not be writing things down, … You will be listening very carefully and collecting and cataloguing constraints and refining the abstractions in your mind.

In this case, the information that was used to build the mental models came from a group of users, from project meetings and the general requirements gathering process and activities. The more technical aspects were done later on with discussion with other members of the analysis team. The consultant explicitly saw this activity as abstraction:

> You will come up with will be an abstraction in your own mind, which you probably cannot fully express, … you wouldn't want to. For me anyway it's a mistake to try and rush in and write that down and stick it into a case model. … So you might carry round an event or an account or a customer object or something, you carry around a picture of how that is shaping in your mind. Someone in a meeting or a discussion will say, "Of course, you know

> we only ever had one of these, and that will change" and you can say Ah! Test that against my understanding of what a customer, or event or a facility or whatever the abstraction is going to be and that might either verify or it might contradict it. If it verifies it you probably let things go and move on to the next point. If it contradicts it, you need to pick it up and mine that and get to the bottom of that.

5.1.2 Case 2

The *consultant* is a director and a partner of a small organisation whose members work as consultants in the OO field. He had been with the organisation for just on a year and had been doing RE for about 22 years. The consultant did not use any specific methodology in this project or any other projects. "I haven't been, let's say, an advocate of any particular methodology from start to finish ... See I don't believe in methods as such ... What I talk about is a underlying concept rather than a methodology ... you need to have rigour and the diagramming notations and the steps in the methodologies give you that, but you also need to play in the sandpit".

The client was a commercial software developer that builds generic packages and then sells them on to clients in the financial sector. The organisation believed that it needed to move to OO systems. The consultant was engaged "... not so much as a requirements analyst but more as an OO mentor. Now in that role one of the first things we did was requirements and then moved on to design". The objective of the project was to produce a receipting component for a larger insurance system that would be a commercially saleable stand-alone product.

Analogy Transfer and Abstraction

A discussion about how the consultant went about initial modelling of requirements led to the following illustrative comment about the usefulness and necessity of ad hoc or informal models "... and in every project I've ever worked on be it a mathematical project or software development project there's been a few key pictures. The one I'm working on at the moment is the billing cycle – it's a wheel and its got the steps in the billing cycle on it and that's in everybody's head and everybody talks in those terms and it's just the key base thing – it's the conceptual core of the thing ... I'm a great believer in ad hoc diagrams that give the picture that springs from your understanding of the problem and in a lot of OO work the process of development hinges on one or two of these pictures. ...[and] the trouble with that [using ad hoc diagrams] as a methodology is that its difficulty is that you can't capture it, you can't describe it in some way that anyone [else] can really use it and that's precisely its strength because it handles those parts of the things that don't fit in the normal descriptions and every project's got an aspect like that".

Mental Modelling

The consultant saw OO modelling as "... a superb way of modelling the real world" which allows a high level of abstraction "... I'm unlearning some of my [data modelling] prejudices being back with [my old colleagues] and starting to look at objects and classes more in terms of services than as data and deferring the internal structure later and later and later into the design".

> I think that I do immediately start thinking of key objects during requirements gathering, not in any formal way, they just pop into one's head.

5.1.3 Case 3

The *consultant* had been involved in OO systems for about 5 years. He believed that although OO approaches have certain advantages, they also have some limitations for developing systems: "The problem with using DFDs is that the models were far too data-centric which was fine if you were doing a lot of retrieval but to do good transaction processing was quite awkward and you really did need very high levels of expertise to get it right in the RDBMS world. That is much less so [in OO] and therefore you can end up with much simpler solutions with OO techniques as long as you keep the 'propeller heads' so to speak away, you actually end up with systems which are very easy to understand and easy to maintain".

The *client* project was a back office system for stockbrokers configurable to individual client needs. It was neither an off-the-shelf package nor a one-off so there was a set of core requirements. Assistance was provided by the organisation to clients in customisation and ongoing support.

Analogy Transfer and Abstraction

The consultant believes that as a professional, much of his requirements technique comes from his knowledge and experience on other projects. "It works in two ways. There is using your experience to recognise 'Hang on I've seen something like that before . . . Yes, I recognise what it's doing therefore I can do it'. Secondly I know . . . I don't have the same view of the business as a user because I'm used to the view from the software development side . . . the side of the business I understand and therefore when I'm requirements gathering I actually understand what the user is talking about and can actually relate that to software development so therefore I can make sure I try and work my requirements gathering around 'OK, if I had to build that what information would I need? Have they given me enough? If I built that what are the exception cases? There are some of the questions about what they want to do if it doesn't meet these criteria, all of that. And that's predominantly drawn on a nearly 20 year career and 12 years in the same environment".

Mental Modelling

It was not until the requirements gathering was finished that the consultant or team started UML modelling "... all through requirements gathering we are talking textual, primarily". When questioned about building mental models at this elicitation stage, this consultant agreed "... I don't put anything on paper – that these mental models are sort of living in the back of the mind and as new information comes in that it sort of alters or adds to that mental model ... and its very abstract and not really anything that is put on paper or shown anyone".

5.1.4 Case 4

The *consultant* worked for a business consultancy which provides IT consultancy and educational services to a broad range of clients. The organisation's philosophy is outlined on their website as follows: "We do not subscribe to a single, rigid methodology. Each assignment is treated as a unique challenge. We tailor our approach to meet the specific requirements of each client, drawing on a wide range of well-researched techniques and the combined experience of our consultants".

The *client* was a central support group within government which was assisting other departments or client organisations with the implementation of on-line service delivery to the general public. The client organisations were independent organisations within the government that provide or sell products or services to the general public.

Analogy Transfer and Abstraction

In this project, the analogy transfer and abstraction was explicitly built in to the RE process. The client organisations were given a general pattern for a transaction that could be configured to how the client/end client wanted it. "There is one general methodology, one [generic] object model and there are a set of seven different templates, for each [common] transaction type that you can use and you can tailor the templates ... i.e. they are half filled out - it is not a blank form".

The first task for the client was to work through the general use-case flow diagram and instructions to see how well their transaction matched the common model. This "goodness-of-fit" test was important because pricing was based on it. The more variation from the common model, the more it cost the client organisation.

Customising the template involved modifying the basic flow diagram (based on the "goodness of fit") and the object model, modifying the use-case script by striking out (not removing) elements, so that someone could look across the page and see what had been changed. Few modifications were made to the object model by client organisations. The generic model seemed to apply to most situations. The generic templates presented an abstract view of transactions which could be mapped to different end-client systems as necessary.

Mental Modelling

The elicitation process was not seen as specifically OO, but this consultant claimed to think in terms of objects at this stage because "...that's the way I think ... so it's hard for me to unbundle it ... we don't say to them [the client] we are really talking about objects and we are using an OO methodology. We just do it and they just want to specify their transactions".

6 Discussion

The findings from this study show that modelling for OO requirements specification relies on cognitive skills including abstraction and mental modelling together with problem-solving and reasoning skills particularly analogical reasoning skills on the part of the analyst.

In the four cases reported in this chapter, requirements specification involved mental modelling during the transformation from elicitation to concrete models for design and implementation. Overall the four analysts believed that they were continually "modelling in the mind" during the elicitation process and that these mental models were further refined in the mind before they were communicated to others (users or fellow analysis team members) or before they were committed to paper. The consultants examined and discussed problems encountered with the users during elicitation and then modelled the solutions to those problems in their own minds before committing to concrete models used for design and implementation. This mental modelling process appeared to involve abstraction and analogous reasoning as well as problem-solving activity.

This study highlights several factors which are important in understanding the practice of OO RE in particular and RE in general:

Analogical Transfer: A critical skill in the cognitive processes required in conceptual modelling and conceptual reuse is the ability to apply the principle of abstraction where a problem is seen as a general instance based on a basic known analogue. The recognition of structural similarity and analogical transfer may be a factor for experienced analysts when approaching new problems or systems development projects. Gentner's (1983) structured mapping theory which consists of objects with attributes and relations between objects can be mapped directly to OO modelling approaches for RE.

Cognitive Mental Modelling: Mental modelling used by analysts when developing requirements or design models where models are tested privately before developing more concrete paper-based models is consistent with general cognitive modelling theory in both problem-solving theory and cognitive psychology.

As with many professional activities involving analysis and design (Schön 1983; Khushalani et al. 1994; Johnston 1999; Galal and McDonnell 1998), OO RE and RE in general can be considered to be a creative process (Lawson 2006) particularly on the part of the requirements engineer or the analyst undertaking the requirements specification. Creative modelling as demonstrated by the analysts in this study relied on cognitive skills including abstraction and mental modelling and problem-solving and reasoning skills particularly analogical reasoning skills on the part of the analyst.

Also, the RE process is fundamentally a social process involving two main groups: the users/clients and the professional consultants (Urquhart 1998; Loucopoulos and Karakostas 1995; Macaulay 1996), and this social process requires understanding by all parties to reach agreement. The facilitation of understanding and agreement requires creative modelling skills on the part on the analyst to produce understandable usable models.

It is clear that the consultants in this study demonstrated Mayer's three conditions for successful analogical transfer:

- *Recognition* – recognising similarities with previous problems or system characteristics
- *Abstraction* – abstracting a solution from one or more previous systems
- *Mapping* – using the characteristics of this solution in the new system problem space

Understanding this analogical transfer in RE may allow system designers to develop appropriate tools and modify to explicitly use this process in future or modified development methodologies.

7 Conclusion

The research project described in this chapter has provided evidence that mental modelling based on abstraction and analogous reasoning is used by professional analysts in the development of requirements specifications for system development. Although this study was specific to OO system development, it has implications for requirements engineering in general. It has shown that requirements engineering like other professional analysis and design activities is a highly cognitive and creative activity.

A longitudinal study of professional systems developers – using both OO approaches and non-OO approaches – is currently underway. An investigation of emerging techniques and approaches to requirements specification and system development such as agile development and extreme programming is being incorporated in this study. The long-term aim is to gain more knowledge of how large real-world organisational systems are being specified and developed and which tools and techniques facilitate the successful specification for the design and implementation of these systems.

References

Booch G, Maksimchuk R et al (2007) Object-oriented analysis and design with applications. Addison-Wesley, Upper Saddle River
Budd T (1997) An introduction to object-oriented programming. Addison-Wesley, Reading
Burton-Jones A, Meso PN (2006) Conceptualizing systems for understanding: an empirical test of decomposition principles in object-oriented analysis. Inf Syst Res 17(1):38–60
Checkland P, Scholes J (1990) Soft systems methodology in practice. Wiley, Chichester
Craik KJW (1943) The nature of explanation. Cambridge University Press, Cambridge
Dobing B, Parsons J (2006) How UML is used. Commun ACM 49(5):109–113
Fitzgerald B (1997) The use of systems development methodologies in practice: a field study. Inf Syst J 7:201–212
Galal G, McDonnell JT (1998) A qualitative view of requirements engineering. Third Australian conference on requirements engineering, Geelong, Australia
Gasson S (2004) A framework for behavioral studies of social cognition in information systems
Gendreau O, Robillard PN (2009) Exploring knowledge flow in software project development. International conference on information, process, and knowledge management: Eknow 2009, Proceedings, pp 99–104
Gentner D (1983) Structure mapping: a theoretical framework. Cognit Sci 7:155–170
Gentner D (1989) The mechanisms of analogical learning. In: Vosniadou S, Ortony A (eds) Similarity and analogical reasoning. Cambridge University Press, Cambridge
Gick ML, Holyoak KJ (1980) Analogical problem solving. Cogn Psychol 12:306–355
Grant ES, Reza H (2007) Towards the development of a rigorous model-driven domain-specific software engineering environment. Third IASTED international conference advances in computer science and technology, Phuket, Thailand
Henderson-Sellers B (1997) A book of object-oriented knowledge. Prentice-Hall, Upper Saddle River
Hirschheim RA, Klein HK (1989) Four paradigms of information systems development. Commun ACM 32:1199–1216
Holyoak KJ, Koh K (1987) Surface and structural similarity in analogical transfer. Mem Cognit 15:337–340
Jacobson I, Booch G et al (1999) The unified software development process. Addison Wesley Longman, Reading
Johnson-Laird PN (1983) Mental models. Cambridge University Press, Cambridge
Kautz K, Hansen B et al (2004) The utilization of information systems development methodologies in practice. J Inf Technol Cases Appl 6(4):1–20
Khushalani A, Smith R et al (1994) What happens when designers don't play by the rules: towards a model of opportunistic behaviour and design. Aust J Inf Syst 1(2):2–31
Lang M, Fitzgerald B (2007) Web-based systems design: a study of contemporary practices and an explanatory framework based on "method-in-action". Requir Eng 12(4):203–220
Lawson B (2006) How designers think: the design process demystified. Architectural Press, Oxford/Boston
Loucopoulos P, Karakostas V (1995) Systems requirements engineering. McGraw-Hill, London
Macaulay L (1996) Requirements engineering. Springer, London
Mayer RE (1992) Thinking, problem solving, cognition. W.H. Freeman and Company, New York

Mayer RE, Gallini J (1990) When is an illustration worth a thousand words? J Educ Psychol 82:715–726
Menzies T, Easterbrook S et al (1999) An empirical investigation of multiple viewpoint reasoning in requirements engineering. IEEE international symposium on requirements engineering, proceedings, pp 100–109
Miles MB, Huberman AM (1994) Qualitative data analysis: an expanded sourcebook. Sage Publications Inc, Thousand Oaks
Myers M (2010) Qualitative research in information systems. 1999
Napier NP, Mathiassen L et al (2009) Combining perceptions and prescriptions in requirements engineering process assessment: an industrial case study. IEEE Trans Softw Eng 35 (5):593–606
Nelson HJ, Monarchi DE (2007) Ensuring the quality of conceptual representations. Softw Qual J 15(2):213–233
Norman DA (1983) Some observations on mental models. In: Gentner D, Stevens AL (eds) Mental models. Lawrence Erlbaum Associates, Hillsdales, pp 7–14
Norman DA (1988) The psychology of everyday things. Basic Books, New York
Overbeek S, van Bommel P et al (2007) Characterizing knowledge intensive tasks indicating cognitive requirements; scenarios in methods for specific tasks. Situat Method Eng: Fundam Exp 244:100–114
Pohl K (1994) The three dimensions of requirements engineering: a framework and its applications. Inf Syst 19(3):243–258
Polya G (1957) How to solve it. Doubleday/Anchor, Garden City
Preece J (1994) Human-computer interaction. Addison-Wesley, Wokingham
Schön DA (1983) The reflective practitioner: how professionals think in action. Basic Books, New York
Shaft TM, Albert LJ et al (2008) Managing change in an information systems development organization: understanding developer transitions from a structured to an object-oriented development environment1. Inf Syst J 18(3):275–297
Siau K, Wang Y (2007) Cognitive evaluation of information modeling methods. Inf Softw Technol 49(5):455–474
Sommerville I, Sawyer P (1997) Requirements engineering: a good practice guide. Wiley, Chichester
Steen O (2007) Practical knowledge and its importance for software product quality. Inf Softw Technol 49(6):625–636
Thomas DM, Bostrom RP (2007) The role of a shared mental model of collaboration technology in facilitating knowledge work in virtual teams. 40th annual Hawaii international conference on system sciences (HICSS'07). Hawaii, p 37
Urquhart C (1998) Analysts and clients in conversation: cases in early requirements gathering. Nineteenth international conference on information systems, Helsinki, Finland
Wang XC, Fu XL (2004) The cognitive skill acquisition of information reduction. Int J Psychol 39 (5–6):57

Chapter 11
Development of a Prototype Knowledge Discovery Portal for Energy Informatics

Gabriel J. Costello, Ray Clarke, Brian Donnellan, and John Lohan

1 Introduction

The issue of environmentally sustainable development and the area of ‘green IS’ has recently begun to appear on the information systems research agenda (Watson et al. 2010; Boudreau et al. 2008; Chen et al. 2008; Webster 2009). The aim of this chapter is to contribute to this debate by describing the development of a prototype knowledge discovery portal (KDP) for energy informatics. The research domain is Ireland which is increasingly challenged to implement energy efficiency targets and renewable energy systems (RES) to decrease dependency on imported fossil fuels. The success of the Irish economy in the 1990s resulted in an increase in energy demand with an associated rise in greenhouse gas emissions. Recently, an energy white paper has been published on delivering a sustainable energy future for Ireland (DCMNR 2007). Priority has been given to energy management practices in industry, the services sector and the public sector (Forfas 2010). Also, an Irish government report has outlined the vision of how the development of the green economy can create employment and exploit export opportunities (DETI 2009). The longer-term goal of developing this portal is to provide a mechanism to disseminate information on energy efficiency and renewable energy technologies to a number of sectors: community, educational, industrial and research.

This chapter will be presented as follows. Firstly, the background and drivers of the development of a KDP are outlined. Then, a literature review is presented. The research approach will then be outlined, followed by an overview of the knowledge discovery portal (KDP) for energy. Finally, a discussion of the significance of the KDP will be presented together with conclusions and plans for future work.

G.J. Costello (✉) • R. Clarke • J. Lohan
Galway-Mayo Institute of Technology, Galway, Ireland

B. Donnellan
National University of Ireland, Maynooth, Ireland

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_11,

2 Background

The research is based in a third-level educational institution in Ireland that has a total enrolment of approximately 9,000 students. The mission statement and vision of the institution is to ‘develop life-long learning opportunities through our teaching and research, by supporting regional development consistent with national higher education policy’. The knowledge discovery portal for energy (KDP-energy) concept supports all these broad institute goals. It was identified by the institution’s Centre for the Integration of Sustainable Energy Technologies (CiSET) in 2007 as an efficient means of disseminating information to a wide range of stakeholders including the wider external community, undergraduate students, internal/external researchers and industry. In recent years, a department within the institution has undertaken a number of initiatives to build a state-of-the-art teaching and research infrastructure which consists of an undergraduate energy training laboratory, campus energy monitoring programme and the development of postgraduate research facilities through CiSET.

Each of these initiatives is heavily instrumented so that energy system performance or energy consumption can be continually monitored. The prototype KDP presently includes one solar thermal collector. Figure 11.1 presents a vision of the hardware energy infrastructure on which the KDP software application will be built.

A brief literature review of the emerging area of energy informatics is presented in the next section.

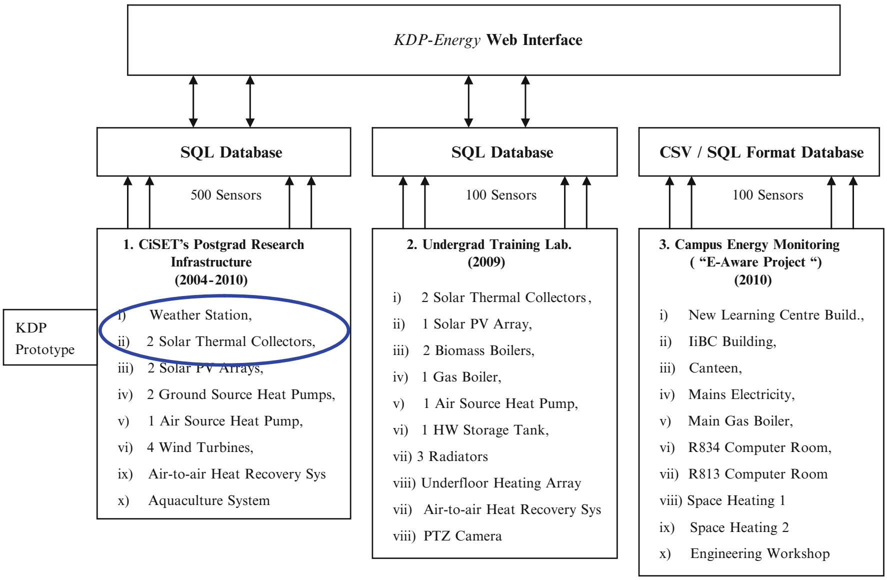


Fig. 11.1 Top-level schematic of the energy efficiency and sustainable energy initiatives

3 Literature Review

For every extra kilowatt hour (kWh) of electricity used by ICT equipment, the US economy increased overall energy savings by a factor of around 10. That was the conclusion of research by the American Council for an Energy-Efficient Economy (ACEEE), which claims that the current focus on soaring IT energy use has distracted from the net energy savings IT systems deliver for other sectors of the economy. The report, *Information and Communication Technologies: The Power of Productivity* (Laitner and Ehrhardt-Martinez 2008), argued that IT's role in the design of new products and services and its ability to replace many energy intensive processes have led to a net improvement in energy use. It proposed that the ICT systems were subject to an 'energy paradox' whereby 'more attention tends to be paid to the energy consuming characteristics of ICT than to the broader, economy wide, energy saving capacity that emerges through their widespread and systematic application'.

Despite the fact that green IT (Sobotta et al. 2009) now holds a significant position on the strategic agenda of many large corporations and government agencies, relatively little has been published in the IS academic literature about either the theoretical or practical aspects of managing and measuring this nascent phenomenon. Academics are starting to look at green IT from three viewpoints:

1. The innovation perspective where IS researchers (Webster 2009) have drawn attention to the potential of IT Innovation (in this case green IT Innovation) to contribute to company competitiveness.
2. Competitive strategy perspective where researchers such as Michael Porter explore the concept of 'innovation offsets' – where companies can 'not only lower the net costs of meeting environmental regulations, but can lead to absolute advantages' over competitors (Porter and Van Der Linde 1996). Banking on growing consumer demand for green products and services, some retailers have developed 'sub-brands' with a green angle. Increasingly, manufacturers across industries will use green products and green corporate behaviour as a way to appeal to this growing segment of customers.
3. Corporate social responsibility.

Appelman and Krishnan (2010) argue that while ICT energy efficiency objectives are important, they need to be supplemented by eco-efficiency programmes to maximise the impact. Furthermore, in a point that is very relevant for this chapter, they conclude that better building design, management and automation could save 15 % of buildings emissions. They calculate that globally, smart buildings' technologies would enable 1.68 $GtCO_2e$ of emissions savings, worth €216 billion ($340.8 billion). An important distinction to be made here is the key difference between 'green IT' and 'green IS' (Boudreau et al. 2008):

- An information technology (IT) transmits, processes or stores information.
- An information system (IS) is an integrated and cooperating set of software using information technologies to support individual, group, organisational or societal goals.

The development of this KDP for energy, we argue, meets the definition of a 'green IS' as it uses ICT to support the 'individual, group, organisational' and 'societal goals' as outlined above.

Some authors (Robèrt et al. 2002) advocate a systems approach to sustainability, and they map essential elements for developing sustainability. This includes the documentation of how each of the elements relates to the application of sustainability tools pioneered by the authors. Another interesting addition to the sustainability debate is the cost reduction and increased efficiencies associated with the move to 'cloud' computing. In a recent report commissioned by Microsoft, it was found that carbon emissions were reduced by 30 % when cloud-computing applications were used compared with on-premise installed business applications (Accenture 2010). Vezzoli and Manzini (2008) contend that ICT has the capacity to support environmental sustainability by gathering together 'individuals concerned about the same topic, to collect the critical mass required to do something, and act accordingly' (p. 35).

Appendix 1 presents a cross section of e-learning initiatives deployed by educational institutes in Europe, North America and Australia to promote sustainable energy systems. These were used to provide a background search for development of the institution's KDP.

The next section will outline the research approach.

4 Research Approach: Design Science

This section will provide an overview of the research approach employed in this study. The seminal paper by Hevner et al. (2004) provides 'a concise conceptual framework and clear guidelines for understanding, executing and evaluating [design-science] research' (p. 75). They go on to state that design science is fundamentally a problem-solving paradigm that seeks to 'create innovations that define the ideas, practices, technical capabilities, and products through which the analysis, implementation, management and use of information systems can be effectively and efficiently accomplished' (p. 76). Furthermore, they trace the roots of design science to Simon's well-regarded publication in the 1960s of *The Sciences of the Artificial*. In an earlier work, Markus et al. (2002) outline their use of design science to address the challenge of developing executive information systems (EICs). An important concept in design science is that of an IT artefact which is summarised in Table 11.1 below.

Hevner et al. describe the primary goal of their paper is 'to inform the community of IS researchers and practitioners of how to conduct, evaluate, and present design-science research' (p. 77). According to Walls et al. (1992), design is both a

Table 11.1 A taxonomy of IT artefacts (From Hevner et al. 2004, p. 77)

Artefact	Description
Constructs	Vocabulary and symbols
Models	Abstractions and representations
Methods	Algorithms and practices
Instantiations	Implemented and prototype systems

Table 11.2 Design-science research guidelines (From Hevner et al. 2004, p. 83)

Guideline	Description
Guideline 1: Design as an artefact	Design-science research must produce a viable artefact in the form of a construct, a model, a method or an instantiation
Guideline 2: Problem relevance	The objective of design-science research is to develop technology-based solutions to important and relevant business problems
Guideline 3: Design evaluation	The utility, quality and efficacy of a design artefact must be rigorously demonstrated via well-executed evaluation methods
Guideline 4: Research contributions	Effective design-science research must provide clear and verifiable contributions in the areas of the design artefact, design foundations and/or design methodologies
Guideline 5: Research rigour	Design-science research relies upon the application of rigorous methods in both construction and evaluation of the design artefact
Guideline 6: Design as a search process	The search for an effective artefact requires utilising available means to reach desired ends while satisfying laws in the problem environment
Guideline 7: Communication of research	Design-science research must be presented effectively both to technology-oriented as well as management-oriented audiences

process (or set of activities) and a product (artefact) while Markus et al. (2002) explain that a build-and-evaluate loop is usually iterated a number of times in the development of an artefact. Table 11.2 summarises seven guidelines proposed by Hevner et al.

Recent research on the implementation of DSR has found that while the guidelines of Hevner et al. are largely endorsed, caution needs to be exercised when applying them (Venable et al. 2010).

We will evaluate the research in terms of the design guidelines above in the discussion section of this chapter. However, we must first provide an overview of the artefact developed in this study.

5 KDP Prototype

This section describes the current status of the KDP prototype which has been developed for the area of energy informatics. The home page is shown in Fig. 11.2, and it has been designed to deliver relevant information on energy systems and efficiency to the following four user groups:

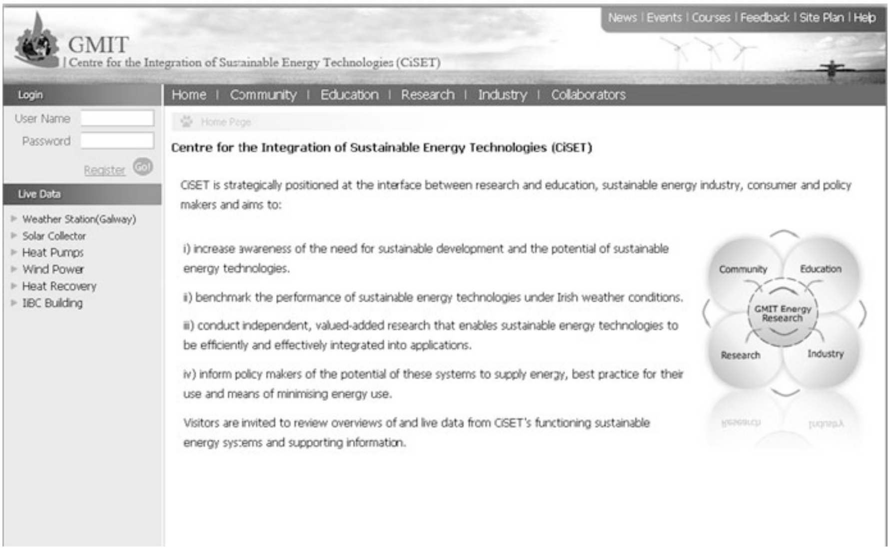


Fig. 11.2 Home page of the prototype KDP for energy

1. *Wider community*: Including both general interest audience (through 'general interest' content on the *home page*) and proactive communities who are undertaking different community-based energy awareness projects within their communities (e.g. E-Aware Project, SEZ Project at Tuam and the Sustainable Islands Project on Innis Oir) through content on the *community pages*.
2. *Education*: Students undertaking formal education, with the priority being third/fourth level; second level and primary level (in that order). This information will be made available through the *education pages*.
3. *Research*: Activities of the Centre for the Integration of Sustainable Energy Technologies (C*i*SET) and its collaborators will be reflected on the *research pages*.
4. *Industry*: Delivers material that supports related industry groups such as solar industry, wind power industry, BER assessors and building/construction industry on the *industry pages*.

The KDP home page is presented in Fig. 11.2 below.

5.1 e-Learning Methodology

An e-learning environment has been developed for two technologies, a solar thermal collector representing one element of the topic – the energy system – or contemporary phenomenon being investigated, together with a weather station representing the real-life context.

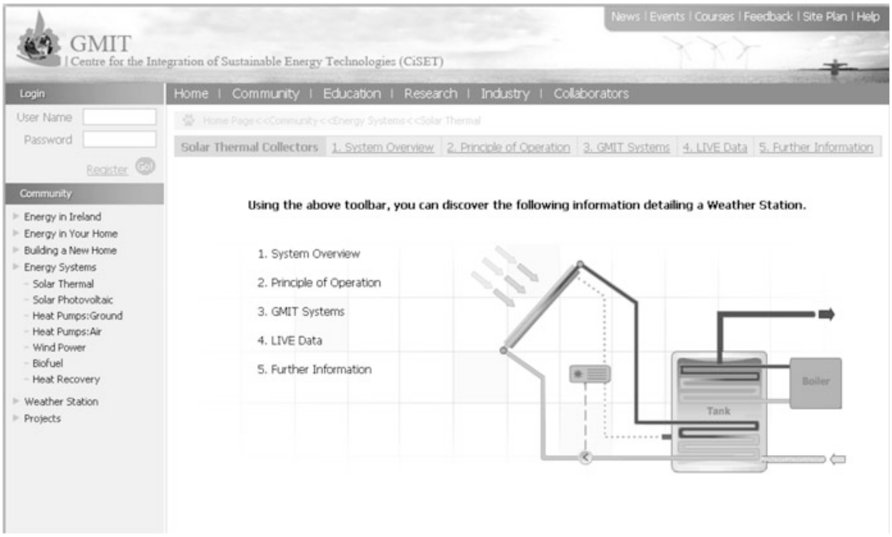


Fig. 11.3 Index page for the educational component for on-line solar system

Figure 11.3 presents the IS interface for the solar thermal collector. Figure 11.4 shows the five-step educational components of an on-line thermal collector system. This sequence was specially designed to support e-learning. For both systems, live data can be retrieved from the relational database (Step 4) (if viewed from within the firewall) and displayed on a flash-enhanced system schematic. While it is possible to retrieve historical weather data, such an interface has yet to be developed for the solar thermal collectors.

5.2 *Graphical User Interface and Programming Languages*

Emphasis is placed on the use of simple, yet attractive, consistent and engaging graphics, which are supplemented by small portions of relevant information that pop up when the user moves a mouse over certain active zones. The intention is to engage the user with a combination of interactive graphics and take them on an educational journey of ‘discovery’ (steps 1–3) that culminates in an opportunity to view real (live) system data (step 4). Links to other useful or relevant sites are also presented under Further Information (step 5).

This journey should be of equal interest to specialists in the energy field and non-specialists alike, whether student or casual visitor, researcher or industrialist.

This demo web portal has been developed using the following software: Microsoft Visual Studio 2005, Microsoft SQL Server 2000, Flash CS3, Dreamweaver, Photoshop and Flex 3.0, and programmed using ASP.NET(C#), T-SQL, ActionScript 3.0 as well as HTML/CSS/JavaScript and XML web services.

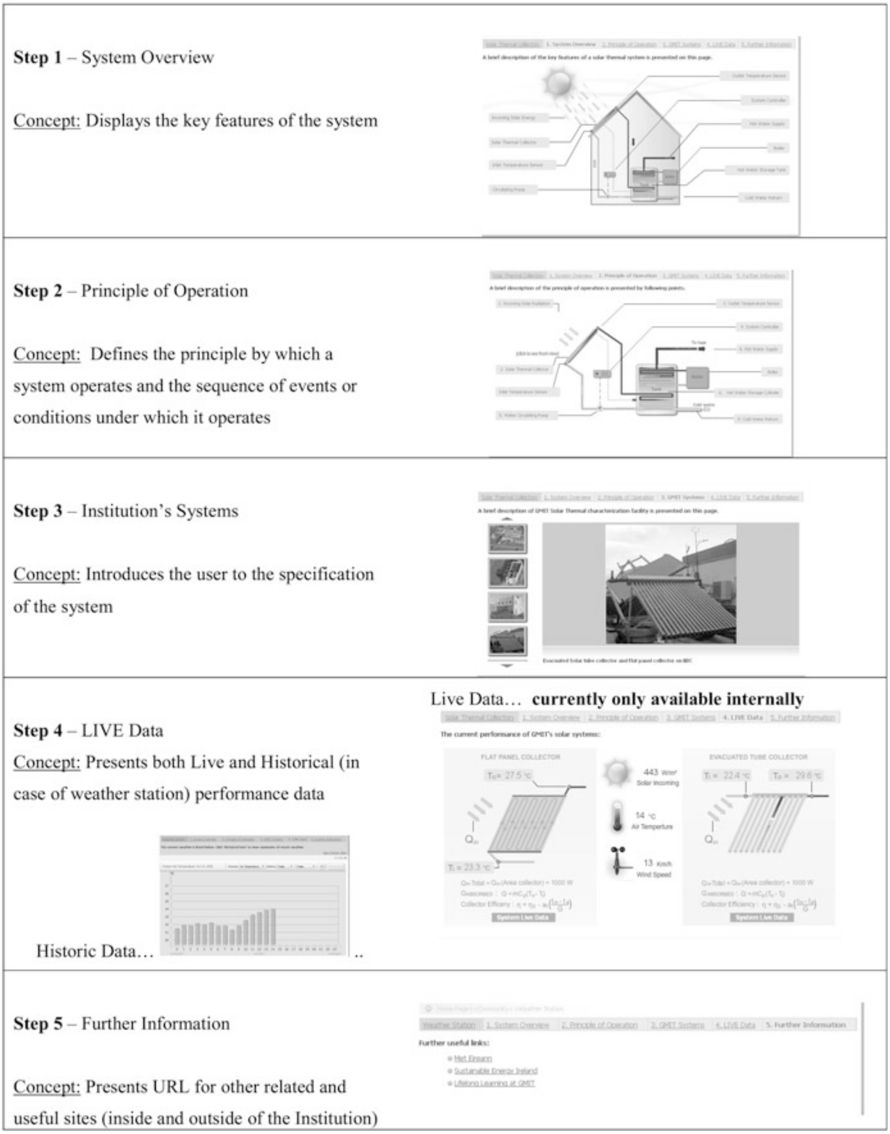


Fig. 11.4 Five-step e-learning methodology employed in the KDP

6 Discussion

The concept of establishing a knowledge discovery portal (KDP) is a key and distinctive element to enable the institution to fulfil the research strategy articulated in the strategic plan (2010–2014) to 'integrate research, teaching and regional development'. KDPs aim to encourage the transfer and sharing of knowledge and research outcomes to all learners and end users so as to maximise the returns to education, the economy and society. They will foster the vision of promoting knowledge discovery and innovation which will allow the institution to deliver new learning opportunities for the knowledge society. This dovetails perfectly with the institution's teaching and learning remit and facilitates the integration of research, teaching and regional development.

The knowledge discovery portal will place research, including the personnel, project or initiative, motivation, methodology and findings, on a more accessible platform within the wider community. This should help to increase awareness of the knowledge creation process as well as maximise the economic and societal impact derived from ongoing research by promoting the exchange of ideas between the education (including primary and secondary level) and the research community, policy makers, private enterprise and societal stakeholders. The success of this initiative depends heavily on the ability of researchers and the institute to build in a new level of access, communication and dissemination about research so that new knowledge and methodologies can be disseminated effectively.

Accessibility will be provided by ensuring both physical access to visit research facilities, meet researchers and view animations, demonstrations, laboratories and posters as well as remote access through interactive web-based models, information sites and virtual laboratories.

Effective knowledge discovery platforms would help to excite younger generations about possible research careers; promote ongoing research and aid the development of the knowledge society regionally, nationally and internationally; and stimulate interaction with external stakeholders to generate new research ideas, while also helping to attract further funding to help sustain future research initiatives.

This concept embraces the traditional modes of disseminating research findings such as publication and presentation but also builds a new level as accessibility to the research environment whether technology enabled via the internet or physically via demonstration or site visits. We will now examine the work using the guidelines proposed by Hevner et al. which were outlined in Table 11.2.

Guideline 1: Design as an Artefact: Hevner et al. propose that the result of design-science research must produce an IT artefact that addresses an important problem. The prototype KDP is an artefact, albeit an early version of a longer-term project, that meets this criterion.

Guideline 2: Problem Relevance: It goes without saying that the issues of sustainability and energy sources are a pressing problem for not alone business (p. 84) and industry but society. The KDP is an attempt to use information systems to contribute not only to the debate but to assist those who wish to implement solutions to this global problem.

Guideline 3: Design Evaluation: Hevner et al. point out (p. 85) that the technical infrastructure is incrementally built and the present KDB artefact is a step on the complete research journey. They provide five design evaluation methods for the designed artefacts (p. 86). We believe that our study meets two of these evaluation methods. It meets method 1 – *observational* – as this is part of a case study of the artefact in an organisational environment. Also, it meets method 5 – *descriptive* – as detailed scenarios have been implemented to test the KDP in a simulated environment.

Guideline 4: Research Contributions: We believe that the KDP is a novel contribution to an emerging area in the provision of live data from renewable energy systems and from energy management systems.

Guideline 5: Research Rigour: The authors have striven to meet both rigour and relevance in the development of the artefact which have used established information systems development techniques.

Guideline 6: Design as a Search Process: Appendix 1 provides a detailed summary of the search process that was employed to measure the KDP against initiatives in this area.

Guideline 7: Communication of Research: The KDP has been extensively demonstrated to energy professionals, IT developers and member of the public interested in this area. This chapter is the first attempt to bring the work to the attention of the IS community.

We argue that this chapter makes a contribution to both research and practice. Some similar initiatives have been introduced in the educational area (Harvard 2010). However, this chapter integrates information both in the horizontal and vertical axes. In the horizontal plane, it provides information to community users, educational bodies and industrial companies. In the vertical plane, it allows deeper access depending on the requirements of the user: from technological overviews to detailed data from the energy sensors. This chapter is limited in that it is the first phase of a more detailed project, articulated in Fig. 11.1 that will require more extensive evaluation.

7 Conclusions

This chapter makes a contribution to the emerging debate in the information systems discipline concerning sustainability. It outlines the initial prototype development of an energy informatics portal that will integrate both RES (renewable energy sources) and RUE (rational use of energy) data. The approach taken to this chapter was based on design-science methodology. It is constructed from a real-life case that is based in the educational sector but had the objective of reaching out to both industrial and community sectors.

A number of advantages are foreseen by this chapter that includes:

- Branding the region as environmentally aware (through a virtual 'green' network)

- Supporting the development of sustainable energy zones and sustainable communities
- Supporting product development in local 'green energy' industries

Future work will involve further developing the KDP based on four goals:

1. Evaluate the aesthetic design of the current portal to enhance appeal and navigation (Task 1)
2. Refine the existing web-portal architecture to increase performance and functionality (Task 2)
3. Develop engaging GUIs for each of the 4 remaining energy systems (Task 3)
4. Profile the campus' E-Aware energy monitoring project (Task 4)

Also, there is room, we argue, to use this study to contribute to the philosophical debates on energy and sustainability. For example, the work of Jacques Maritain on the concept of the 'common good' could provide a fruitful basis for this discussion (Maritain 2002). The development of the KDP provides an exemplar of how ISD can be used for eco-sustainability from the knowledge creation, dissemination and discovery perspectives. Watson et al. (2010, p. 24) call for action by IS scholars in incorporating eco-sustainability as an 'underlying foundation in their teaching'. We hope that this chapter helps to stimulate debate on how the ISD discipline can contribute to the important sustainability agenda by generating, disseminating and using energy for both economic and social advantage.

Appendix 1. Summary of e-Learning Initiatives Deployed by Educational Institutes to Promote Sustainable Energy Systems (SES)

#	Researcher/Institute/Url	Technologies	Goals of Site
1	Lund et al. (2001) Murdoch University, Australia. www.rise.org.au. Accessed Nov 2010	Live data, sustainable energy systems, energy awareness and promotion, labs	"To test, research, promote, educate and demonstrate RES in order to support Australia's sustainable energy Industry"
2	Queens University, Canada. http://livebuilding.queensu.ca/. Accessed Nov 2010	Live data, energy awareness and promotion, sustainable energy systems, green features, labs	"To create a 'live building' to be used as a lab and to be made available to researchers, students and the public to further advance engineering theories and ideas"
3	Willis College, Canada. http://toronto-centre.williscollege.com/	Sustainable energy courses	"To offer flexibility to students wishing to study online and for individuals from anywhere in the world"

(continued)

#	Researcher/Institute/Url	Technologies	Goals of Site
	wind_energy.php. Accessed Nov 2010		
4	University of Vermont, USA. http://buildingdashboard.com/clients/uvm/davis/. Accessed Nov 2010	Energy awareness, sustainable energy systems, Live data, green features	"To hold the college community accountable for their environment and to display to the college community and visitors the energy flows, water and materials used to create a more ecological conscious place"
5	Oberlin College, USA. http://www.oberlin.edu/ajlc/ajlcHome.html. Accessed Nov 2010	Energy awareness, sustainable energy systems, Live data, green features	"To be used as a core area of the students environmental studies and to share this information with other students, educators, activists, architects or other interested parties in creating a sustainable relationship among humans, technology and the natural world"
6	Blanchard et al. (2006) CREST, Loughborough University. http://crestdl.lboro.ac.uk/outside/index.php. Accessed Nov 2010	Biomass, hydro-electric power, Solar PV	"To offer flexible and distance learning for study at home or in your workplace in sustainable energy"
7	Moreno et al. (2007), Desire – Net Project, Italy. http://www.desire-net.enea.it/Layout/Desire/index.asp. Accessed Nov 2010	Sustainable energy courses	"E-Learning in sustainable energies for managers, designers and operators in Central, East and South East European and South Mediterranean countries"

References

Accenture (2010) Cloud computing and sustainability: the environmental benefits of moving to the cloud. Green IT Expo, QEII Conference Centre, London, 9 Nov 2010

Appelman JH, Krishnan SS (2010) Efficient and effective – toward a strategy that works forever. INNOV 2010 – international multi-conference on innovative developments in ICT, Athens, Greece, 29–31 July 2010

Boudreau M, Chen A, Huber M (2008) Green IS: building sustainable business practices. In: Watson R (ed) Information systems: a global text. Global Text Project, Atlanta. http://globaltext.terry.uga.edu/

Chen AJW, Boudreau MC, Watson RT (2008) Information systems and ecological sustainability. J Syst Inf Technol 10(3):186–201

DCMNR (2007) Delivering a sustainable energy future for Ireland. 'The energy policy framework 2007–2020'. Government White Paper, Dublin

DETI (2009) Developing the green economy in Ireland. Department of Enterprise, Trade and Innovation, Dublin

Forfas (2010) Adaptation to climate change: issues for business. www.forfas.ie

Harvard (2010) Harvard University building dashboard. http://buildingdashboard.com/clients/harvard/cowperthwaite/. Accessed Nov 2010

Hevner AR, March ST, Park J, Ram S (2004) Design science in information systems research. MIS Q 28(1):75–105

Laitner J, Ehrhardt-Martinez K (2008) Information and communication technologies: the power of productivity: how ICT sectors are driving gains in energy productivity, Report no. E081, American Council for Energy Efficient Economy, February 2008

Maritain J (2002) The person and the common good (trans: Fitzgerald JJ). University of Notre Dame Press, Notre Dame. New Impression edition (October 31, 1973)

Markus ML, Majchrzak A, Gasser L (2002) A design theory for systems that support emergent knowledge processes. MIS Q 26(3):179–213

Porter M, Van Der Linde C (1996) Towards a new conception of the environment-competitiveness relationship. J Econ Perspect 9(4):97–118

Robèrt K-H, Schmidt-Bleek B, de Aloisi LJ, Basile G, Jansen JL, Kuehrg R, Price TP, Suzukig M, Hawkeni P, Wackernagelj M (2002) Strategic sustainable development – selection, design and synergies of applied tools. J Cleaner Prod 10(3):197–214

Sobotta AT, Sobotta IN, Gøtze J (2009) Greening IT: how Greener IT can form a solid base for a low-carbon society. Creative commons attribution non-commercial share alike license v3.0. http://creativecommons.org/licenses/by-nc-sa/3.0/legalcode

Venable JR (2010) Design Science Research Post Hevner et al. Criteria, standards, guidelines, and expectations. In: Winter R, Zhao JL, Aier S (eds) Global perspectives on design science research, 5th international conference, DESRIST 2010, St. Gallen, Switzerland, 4–5 June 2010. Lecture notes in computer science, vol 6105. Springer, Heidelberg, pp 109–123

Vezzoli C, Manzini E (2008) Design for environmental sustainability. Springer, Berlin/London

Walls JG, Widmeyer G, Sawy OE (1992) Building information system design theory for vigilant EIS. Inf Syst Res 3(1):36–59

Watson RT, Boudreau M-C, Chen AJ (2010) Information systems and environmentally sustainable development: energy informatics and new directions for the IS community. MIS Q 34 (1):23–38

Webster J (2009) A call for action in tackling environmental sustainability through green information technologies and systems. Panel discussion at ECIS 2009 conference, 8 June 2009

Chapter 12
Product or Service? An Interpretive Case Study of Web Development

Michelle Tellam Lawrence and Briony J. Oates

1 Introduction

Information systems development (ISD) should be seen as a socio-technical process because:

- It takes place within an organisational context, meaning that *technical* methods and artefacts are influenced by *social* organisational structures, cultures and work practices (Checkland 1981; Checkland and Howell 1998; Dittrich et al. 2008).
- The *technical* methods and artefacts are also the outcome of processes of *social* negotiation and meaning construction between the people involved, so that shared understandings become embedded in the development process or product as hidden assumptions and belief systems (Floyd et al. 1992; Suchman 2007).

Web development is one type of ISD. Kappel et al. (2005) argue that web applications differ from traditional, non-web-based applications along four dimensions: product, usage, development and evolution. The product-related characteristics include multimedia content, non-linear hypertext-based structures and the need for self-explanatory presentation designs. Usage-related characteristics include multiculturalism, multi-platform delivery and 24/7 availability. Development-related characteristics include multi-disciplinarity (e.g. teams including both marketing experts and IT experts) and short development times. Finally the evolution dimension emphasises the continuous change of requirements and conditions in most commercial contexts and the short lifespan of many web applications.

M.T. Lawrence (✉)
Cornwall College Camborne, Trevenson Road, Pool, Redruth, Cornwall, UK

B.J. Oates
School of Computing, Teesside University, Middlesbrough, UK

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_12,

In this chapter, we present a case study of web development in one particular organisation, using qualitative data. Qualitative research enables an understanding of the social negotiations within an ISD project and the influence of contextual constraints and practices on the method in use (Dittrich et al. 2008). Our research is also based on the interpretive paradigm. An interpretive study does not seek to prove or disprove a hypothesis or establish ‘the truth’, as in positivist research; instead it explores how different people or groups of people perceive their world, and it seeks to understand phenomena through the meanings and value that people assign to them (Oates 2006).

Our case study explores how the social and organisational context influenced the design of the Gem Company’s website and the employees’ differing perceptions about appropriate technical work practices and the resulting effectiveness of the website. This chapter’s contribution is a rich, interpretive case study which adopts a socio-technical perspective to propose an explanation for the differing perceptions and conflicts we observed, which is potentially transferable to other ISD situations.

2 The Case Study

Gem Global is a multinational manufacturer of B2B (business-to-business) industrial electrical products. It acquired an engineering consultancy, which it renamed as Gem Consulting. The purchase meant Gem Global extended its business portfolio by starting to offer consultancy services, and Gem Consultancy gained access to Gem Global’s large network of offices and customers in over 100 countries. As researchers, we were interested in the socio-technical context of web development in Gem Global and in Gem Consulting.

Initial discussions about the employees’ perceptions and practices showed that Gem Global web developers considered the existing company website and the organisational approach to web development had served the company well. Indeed, in 2004, Gem Global had won two significant international web awards for intranet and graduate recruitment design. However, Gem Consulting employees believed that the corporate website was not reaching their potential customers well. We tested this assertion by searching the web for JET (Jewel Extrusion Technology) (a pseudonym), the most strategically significant consulting expertise offered by Gem Consulting, which was experiencing growing demand in emerging manufacturing economies such as China. We typed JET and jewel extrusion technology into a range of search engines. The highest ranking for JET was on page 5 of Google. This link opened a generic PDF document that required scrolling to find any reference to the company or brand. The Gem website had indeed failed to deliver key information to potential customers on Gem’s most important consulting service.

Our research focus was triggered by these conflicting views about the company’s website. Our research question was how can we understand the perceptions and technical practices of web developers in Gem? We investigated the perceptions and practices of two groups of people: those concerned with web development in Gem

Global and those who worked for Gem Consulting. We sought to understand them by examining the socio-technical context of the web developers in each organisation, in which their perceptions and technical practices were situated. The socio-technical context included the history of ISD in the two organisations, the relationship between them and the cultures of the two organisations.

A multi-method approach was used for data collection. Twenty-four one-to-one interviews were conducted with personnel in roles related to web development, information systems support, the management of customer accounts and new product development, as well as senior managers. Most of the interviews lasted approximately 1 h, but six lasted up to 2 h: those with the six personnel whose primary responsibilities involved web design, development and content management. All interviews were taped with the permission of the respondents and then transcribed. We were also granted unlimited access to the corporate intranet site containing company manuals, policy documents and reports. Comments and assertions made by our interviewees were, where possible, corroborated via other data generation methods. For example, where an interview identified a belief about website search engine performance, this was tested through a search with the respondents during the interview, using the search engines suggested by the respondents. Where respondents commented that customers were experiencing problems with the corporate website, similar observations were sought in e-mail exchanges or contact reports. The transcripts of the interviews were uploaded to the qualitative research software package *NVivo*. An inductive approach to data analysis was adopted, where we looked for themes and patterns which could help us indentify how the social and organisational context(s) could have influenced different groups' perceptions and technical practices.

3 Results

This section first presents the perceptions and technical practices of the employees concerned with web development within Gem Global and then presents the perceptions and practices of the employees concerned with web development within Gem Consulting.

3.1 Gem Global Employees' Perspectives

The first UK-hosted Gem website was launched in 1995. It was not developed using formalised methods. The developers involved were those employees who expressed a keen interest in the new web technology. Due to the highly technical nature of the Gem product range, the technical artists and catalogue copywriters tended to populate the new roles emerging in web development.

> There was no planned route to the job I do today. I came from the product side, rather than the IT department. My previous role was making sure that manuals for Gem electrical products were professionally produced and technically correct, which is more of a marketing role really. I sort of evolved into it . . .. . . because I got interested in the Internet at home.
> (Keith, Gem Global UK-based Webmaster)

In the 1990s, there was limited understanding of the web's potential, and it was perceived simply as an electronic advertisement for the various parts of the Gem organisation. From the mid-1990s, most Gem business units developed individual websites without considering the implication to international customers of the combined global interface.

In 2001, Gem Global recognised that there was a proliferation of platforms, cost and styles, and it addressed this ad hoc approach to web development by adopting a web engineering approach, with a common web platform and strongly centralised web management. Since then, Gem Global had invested considerable resources to develop its corporate website. The corporate website aimed to focus on one global brand and drew on widely accepted usability guidelines (Nielsen 2000). Gem Global's intranet site contained clear web policy guidelines for business units, stating that the organisation must place 'the customer first, Gem Global second and the individual business unit third'. The widely understood business and development context for Gem Global was that the organisation sold industrial electrical goods and that service values were added through detailed, accurate technical and after sales information. Since each product article was complex but standardised, the technical information was the same for all world markets (albeit translated). Gem Global insisted that the common web platform be used for all applications except those with specialist functionality that could stand alone only after central management approval. In particular, 'rogue sites' – i.e. separate websites set up by individual Gem units without following the Gem standard – were not allowed. Critical to this view was the control of content management systems, allowing each unit to alter product details, pricing and press information whilst leaving the overall navigation system and architecture fixed, unable to be altered without a long process of requirements analysis.

Subsequently the website's role as a B2B (business-to business) purchasing medium was developed, and then further development work integrated it with legacy systems monitoring manufacturing and logistics. These more recent increases in complexity, involving transactions and integration with management information systems, fostered the view that security and accuracy may be preferable to innovation or flexibility.

> Our website has been developed from the need to create an online catalogue and electronic technical manuals. B2B transactions are becoming very important as is integration with our management systems . . . security is critical . . . the site needs to be robust.
> (Keith, Gem Global UK-based Webmaster)

In our interviews, the Gem Global web team said they had adopted a formalised approach towards the development of complex web systems. Although the term 'web engineering' was not specifically used in interviews, the team members referred to a ‘well-engineered website’, achieved by ‘strict process controls’. Disciplined and systematic approaches were adopted using hierarchical management principles. The development process was largely sequential, and requirements were considered to be relatively static and unchanging. The central web development team required considerable up-front analysis and planning to be ‘signed-off’ before proceeding with any changes to the site. Respondents produced examples of archived questionnaires and reports for web requirements, based upon long-term planning processes. When we tracked the paperwork and electronic records of specific changes, they revealed that even small changes often took months to implement. For instance, for one unit to justify obtaining a specific URL for a promotional newsletter took 7 months.

However, Gem Global web developers believed that their development process was as flexible as possible given the need for security, quality assurance and technical rigour.

> We created a fairly rigid web policy because you wouldn’t believe what some units around the world would put up if left uncontrolled. But if requests are made for changes that are supported by a clear business case, we’ll either support the change or explain why the change cannot happen at the present time.
>
> (Wilhelm, Gem Global Central Web Manager)

Systems managers who implemented the changes to various databases at a global level did reveal that requests for structural changes to the website did not always seem to be handled via the rational process outlined in the web policy documents. To them, it appeared that a successful change may depend on the profile of the business unit or the manager making the request.

> It’s a bit of a black art. If you get the right people on board and maybe if they (the strategic web management team) are not overwhelmed with other projects …. They are very concerned with proliferation of pages, you know, units not meeting standards.
>
> (Jake, Gem Global Systems Technician)

Post acquisition the original Gem Consulting website, which contained service descriptions and contact details, was integrated into the Gem Global corporate site and modified to comply with Gem Global standards. This meant that the engineering consultancy information was placed two levels below the Gem main page. The Gem Global rationale was that engineering consultancy was a subsection of its global services offering. When consultants within Gem argued that potential customers searching for consultancy services were generally looking for a specific solution provider (such as Gem Consulting) and did not want or need to see the Gem Global corporate profile on the homepage nor to search through two levels for Gem Consulting, this was interpreted by the Gem Global web team as the usual business unit complaint about enforced compliance:

> They see their business as unique and the guys on the ground can get quite inflexible.
>
> (Lars, Gem Global Central Web Manager)

There was some recognition within Gem Global that they may not be providing the ideal website for Gem Consultants:

> I don't think we have really looked at consultancy in a web environment.
> (Keith, Gem Global UK-based Webmaster)

However, the blame was placed with Gem Consultants, who were seen as reluctant to contribute to the larger organisation:

> Inevitably, we build what we know and Gem Consulting has not been, shall we say, particularly good at communicating web requirements.
> (Keith, Gem Global UK-based Webmaster)

Next, we examine the perspective of the employees in Gem Consulting.

3.2 Gem Consulting Employees' Perspectives

The core competitive advantage of the Gem electrical product business was the technology of the tangible products, which could be classified, described, photographed and placed hierarchically in an electronic catalogue. In contrast, the core competitive advantage of Gem Consulting was the knowledge in the heads of the consulting team, along with the trust and relationships they developed with customers. Only a small percentage of Gem Consulting revenue came from first orders from new customers. Respondents explained that large revenue streams were built upon successful completion of a small project followed by identification of further opportunities with the same client through continuous contact. They provided an ongoing consultancy service to these clients, which could not be defined in advance. Each consultancy service delivery varied, as did each set of customer needs. Competitive advantage lay in the company's well-designed response to each situational problem.

Respondents in both Gem Global and Gem Consulting considered Gem Consulting to be populated with clever, innovative individuals. Gem Consulting had a history of recruiting first class graduates and supporting access to postgraduate studies. Its employees had been early adopters of information technology, were comfortable dabbling with IT systems and quick to identify improvements relevant to their own way of working. Teams often believed their own version of a database or spreadsheet was better than the existing system, but they were frustrated that they could lose control of them.

> People adapt systems all the time because they feel they have a simpler way of doing things. Management are trying to pull things in line to get us to be a more joined up company.
> (Nicky, Gem Consulting Service Product Team Member)
>
> The minute they see you have developed something that could be of central value, you lose control of it. Responsibility gets relocated to a bigger unit and you lose the chance to develop your knowledge.
> (Paul, Gem Consulting IT System Support Team Member)

The Gem Consulting response to the centralised control demanded by Gem Global was, according to one respondent, to 'work below the radar', developing software systems without informing Gem Global. There were also examples where

Gem Consulting found ways of apparently complying with Gem Global systems ('to keep them off our backs') without implementing the system as designed.

> Capturing customer complaints should be about internal learning not spending hours filling in data fields for some top level management report. We've simplified the Gem Global complaints system and then just do a bit of manual re-keying for the monthly global management report.
>
> (Pat, Gem Consulting Systems Support Team Member)

Gem Consulting had identified four key mechanisms for effective marketing communications: printed direct mail, seminars, speed and presentation standard of bids and a web presence. During the interview process, it became clear that Gem Consulting had a high degree of autonomy and control over three of these communication channels. The company organised seminars around key topics, particularly those driven by legislation or cost reduction initiatives. These were advertised through direct mail to customers and journal placements, producing good leads since their launch. In addition, speedy and high quality bid presentation was coordinated through a bid team. For the fourth, their web presence, control was ceded to Gem Global. Interviewees considered the development of web-based marketing promotion 'a non-starter'. There was a belief that the business unit was seriously disadvantaged by the architecture of the Gem Global website. They considered it extremely difficult to use in terms of finding information about the services offered by Gem Consulting. Most attributed this to a belief that Gem Consulting was very different to Gem Global, i.e. engineering consultancy versus electrical products. Potential customers searching for consultancy services were generally looking for a specific solution provider. That effectively meant expert knowledge based in the business unit, not the corporate entity.

> I'm not sure how useful the internet would be in the future. It's so bad now that I can't see it being useful at all. We've been swallowed by the parent company.
>
> (Nicky, Gem Consulting Service Product Team Member)
>
> As soon as I get information about a company being specialist in something, I do an Internet search for that topic, just to see if they (the company) come up. It's not rocket science that others are doing that in our field of expertise . . .. except they probably wouldn't find us.
>
> (Ally, Gem Consulting Business Development Manager)

Interviews with Gem Consulting employees responsible for service products or new markets revealed a belief that responsibility for the website was held almost entirely at a global level with the 'communications people' or the 'central web people'. There was a common view that global web managers were unwilling to make changes that did not fit with their own top-down vision of the global site hierarchy. No respondent claimed to feel empowered to make significant changes to the global web system.

> Gem Global seem to be imposing an e-commerce model based on products, transactions, deals and discounts. We pay a huge overhead charge to the group but can't seem to get what we want out of it.
>
> (Eric, Gem Consulting CEO)

This perception of disadvantage seemed to be exacerbated by a culture with less than whole-hearted commitment to web-based marketing communication. Our interviews revealed mixed views about the general usefulness of websites as a marketing communication tool.

> Customers don't come and find you electronically for the kinds of things we do. It's not a channel to market that's worth paying a lot of attention to.
> (John, Gem Consulting Service Product Team Manager)

Contrasting views were expressed by respondents who had more day to day experience of emerging markets, particularly East Asian markets.

> Electronic delivery of information goes down well in places like China. Unlike our established UK customers, they're less impressed with just two grey-haired guys in suits.
> (Ally, Gem Consulting Business Development Manager)

These Gem Consulting employees recognised an urgent need for internet-based business transformation and were becoming impatient with the rigid web policy imposed by Gem Global.

> We should develop our own website independently of Gem Global . . . they're so slow that by the time they found out we could argue a case for it!
> (Jim, Gem Consulting JET Engineer)

However, the sceptical view about the relevance of web-based marketing communication dominated, and the Gem Consulting executive team allocated minimal resources to new web-based ideas or even basic maintenance. For example, content management for the Gem Consulting section of the Gem Global site was assigned to a junior member of the business development team as one duty among many.

> To be honest I can't spend much time updating the site because it's only a small part of my job and not what I'm judged on.
> (Emma, Gem Consulting Business Development Executive)

Gem Consulting's lack of clarity about the role of web-based marketing, and its role as 'new boy' within Gem Global, may explain why Gem Consulting members had not felt politically empowered to push for a website which would meet its needs.

At the time of acquisition, Gem Consulting had been in the process of developing a service-based web portal with a greater range of functionality than the Gem Global service had at the time of our study. Developed via an agile methods approach (Beck 2000), with continuous oral and e-mail communication with a small group of customers and focusing on design simplicity, the experimental portal was launched after a short analysis period and continuously improved over time. It included online project management to allow customers to track work in progress and personalise the presentation of data. This was planned to be additional to the interim reports and personal presentations during the life cycle of each consultancy project. It had not yet become a fully functioning part of the Gem Consulting website at the time of acquisition but was regarded by some respondents to have great potential for customer relationship management.

However, this service portal failed to comply with Gem Global standards, some of which appeared ‘petty’ to Gem Consulting respondents. These included such things as the width of the navigation bars, the use of colour, and the size of logo. Gem Consulting respondents argued that the standards reflected corporate convenience and convention rather than customer needs. At the time of our research, the portal was hosted separately with central management knowledge, and there was an uneasy truce between Gem Global and Gem Consulting. Gem Global viewed the portal as a problem site that ultimately must comply with Gem standards, but there was an awareness that the system was of potential use to other consultancy services that Gem Global had also recently acquired. However, no significant new development had occurred with the portal since acquisition.

4 Discussion

A picture emerges of how the social and organisational context of Gem Global and Gem Consulting influenced the differing web development perceptions and technical practices.

Web development at Gem originated in an ad hoc manner prior to 2001, but Gem Global then developed a formalised method. This is in line with the general trends in web development, where a new web engineering discipline is emerging to replace the former ad hoc processes (Ginige and Murgesan 2001; Kappel et al. 2005). The Gem Global web policy grew from the need to create a unified web presence from hundreds of disparate sites with multiple styles and platforms that were costly and inefficient and failing to convey a global branded message.

The employees who had become involved and subsequently controlled web development at Gem Global had a technical copywriting background. This supports previous research showing that web developers have originated from a broader range of career backgrounds than the traditional information technology departments (Fitzgerald et al. 2002; Murugesan and Deshpande 1999; Pressman 2000). Gem Global’s web developers shared an understanding that their products were industrial electrical products. This understanding became embedded in the website when they created deep hierarchical web pages resembling an electronic catalogue enabled for B2B transactions, complete with PDF files of technical manuals and after sales care information, all presented in the corporate style. This provided an efficient information service for customers whilst lowering printing costs for Gem.

Influenced by the business context of its complex international range of standardised products, unity was pursued through the formal development process leading to a rigid web structure that favoured units with business backgrounds closest to the original Gem product portfolio. When Gem Global acquired Gem Consulting, however, the Gem Global developers did not adapt to the changing context of their organisation, i.e. the service-oriented approach of consultants and their customers, and so applied a similar product-oriented view to consultancy services. As other researchers have found (Butler 2003; Eschenfelder 2003; Klein and Hirschheim 2001), the rigid approach to web systems development at Gem Global served the needs of its stable, product-oriented business but disadvantaged its newer consultancy business.

Gem Global employees believed that the corporate website and the organisational approach to web development had served the company well, with external awards for intranet and graduate recruitment web design. On the other hand, Gem Consulting employees believed that the product-oriented approach assumptions embedded in the website meant that their services, and indeed very existence, were buried deep within the site, and key topics such as JET were not easily found by search engines.

Gem Consulting had historically allowed employee autonomy to dominate, with consultants applying their skills to unique client problems and developing their own IT systems as required. When Gem Consulting became part of a much larger organisation, the social and organisational context for IT systems development changed, and the consultants' shared assumptions about autonomous IT work practices were not seen as desirable by the new parent company. The management style of Gem Global favoured a high degree of control. Its web engineering approach had worked well in terms of managing the complexity of the global product portfolio, together with secure transactions and integration with manufacturing and logistics management systems. However, this was not a responsive approach for dealing with an evolving marketing environment, i.e. the need to reach new markets (consultancy) in more innovative ways such as via the web, where speed of delivery in web development needs to be much quicker than previous IT systems development (Kappel et al. 2005). These differing embedded assumptions explain how the term *web policy* had come to mean usability, order, security and control to the Gem Global central web team, but to Gem Consulting employees, it meant bureaucracy, unresponsiveness and inflexibility.

The Gem Global web team shared a well-defined purpose for the website: offering products. Gem Consulting employees believed they were not offering a product but a service but had not developed a shared understanding of how the website could support this. The majority felt that web-based marketing was fairly irrelevant to their services. This led them to abdicate responsibility for the development process, and they contributed little to it. The lack of any real contribution from Gem Consulting also influenced the architecture and content of the Gem website. It was also discovered in our interviews that agreement to structural changes to the website could depend on which business unit or manager made the request. Previous research has shown that business units are often unequally represented during the development of a corporate website (Eschenfelder 2003; Star and Ruhleder 1999). Units with more negotiation power as a result of historical development and location within the corporation tend to achieve better market orientation through the corporate website than other units. Gem Consulting members were either not sufficiently engaged or not sufficiently empowered to achieve a website better oriented to their needs.

In summary, the social and organisational context of both organisations changed when Gem Global acquired Gem Consulting, but neither Gem Global nor Gem Consulting web developers had changed their perceptions and technical practices to suit the new context.

5 Conclusion

An interpretive case study explores shared understandings and meanings and provides a rich description and explanation of a situation which may be unique. Our case study has explored the differing perceptions and practices of web developers in Gem Global and Gem Consulting, suggesting how they were influenced by the social and organisational context and how one set (that of Gem Global) became embedded in the company website. Further work has since been undertaken with Gem Global and Gem Consulting, using action research, to help all those concerned with web development to reach a shared understanding about how the corporate website could best offer consultancy services to the market.

An interpretive study's validity cannot be evaluated by positivism criteria of repeatability and generalisability; instead, plausibility and transferability are the aims (Oates 2006). We hope that our summary of our research approach, our rich description of the two Gem organisations and our evidence based on quotes from the respondents have convinced readers of the plausibility of our explanation concerning the influence of the social and organisational context on the differing perceptions and practices of web developers in the two parts of Gem. Our explanation might be unique to Gem. Transferability of this explanation, as with all interpretive research, can only be judged by the readers – whether our explanation proffered for Gem can be transferred to other web development organisations with which the readers are familiar. Certainly we encourage other ISD researchers and practitioners to recognise that ISD does not occur in a vacuum, so they should also consider the influence of the socio-organisational context.

Much of the conflict between the employees in Gem Global and Gem Consulting can be explained by their differing views about whether the web system was marketing a product or a service. This product-versus-service viewpoint can be taken further. Gem Global web developers appeared to view the corporate website as a product, similar to the company's industrial products marketed to customers, when they spoke in terms of a well-engineered website, achieved by strict process controls. They could reconceptualise the website and their relationship with Gem Consulting (and other Gem business units) to be a service, where *they* now act as consultants, bringing their web expertise to address the problems and needs of *their customers*, i.e. Gem Consulting. Further research could therefore investigate whether conceptualising systems developers as service providers rather than product providers proves useful in understanding other systems development situations.

References

Beck K (2000) Extreme programming explained: embrace change. Addison-Wesley, Reading

Butler T (2003) An institutional perspective on developing and implementing intra-and internet-based information systems. Inf Syst J 13(3):209–231

Checkland P (1981) Systems thinking, systems practice. Wiley, London

Checkland P, Howell S (1998) Information systems, and information systems: making sense of the field. Wiley, New York

Dittrich Y, Rönkkö K, Eriksson J, Hansson C, Lindeberg O (2008) Cooperative method development. Combining qualitative empirical research with method, technique and process improvement. Empir Softw Eng 13(3):231–260

Eschenfelder K (2003) The customer is always right, but whose customer is more important? Conflict and Web site classification schemes. Inf Technol People 16(4):419–439

Fitzgerald B, Russo NL, Stolterman E (2002) Information systems development method in action. McGraw-Hill, London

Floyd C, Züllighoven H, Budde R, Keil-Slawik R (eds) (1992) Software development and reality construction. Springer, Heidelberg

Ginige A, Murgesan S (2001) The essence of web engineering. IEEE Multimed 8(2):22–25

Kappel G, Michlmayr E, Proll B, Reich S, Retschitzegger W (2005) Web engineering – old wine in new bottles? In: Proceedings of the 5th international conference on web engineering, (ICWE 2005). Springer LNCS 3255, Munich, Germany

Klein HK, Hirschheim R (2001) Choosing between competing design ideals in information systems development. Inf Syst Front 3(1):75–90

Murugesan S, Deshpande Y (1999) Web engineering: a new discipline for web-based system development. In: Proceedings of the first international conference on software engineering (ICSE), Workshop on web engineering, University of Western Sydney, Australia

Nielsen J (2000) Designing web usability. New Riders Publishing, Indianapolis

Oates BJ (2006) Researching information systems and computing. Sage, London

Pressman RS (2000) Software engineering: a practitioner's perspective, 5th edn. McGraw-Hill, New York

Star SL, Ruhleder K (1999) Steps towards an ecology of infrastructure: design and access for large information spaces. Inf Syst Res 7(1):111–134

Suchman L (2007) Human-machine reconfigurations. Plans and situated action, 2nd edn. Cambridge University Press, Cambridge

Chapter 13
Balancing the Paradox of Formal and Social Governance in Distributed Agile Development Projects

Esmeralda Thomson and Richard Vidgen

1 Introduction

Distributed software development may offer the prospect of creating enhanced value from IT projects, but it brings its own opportunities and challenges, such as cultural incompatibility and lack of trust, control, communication, and collaboration (i.e., Hildenbrand et al. 2008). One way of addressing these challenges is to apply formal governance mechanisms to safeguard value creation. However, in an agile environment, the imposition of formal mechanisms, such as standards, procedures, policies and top-down command, and control structures, will pull against the need to foster a social process of collaboration, autonomy, and self-organization. The choice between formal governance and social governance is not an either/or – it is a both/and since both can contribute effectively to successful project outcomes. The both/and perspective on governance raises the question of how project managers can deal with the tension between these seemingly contradictory approaches.

The objective of this chapter is to explore the potential benefits and challenges of jointly adopting formal and social governance mechanisms in distributed agile development (DAD) projects. In other words, we want to secure the benefits brought by formal governance mechanisms while retaining the agility of the project team through social governance. Further, we will use paradox theory consider how the project manager can live with and exploit the tension of formal and social governance mechanisms. To this end, we will first explore the domain of distributed agile software development, second review IT and software development governance, third introduce a theory of social governance, and fourth propose a combined model for formal and social governance of DAD projects. This chapter concludes with a summary and ideas for future work.

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_13,

2 Distributed Agile Development (DAD)

The founding principles of agile software development (ASD) are expressed in the Manifesto for Agile Software Development (Agile Manifesto 2001). These principles are made concrete through practices such as users and developers being colocated, short development iterations (typically from 1 week to 1 month), daily stand-up meetings to review progress, and the planning game to prioritize user requirements (Schwaber and Beedle 2002). The project management approach in ASD encourages teamwork, self-organization and accountability, a set of best practices that allow for rapid delivery of high-quality software, and a business approach that aims to align development activity with customer needs. Over the course of the first 10 years of the twenty-first century, ASD has gone from being a controversial practice to business as usual for many firms with various agile methods such as Scrum (Schwaber and Beedle 2002) and Extreme Programming (XP) (Beck 1999) being particularly popular.

At the same time that ASD has become common practice, distributed software development has become a common business reality. However, such increasing movement toward distributed software development and the formation of distributed teams poses a potential dilemma for organizations who have adopted agile methods. Agile and distributed development approaches differ significantly in their key principles. For example, while agile development approaches mainly rely on social processes to facilitate coordination, distributed software development typically relies on formal mechanisms to manage and control the process (Ramesh et al. 2006).

Due to the physical separation of development teams in a distributed environment, many of the key assumptions within agile development such as customer interaction and collocated teams with daily face-to-face communication are not applicable. Indeed, within the context of distributed software development, team members must rely on different technology tools to communicate, and in some cases, they never meet in person. The intense involvement between the developers and between the development team and customers that is typical of agile development projects suffers from the challenge of distance and may suffer further from cultural differences within the team (Hildenbrand et al. 2008).

Taking the above challenges into consideration, one can conjecture that the more distributed the software development project is over time and space then the less compatible distributed development is with agile practices. Rather than add to the loss of agility by imposing formal governance mechanisms on to DAD, we propose that social governance can be used to preserve and maintain agility in DAD. In the next section, we consider formal and social governance mechanisms.

3 IT and Software Development Governance

Weill and Ross (2004) identify effective governance as the single most important predictor of the value that an organization generates from its IT activities. Pye and Warren (2006) find that organizations with effective IT governance programs gained 20 % higher profit margins than those organizations with poor quality governance programs with similar strategic goals.

Complexity in information technology architectures and infrastructures and an increasing need for firms to align their IT organizations with their business to secure value generation processes call for an increasing awareness and understanding of IT governance. IT governance research has investigated the structuring of IT activities and IT decision-making within organizations. For example, Weill and Ross (2004) characterize IT governance as addressing the "what," "who," and "how" of decision-making, and Simonsson and Johnson (2006) refer to IT governance as a decision-making process concerned with goals, processes, people, and technology at both tactical and strategic levels. The research has further considered the contingency elements that may affect governance such as structure, strategy, industry, and size (Bannerman 2009).

Although various definitions of IT governance have been proposed, a widely accepted view is that IT governance is the top management concern of controlling IT's strategic impact and its delivery of value to the business (Weill and Ross 2004). It is about ensuring that IT investments create business value for firms and IT project outcome is enhanced.

Weill and Ross (2004) define other key principles that are essential for businesses to have effective IT governance including involvement of senior managers in committees, decisions, and performance reviews; a champion who owns and is accountable for the IT governance process; and the allocation of adequate resources. While most of these principles have focused on the organizational level, software development governance (SDG) – a relatively new term in the software development literature – extends governance principles from the organization/IT governance level to the software development level (Yaeli and Klinger 2008).

3.1 Software Development Governance (SDG)

SDG covers the management of software and system development projects in development organizations (Bannerman 2009). SDG helps to align the software development activity with business objectives and the strategic requirements of the organization, as defined by enterprise governance (Yaeli and Klinger 2008). Thus, an effective SDG enables the projects to enhance the outcome and create business value.

The goals of SDG are to increase predictability, safeguard value realization from software projects, manage risk and change, and provide clarity and accountability (Tarr et al. 2008). SDG is enacted by defining organizational structures through establishing chains of responsibility, authority, and communication to empower people within a software development organization (Chulani et al. 2008). Further, SDG achieves its goals and objectives by establishing measurement and control mechanisms to enable software developers, project managers, and others within a software development organization to carry out their roles and responsibilities (Jensen 2001).

3.2 Agile IT Project Governance

Qumer (2007) claims that effective agile governance facilitates business value, improved performance, monitoring, and control of large agile software development environments by aligning business goals and agile software development goals. A recent study into governance of agile teams claims that the greater existence of visibility and opportunities to guide and steer the stakeholders in agile projects makes agile project teams easier to govern as compared with traditional teams (Dubinsky and Kruchten 2009). The greater visibility, it is argued, results in the accomplishments of the project team being made explicit and therefore easier for stakeholders to exercise governance of agile project teams. Ambler (2009) says that the self-organizing characteristic of agile team makes individuals within agile project team accountable for the project scope despite being faced with flexible and frequent changes in requirements.

In an agile setting, every individual is accountable for their roles and responsibilities. The customer is responsible for providing the product features and requirements, executive managers are accountable and responsible for agile asset prioritization and procurement, executive management and agile managers are both responsible and accountable for the selection of agile principles and agile infrastructure, and empowered agile managers and agile teams are responsible and accountable for their decision on selection of the agile software development methodology. Empowered agile teams that are involved in most of the decision-making processes are also responsible and accountable for the selection of specific agile process iterations. Finally, agile managers and agile teams are responsible and accountable for the delivery of a valuable quality product to the customer. This view of governance in agile software development is particularly compatible with the principles and characteristics of agile project teams.

Other research into agile governance suggests different tools and mechanisms to address the challenges of software development project. For example, Cheng et al. (2009) in their study of controlling and monitoring agile software development provide a list of key process indicators and interventions that enable software development managers to govern agile software processes successfully. They claim that measuring several aspects of the software development process such as team, people, task, and quality enables software development managers to control, manage, and steer the development process. Therefore, governance mechanisms that enable managers to monitor projects effectively are the key success factors to increasing the business value created (Talby and Dubinsky 2009).

Some researchers have argued for governance approaches that mirror the iterative nature of the software development process in an agile project environment claiming that, in order to improve the quality of software products, governance mechanisms need to be performed in iteration. For example, Talby and Dubinsky (2009) suggest that implementing governance iterations matched to the short development iterations provides a basis for continuous reflection on the key issues in development projects.

4 A Framework for DAD Governance

The literature studies and case reports are scarce in investigating the effective governance of software projects in distributed environments. Recently, the Carnegie Mellon Software Engineering Institute (SEI) (Anderson and Carney 2009) proposed a Distributed Project Governance Assessment (DPGA) process, taking external dependencies of a distributed software project into consideration to manage the project risks involved. The study argues that in today's world of network and interorganizational relationships, external dependencies can threaten development projects, and therefore they need to have an effective governance mechanism to control these risks and enhance the project outcome.

To address DAD challenges such as coordination, communication, culture, and technology, researchers propose that firms must combine the flexibility offered by the growing agile development approach with the rigidity offered by the traditional plan-based approach; thus, they attempt to combine those two into a common distributed agile development (DAD) model (Ågerfalk and Fitzgerald 2006; Ramesh et al. 2006). Providing such an agile-rigid environment can help the development organizations mitigate different risks inherent in distributed software development projects (Yadav et al. 2007).

The agile-rigid approach indicates that there is a role for both formal and social modes of governance. There is a role to play for formal governance mechanisms in agile projects and in DAD projects in particular where developers and customers are separated by time, space, and culture. However, an overemphasis of formal governance mechanisms may lead to loss of agility in DAD, where collaboration, autonomy, and self-organization are key attributes. In DAD, formal governance mechanisms can be supplemented through the recognition of social governance mechanisms that reflect the bottom-up, emergent nature of agile development. To develop an agile-rigid model of DAD governance, we consider formal governance, then theorize social governance through network governance theory, and then pull the formal and social dimensions together in a research model.

4.1 Formal Governance Mechanisms

There is intense interest in creating business value from IT project investments through effective governance. One of the key issues which will impact on perceptions of the value of IT project investments is implementing effective governance mechanisms to monitor and control the delivery of business value (Mckay et al. 2003). In other words, implementation of formal governance mechanisms such as policies and procedures, planning, structure and hierarchy, and performance measurement, where there is a direct involvement of management in IT projects whenever required and a set of principles to be followed by all organizational members, can move an organization toward achieving better outcomes with their IT.

Mckay et al. (2003) argue that combination of planning, evaluation to assess performance, and benefits management is a vital and central part of effective IT governance, as each of these components adopts a somewhat different focus on the other. Planning based on business requirements and then subjecting those investments to on-going evaluations linked to proactive benefits realization process will ensure appropriate investments and governance of IT project investment to deliver business benefits. This way of evaluating and realizing the benefits is often called benefits management. Without an effective benefits management process, IT benefits may be unrealized, leaving the investing organization without satisfactory payoff.

There are two main modes of formal governance mechanisms; organizations can be governed either by measuring the behaviors based on managers' direct, personal observation and checking of the behavior of subordinates to ensure that the process is appropriate or measuring outcomes of these behaviors only that rely on an accurate and reliable assessment of members' performance by monitoring performance (Das and Teng 2001).

Formal governance mechanisms are required to mitigate the challenges involved between distributed project teams (i.e., team attitudes about each other, norms, and culture differences), and social mechanisms and collaboration are more suitable to resolve the challenges and conflicts within project teams (i.e., problems of accountability and ambiguities in responsibilities). When there is a conflict within project team, the team members are in better position to deal with it through collaboration as they are involved in the day-to-day project team's activities, but when the conflict is between two distributed project teams, the formal policies and standards can resolve the conflict easier (Winkler 2006).

4.2 Social Governance Mechanisms

To explore social governance, we draw on network governance (NG) theory, which is "characterized by informal social systems rather than bureaucratic structures within firms, and formal contractual relationships between them" (Jones et al. 1997, p. 911). NG theory uses informal social mechanisms to coordinate and safeguard interactions between project teams and provides a theoretical framework for examining how the efforts of DAD project teams can be coordinated and safeguarded effectively. There are loose ties between project teams in the network to govern transactions, and this loose structure allows networked project teams to adapt to environmental changes and coordinate actions, as well as safeguard exchanges against the hazards of opportunism.

There are four preconditions for network governance: demand uncertainty, customized (asset-specific) exchanges, complex tasks executed under time pressure, and frequent exchanges between partners. Under such conditions, networks develop structural embeddedness (the extent to which a "dyad's mutual contacts are connected to one another" Granovetter 1992, p. 35), creating both direct and indirect ties between parties and increasing the visibility of the parties' actions. The presence of high levels of structural embeddedness and visibility of action enables the use of various social mechanisms that resolve exchange problems by coordinating and safeguarding exchanges within networks (Jones et al. 1997). This study refers to exchange, in the meaning of the exchange of information, knowledge, and any other sort of exchange that occur as result of agile project team interactions.

We propose that DAD project teams can be conceptualized as using social mechanisms to coordinate actions and safeguard exchanges. The reason is that the essential characteristics of social structures and individual participation in creating a software product in a complex and uncertain environment are present in DAD project teams. Furthermore, we argue that DAD project teams, through their mode of organization and values, should, if applied faithfully, lead to the structural embeddedness that is necessary to establish a network governance mechanism.

4.3 Coordination and Safeguarding Enhance Project Outcome

Jones et al. (1997) identify four social governance mechanisms: restricted access, macroculture, collective sanctions, and reputation. DAD project teams restrict access to certain business representatives that have been agreed to be point of contact for the duration of the project. This results in more frequent interaction between project IT developers and business representatives, which reduces coordination costs and safeguarding as relationships are established and trusted over time between parties. Macroculture indicates that DAD project teams internalize the principles of agility (i.e., principle of business value creation) within the agile project teams. Safeguarding suggests that there is something to lose – reputation – and a price to pay if norms are not adhered to – collective sanctions. These apply to both project IT developers and business representatives, as there is a chance of opportunistic behaviors and not being faithful for both.

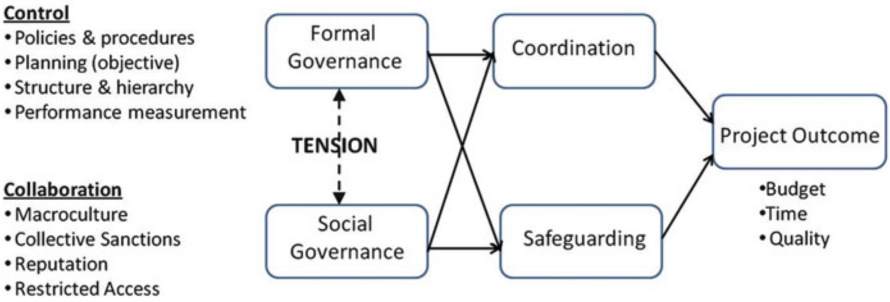


Fig. 13.1 Balancing tension between formal and social governance in DAD projects (Adapted from Jones et al. 1997)

Formal governance mechanisms also facilitate complex and customized exchanges. The customized and complex tasks and knowledge such as development projects increase the need for coordinating and safeguarding exchanges among parties and enhance frequency of interaction. In development environment such as DAD projects, commitments to behave in a responsible manner that are not backed by contract or other credible collateral are assumed unlikely to be always carried out properly. Thus, organizations need to have the ability to exercise controls and safeguard opportunism through formal governance structure (Williamson 1994).

Therefore, formal governance mechanisms such as policies and procedures, planning, central decision-making, structured coordination, and direct management participation (Peterson 2001) also facilitate coordination and safeguarding of exchanges among DAD project teams.

Increased coordination prevents duplication of work, allowing better coordination of tasks among network members, and allows different modules of a complex software development to work together. This reduces time and cost of monitoring and managing resources in project (Williamson 1994). Better coordination also allows project members to spend more time developing features and making the product more useful and feature-rich, which enhances the quality of project outcome. Finally, when project teams work more effectively and are highly coordinated, the project outcome will be enhanced.

Williamson (1994) also suggests that governance mechanisms are required to mitigate opportunistic behaviors and actions and to force compliance through organizational rules, policies, and sanctions. When opportunistic behaviors are controlled by safeguarding exchanges in project, this enhances the project outcome.

In summary, we propose a governance model for DAD projects where formal social mechanisms in line with social governance mechanisms facilitate coordination and safeguarding of exchanges that leads to enhancing project outcome. The theoretical model is summarized in Fig. 13.1.

5 Balancing the Tension of the Formal and the Social

We have argued that DAD projects can benefit from both formal and social modes of governance and that both can help IT projects to coordinate their activities and safeguard their exchanges in order to create business value for customers by enhancing project outcome. This coexistence of formal and social mechanisms in the system raises a paradox to govern the DAD projects. Here, we are facing a tension between proponents of control of formal governance versus collaborative approaches of social governance. The debate is that how executive and knowledgeable board members dominate and control the project, and at the same time, the board allows empowered project teams to manage themselves through collaboration. In other words, how can empowering, collaborative, bottom-up social mechanisms work with contract-based, hierarchical, and top-down formal mechanisms? When do formal and social governance mechanisms work in a complementary way and when they are in conflict?

One way of addressing this seeming conflict lies in paradox theory. We found paradox theory as the most suitable one to explain such tension for the purpose of this study because it explains well the need for the coexistence of formal and social governance and how the tension between them needs to be balanced. Paradox is defined as "the dynamic tensions of juxtaposed opposites" by Rosen (1994, p. 17). Sundaramurthy and Lewis (2003) claim that while the environmental uncertainty is increasing, a more paradoxical approach to governance where the control and collaboration approach coexist is demanded.

The authors highlight the critical needs for coexistence of control and collaboration in governance. They argue that a control approach (formal governance) helps bring human limitations under control through discipline, while a collaborative approach (social governance) benefits individuals' desires via cooperation and empowerment. Yet, if one approach becomes overemphasized, it endangers groupthink and causes distrust. From a paradox perspective, however, embracing and balancing both approaches facilitate learning and adoption (Lewis 2000).

In other words, overemphasizing on one polar may initiate defenses and prevent learning. In contrast, managing paradoxes develop understanding and practices that accept and verify tensions (Sundaramurthy and Lewis 2003). Therefore, paradox theory suggests a balanced approach of control and collaboration in governance to elaborate the underlying tensions between formal and social governance and to emphasize the value of monitoring as well as empowerment. This refers to coexistence of authority and democracy, efficiency and creativity, and discipline and empowerment that exist in formal and social governance.

In line with paradox theory, Meyer (2004) also recognized the imperative for governance processes to balance the tension between the need for flexibility and the need for control. Meyer (2004) proposed a "systemic model" of IT governance comprising five organizational systems including culture, structure, internal economy, methods and tools, and metrics and rewards. When integrated together, the five systems create a "holistic and well-coordinated system of influence" that simultaneously empowers and controls (Meyer 2004, p. 31). Although Meyer posited that these organizational systems are interdependent, he did not present a unifying theory or model that integrated the five component parts into a coherent whole. Moreover, Brown (1999) argues that coordination and collaboration mechanisms are required to enable organizations to respond effectively to the challenges posed by an increasingly complex and uncertain environment. Specifically, organizational structures must balance the hierarchical tradeoffs of control versus autonomy.

Wang et al. (2008) argue that a paradoxical view allows one to discover different paradigms, shift perspectives, question problems in totally different ways, and come up with answers that stretch the bounds of current thinking. The authors then suggest that a deeper understanding of how contradictions can be dealt with in general through a balanced and paradoxical view can lead to a better understanding and therefore new ways of dealing with contradictions in agile development projects. Vidgen et al. (2004, p. 12), in their study of paradox of management in information systems development, suggest project managers not only must accept but also have the courage to embrace the paradox of coexistence of being "in control" and "not in control" rather than avoid tension and accept one or other of these poles.

6 Summary and Future Work

In this chapter, we have argued that in DAD, there is a role for both formal and social modes of governance. Further work is needed to understand how these formal and social governance mechanisms can work together to provide enhanced project outcomes. This raises a number of related questions: How do the formal and social mechanisms live together? Under what circumstances are they complementary and when do they work against each other? What tensions arise from the paradox of the formal and social and how do project managers live with this paradox?

This research is exploratory in nature, with the intention of understanding and investigating contemporary phenomena in a real-life context. Therefore, this study will adopt a case study research design guided by Fig. 13.1. This approach will allow for a detailed contextual analysis of DAD projects' social interaction and underlying governance mechanisms that enhances the project outcomes with the aim of building theory for DAD governance.

References

Ågerfalk P, Fitzgerald B (2006) Flexible and distributed software processes: old petunias in new bowls? Commun ACM 49(10):27–35

Agile Manifesto (2001). http://www.agilemanifesto.org/. Last visit Jan 2011

Ambler WS (2009) Scaling agile software development through lean governance. In: Proceeding of ICSE workshop on software development governance (SDG), pp 1–2
Anderson W, Carney D (2009) Distributed project governance assessment (DPGA): contextual, hands-on analysis for project governance across Sovereign boundaries. In: Proceeding of ICSE workshop on SDG, pp 27–28
Bannerman PL (2009) Software development governance: a meta-management perspective. In: Proceeding of ICSE workshop on SDG, pp 3–8
Beck K (1999) Extreme programming explained. Addison Wesley, Reading
Brown CV (1999) Horizontal mechanisms under differing IS organization contexts. MIS Q 23 (3):421–454
Cheng T, Jansen S, Remmers M (2009) Controlling and monitoring agile software development in three Dutch product software companies. In: Proceeding of ICSE workshop on SDG, pp 29–35
Chulani S, Williams C, Yaeli A (2008) Software development governance and its concerns. In: Proceedings of the 1st international workshop on SDG, ACM, pp 3–6
Das TK, Teng B (2001) Trust, control and risk in strategic alliances: an integrated framework. Organ Stud 22(2):251–283
Dubinsky Y, Kruchten Ph (2009) Software development governance ICSE:455–456
Granovetter M (1992) Economic institutions as social constructions: a framework for analysis. Acta Sociol 35(1):3–11
Hildenbrand T, Geisser M, Kude T, Bruch D, Acker T (2008) Agile methodologies for distributed collaborative development of enterprise applications. International conference on complex, intelligent and software intensive systems, pp 540–545
Jensen MC (2001) A theory of the firm: governance, residual claims, and organizational forms. Harvard University Press, Cambridge, pp 237
Jones C, Hesterly WS, Borgatti SP (1997) A general theory of network governance: exchange conditions and social mechanisms. Acad Manage J 22(4):911–945
Lewis M (2000) Exploring paradox: toward a more comprehensive guide. Acad Manage Rev 25 (4):760–776
Mckay J, Marshall P, Smith L (2003) Steps towards effective IT governance: strategic IT planning, evaluation and benefits management. In: PACIS proceedings, pp 956–970
Meyer ND (2004) Systemic IS governance: an introduction. Inf Syst Manage 21(4):23–34
Peterson RR (2001) Configurations and coordination for global information technology governance: complex designs in a transnational European context. In: Proceedings of 34th Hawaii international conference on system sciences, pp 10
Pye G, Warren MJ (2006) Striking a balance between ethics and ICT 3 governance. Aust J Inf Syst 13(2):201–207
Qumer A (2007) Defining an integrated agile governance for large agile software development environments. Springer, Berlin, pp 157–160
Ramesh B, Cao L, Mohan K, Xu P (2006) Can distributed software development be agile? Commun ACM 49(10):40–52
Rosen SM (1994) Science, paradox, and the moebius principle. State University of New York Press, Albany
Schwaber K, Beedle M (2002) Agile software development with SCRUM. Prentice-Hall, Upper Saddle River
Simonsson M, Johnson P (2006) Defining IT governance: a consolidation of literature. Working paper of the Department of Industrial Information and Control Systems. 18th conference on advanced information systems
Sundaramurthy C, Lewis M (2003) Control and collaboration: paradoxes of governance. Acad Manage Rev 28(3):397–415
Talby D, Dubinsky Y (2009) Governance of an agile software project. In: Proceeding of ICSE workshop on SDG, pp 40–45
Tarr P, Williams C, Hailpern B (2008) Toward governance of emergent processes and adaptive organizations. First workshop on SDG, pp 21–24

Vidgen R, Madsen S, Kautz K (2004) Mapping the information system development process. International Federation for Information Processing, pp 157–171

Wang X, O'Conchuir E, Vidgen R (2008) A paradoxical perspective on contradictions in agile software development. In: Proceedings of ECIS, pp 9–11

Weill P, Ross JW (2004) IT governance on one page. CISR WP No. 49 and MIT Sloan WP No. 4516–04

Williamson OE (1994) Visible and invisible governance. Am Econ Rev 84(2):323–326

Winkler I (2006) Network governance between individual and collective goals: qualitative evidence from six networks. J Leadersh Organ Stud 12(3):119

Yadav V, Ayda M, Nath D, Sridhar V (2007) Investigating an 'Agile-Rigid' approach in globally distributed requirements analysis. PACIS proceedings (paper 12)

Yaeli A, Klinger T (2008) Enacting responsibility assignment in software development environments. In: Proceedings of the 1st international workshop on SDG, ACM, pp 7–10

Chapter 14
Human Resource Behaviour Simulation in Business Processes

Hanwen Guo, Ross Brown, and Rune Rasmussen

1 Introduction

At the operational levels of a business, sets of procedures govern the logical and temporal organisations of physical tasks into workflows. Such procedures are implemented as workflow management systems that guide the operations of a business (van der Aalst and van Hee 2002). In many cases, workflow simulation systems (Aalst et al. 2010) can inform the problem of creating optimal workflow models. Firstly, the workflow simulation, in the "as is" phase, identifies, represents and maps daily activities, choices, resources, messages and processes in an organisation as simulation models. Typically, a resource is either a human resource or non-human resource (Russell et al. 2005). The human resource and their behaviour are defined according to their roles and affiliations. With non-human resources, the model defines properties of objects. Then, the workflow simulation, in its "to be" phase, runs these simulation models to generate simulation results. The analysis about simulation results provides people with insight about bottlenecks and weaknesses in processes. In workflow simulation, these simulation models must be very carefully represented and communicated, which ensures that both experts and stakeholders can understand simulation targets and modelling results.

Despite the fact that we understand what workflow simulation can do, ineffective simulation approaches are frequently used. van der Aalst et al. state that we have already failed in the "as is" phase, where mature human resource modelling approaches are missing (van der Aalst et al. 2008). While concerning the "to be" phase, they note that many workflow simulations start the simulation at an abstract state (van der Aalst et al. 2008). This is not suitable for managers to solve concrete

H. Guo (✉) • R. Brown • R. Rasmussen
Mathematical, Information and Physical Sciences (MIPS) Faculty of Science and Technology, Queensland University of Technology, GPO Box 2434 Brisbane, QLD 4000, Australia
e-mail: hanwen.guo@student.qut.edu.au; r.brown@qut.edu.au; r.rasmussen@qut.edu.au

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_14,

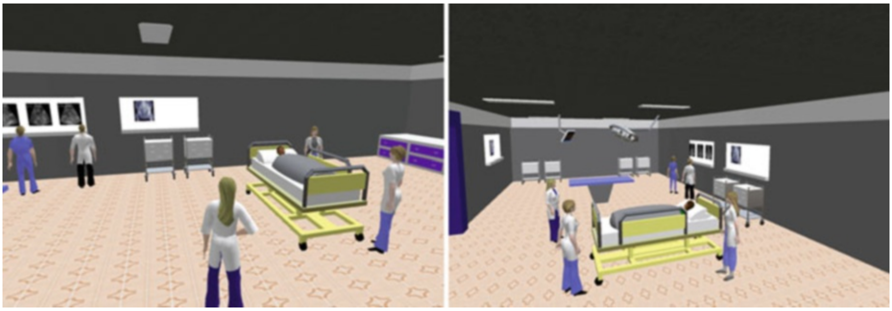


Fig. 14.1 Snapshots of an emergency treatment workflow visualised in a 3D virtual world, where several avatars are about to revive an injured person

problems at the operational level. In addition, we noticed that there is another flaw evident in workflow simulation. The representations of simulations are very useful for experts in workflow, but stakeholders state these representations disrupt their understanding about workflow simulation (Bandara et al. 2007; Sadiq et al. 2007). The major reason for this flaw is that stakeholders are conditioned to real personnel arrangement of physical objects, rather than the abstract representation of models (Bandara et al. 2007; Sadiq et al. 2007; Recker et al. 2007). Thus, the communication of human resource modelling in simulation is still problematic. This introduces risks in transferring resource model knowledge to stakeholders and analysts.

We believe that 3D visual simulation techniques, such as 3D virtual worlds, can provide superior insight into a workflow simulation. A 3D virtual world is a replication of the real world, where every object in it has a 3D representation, and users can interact with these objects. Such a richer environment is better able to demonstrate human work via the clear visual representation of agent behaviours; see Fig. 14.1. Projects have shown that the applications of 3D virtual worlds are very useful for educational and professional training needs in the military (Smith 2006) and healthcare domains (Mantovani et al. 2004); however, only a few researchers (Brown 2010; Brown and Rasmussen 2009) have applied this approach in workflow simulation.

In this chapter, we focus on the human resource behaviour modelling issue to begin the process of developing a comprehensive simulation approach. To our best knowledge, only a few researchers have conducted research on this topic (van der Aalst et al. 2008). We present a human resource centric simulator, which involves a society of intelligent agents that collectively form a multiple agent system. The hierarchical task network (HTN) is an automatic planning algorithm that is used to solve problems by forming and executing sequenced actions (Erol et al. 1994). We employ it to imitate human resources solving assigned workitems. Two rigorous validation techniques (Robert 2007) prove the behaviour of intelligent agents is rational and, therefore, viable as a human behaviour simulation system.

This chapter is organised as follows. Section 2 covers in more detail related simulation approaches and intelligent agent applications. Section 3 introduces our simulator. Section 4 validates and demonstrates our simulator with some simulation scenarios. Section 5 concludes our work and discusses our future work.

2 Related Work

Our work combines aspects of workflow simulation and intelligent agents. The following subsections discuss these two concepts in further details.

2.1 Simulation Approach

Traditional simulations are categorised as simulations that use native resource behaviour modelling, incomplete process modelling approach and less support simulation at the operational level (van der Aalst et al. 2008).

With respect to the resource behaviour modelling aspect, van der Aalst et al. (2008) proposed a model to calculate how an individual human resource allocates working time on tasks. Their model can integrate with Petri-Net based WfMS to estimate the total consuming time of whole business processes.

Modern WfMSs record performed activities in system logs. Many researchers (Agrawal et al. 1998; van der Aalst et al. 2004, 2007) advocate that a process mining tool ProM can be utilised to effectively build process models. Rozinat et al. (2009) proposed a simulation framework to tackle the other two pitfalls. They use ProM to rediscover workflow models from logs and merge these workflow models with current state of the workflow system to generate a near-future scenario.

In this chapter, our agent-based simulation approach focuses on tackling two pitfalls. We designed an HTN-based state transition to model human resource behaviour, driving intelligent agents in the simulation. The behaviour of intelligent agent is recorded in system logs, whereby ProM can be used to rediscover some workflow models in the simulation. Examined simulation models can be provided to assist people in simulated decision making at operational level.

2.2 Intelligent Agent Applications

The term *intelligent agent* can be explained as an autonomous entity that can sense the environment where it locates and acts upon its senses to change the environment (Russell and Norvig 2003). A set of intelligent agents form a multi-agent system (MAS), where they coordinate and cooperate with each other to accomplish a global goal.

O'Brien and Wiegand (1998) advocate that combining MAS with WfMS can enhance workflow management. Researchers (Isern et al. 2010; Decker and Li

1998) advocated that MAS is suitable for modelling organisational structure and solving resource allocation in hospital environments. This is due to the fact that a hospital consists of many individual medical divisions.

In the light of this research (Isern et al. 2010; Decker and Li 1998), an agent-oriented approach should be a viable method for optimising resource utilisation (Peter et al. 2000) in a healthcare application. However, none of them have addressed modelling human resource behaviours in workflow systems. Moreover, there is no work that utilises 3D visualisations as a comprehensive simulation approach. To this end, we aim to establish an agent approach as a critical step towards fully simulating and visualising human resource behaviour using a 3D virtual world approach.

3 Human Resources Centric Simulator

This section discusses details of the human resource centric simulator. Firstly, it defines responsibilities of intelligent agents by analysing simulation requirements. Then, it describes an HTN-based transition mechanism that is used to drive intelligent agents to simulate human resources. Lastly, it describes the implementation of the presented design.

3.1 Simulator Architecture

In daily work, WfMS allocate workitems to resources, provide resources with an interaction platform according to the organisational structure and record the behaviour of resources in logs that can be used for analysis and process modelling. Therefore, we believe that this simulator must satisfy the following requirements:

- Agents should formulate their society, that is, they should replicate the real organisational structure.
- Agents should emulate human resource behaviour according to the responsibilities of human resources, interacting with WfMS.
- Agents should record their behaviour in logs, which is the basis of analysis.
- Agents should be flexible and implemented easily so that they can provide any WfMS with a simulation component, especial a legacy WfMS.

Considering the first three requirements, we identify four types of intelligent agents, namely *resource manager agent*, *resource agent*, *customer agent* and *auditor agent*. Regarding the last requirement, we believe these intelligent agents should be competent to interact with the *workflow reference model* (Hollingsworth 1995), which is a conceptual model that outlines the necessary components in a WfMS. An illustration of the simulation architecture is in Fig. 14.2. A detailed discussion about their social and individual aspects is as follows.

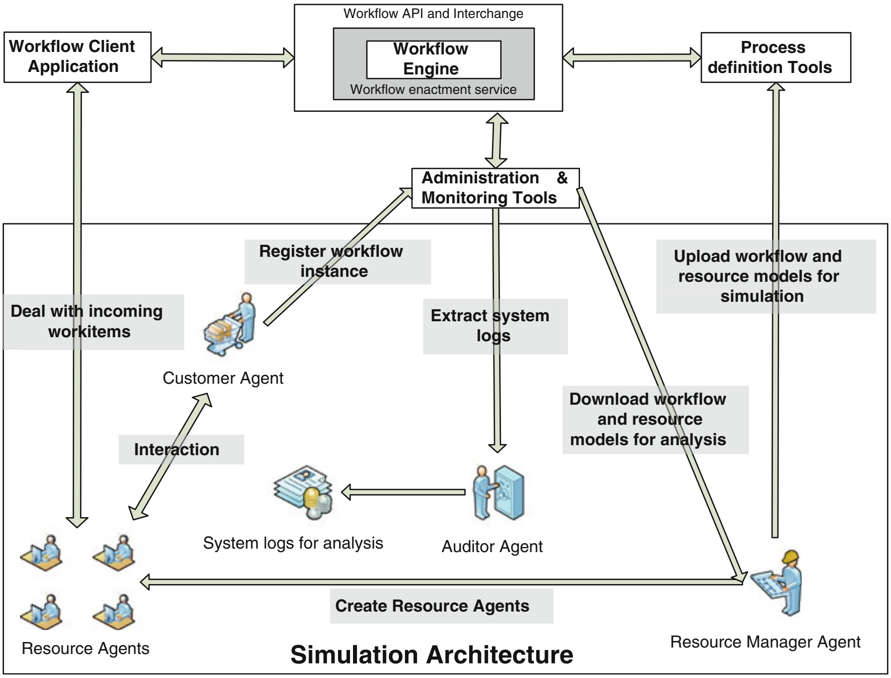


Fig. 14.2 Architecture of our agent-based simulator

The resource manager agent is in charge of replicating the workflow system. This agent downloads workflow and resource models from WfMS via administration and monitoring tools. According to the simulation specification, it will modify workflow and resource models and submit these modified models to the workflow engine via process definition tools. At the same time, it creates a set of another type of resource agent to simulate human behaviour to fulfil the simulated organisational structure.

The resource agent simulates a human resource in the workflow system, processing workitems allocated by the workflow engine and interacting with other agents. When dealing with an incoming workitem, it may require extra information from the customer agent for deciding its behaviour that is driven by a state transition mechanism. Another responsibility of the customer agent is to register workflow instances on the workflow engine according to a simulation specification. As whole workflow instances are accomplished, the auditor agent extracts logs, providing human simulation operators with analysis information.

3.2 HTN-Based State Transition Mechanism

In a workflow management system, behaviour of resources can trigger a state transition of a workitem. Russell et al. (2005) concluded that the state transition

```
Part 1 Decomposition
1.  Input a workitem w.
2.  IF (D(w) returns a primitive task set){
        Find the a sequence of actions σ in A;
        IF (preconditions of σ are satisfied)
            Go to Step 6;
        ELSE
            Return failure;
    }
3.  Choose a non-primitive task t in D(w), whose preconditions are satisfied.
4.  Let w = t
5.  Go to Step 2.
-----------------------------------------------------------------------------
Part 2 Execution
6.  For (each action a in σ){
        Execute the action a and get the execution result;
        Upload the result to workflow management system;
        IF( Upload is successful )
            Go to next turn;
    }
```

Fig. 14.3 Procedure of the adapted HTN transition mechanism

of a workitem is in a finite state set $\boldsymbol{\Omega}$, including *created*, *offered*, *allocated*, *started*, *suspended*, *failed* and *completed*. These workflow transitions are known as *resource patterns*, illustrating the behaviour of resources in a workflow system. In this section, we discuss a hierarchical task network (HTN)-based state transition mechanism used to simulate the behaviour of human resources for triggering workitem state transitions on set $\boldsymbol{\Omega}$.

We adapted a formal representation of HTN in Erol et al. (1994) and define it as a function $HTN = <\boldsymbol{w}, \boldsymbol{i}, \boldsymbol{D}(\boldsymbol{w}), \boldsymbol{A}>$ that returns a sequence of ground *actions* representing a plan for a workitem $\boldsymbol{w}$ that is going to be solved, where $\boldsymbol{i}$ is the initial state of $\boldsymbol{w}$, $\boldsymbol{D}(\boldsymbol{w})$ is function that iteratively decompose the workitem $\boldsymbol{w}$ as a set of ground primitive subtasks and $\boldsymbol{A}$ is the corresponding *action* set related to the decomposition of workitem $\boldsymbol{w}$. The execution of an action $a \in \boldsymbol{A}$, if the preconditions p of action a are satisfied, will finish a subtask and generate some effects e that may lead to a state transition of the workitem $\boldsymbol{w}$. The execution of these sequenced ground actions or plan will lead to its final state that is usually failed or completed in $\boldsymbol{\Omega}$.

We describe a brief HTN in Fig. 14.3. A resource agent will use the HTN to decompose and execute a workitem. Two assumptions have to be made to prevent a problem from becoming a none solution problem (Pollock 1999). First, every primitive task in $\boldsymbol{D}(\boldsymbol{w})$ can find an action in A to resolve, always. Second, a workitem can be decomposed as a set of primitive tasks in finite iterations of the decomposition.

We use a workitem "receive a new patient" to illustrate our HTN-based transition mechanism, see Fig. 14.4, where the workitem is decomposed as three primitive task sets **a**, **b** and **c**. The resolving of any primitive task set **a**, **b** or **c** will trigger the state transition of the workitem on set $\boldsymbol{\Omega}$. For example, in primitive task set **c**, "accept this work" indicates the *offered* state, "check the condition of this patient"

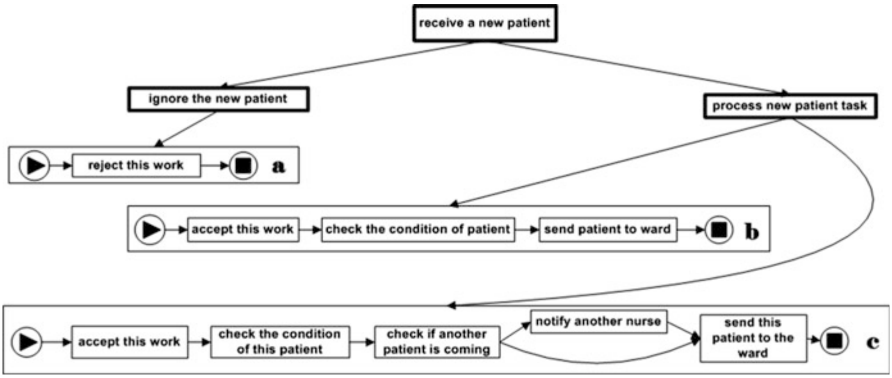


Fig. 14.4 Graphical representation of decomposing the workitem "receive a new patient"

and "check if another patient is coming" indicate this workitem is processing now, that is, it is in *started* state. If the preconditions of "notify another nurse" are satisfied, the resource agent may execute "notify another nurse", and then the current workitem "receive a new patient" may be in the *suspended* state. Lastly, "send this patient to ward" transitions this workitem to the *completed* state.

3.3 Simulator Implementation

We use the JADE platform to implement our simulator. JADE is a JAVA-based agent platform, facilitating agent application development by providing developers with an agent system infrastructure platform. In practice, our simulator will interact with the YAWL system (van der Aalst and Hofstede 2005). The YAWL system is a WfMS developed from the workflow reference model, which facilitates the implementation of a well-structured simulation architecture.

4 Experiments

In this section, we discuss the validation methodology for and the validation results of our simulator. Then, we demonstrate the simulation ability of our simulator in a what-if scenario, which is intended to assist people in operational decision making.

4.1 Validation Methodology and Results

An agent system is modelled from both social and individual perspectives (Wooldridge 2002). Thus, we select two suitable techniques, which are

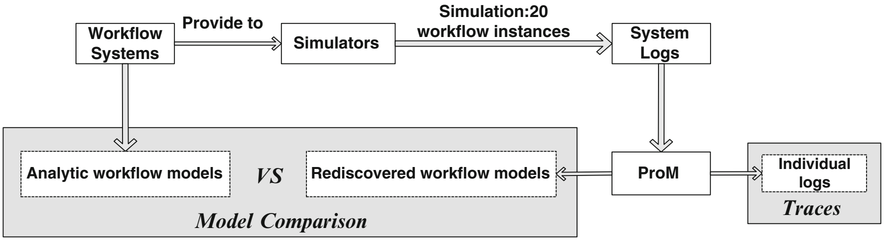


Fig. 14.5 Illustration of the validation framework. Simulators are provided with a number of workflow instances. System logs are extracted to rediscover models which are used for model comparison and traces

"comparison to other models" and "traces" from Robert (2007) to validate our simulator, aiming to provide evidence that the social and individual behaviour of the agents in this simulator is rational. We rename comparison to other models to *model comparison* in the following paragraphs.

On the one hand, model comparison requires us to compare our simulation results with the other simulation results. In simulation, any result is the joint effort of every intelligent agent in the simulator. Thus, this technique validates the social aspect. On the other hand, trace requires us to follow and analyse the behaviour of an individual intelligent agent in the simulator and determine if the logic of the target is correct, that is, it validates the individual aspect.

We additionally create a random simulator to facilitate these two techniques. The random simulator replaces the resource agent with the random agent but still uses the architecture of a human resource centric simulator. The random agent will randomly select an action to transit a workitem from one state to another, without considering the current situation.

In practice, two simulators were validated in three adapted workflow systems (Biffl et al. 2001; Nicodemo et al. 2008; Croce et al. 2007), processing 20 workflow instances, respectively. The structure correctness of these workflow models in workflow systems is considered, since rediscovered ones are only compared with structure similarities, rather than medical treatment purposes. Logs of two simulators were validated through the framework in Fig. 14.5.

In trace, abnormal state transitions are identified in the random simulator. We selected two workitems, *receive patient* and *transfusion*, for demonstration; see Fig. 14.6. Although random agents can select actions to let a workitem transit from one state to another, they ignore what is a suitable action for the transition. Compared with random agents, resource agents can rationally select actions from knowledge.

In model comparison, workflow models rediscovered from the human resource centric simulator's logs are similar to analytic workflow models; however, significant discrepancies are found between the workflow models rediscovered from random simulator's logs and analytic workflow models. We select one adapted workflow system (Biffl et al. 2001) for demonstration; see Fig. 14.7. The model *A* is

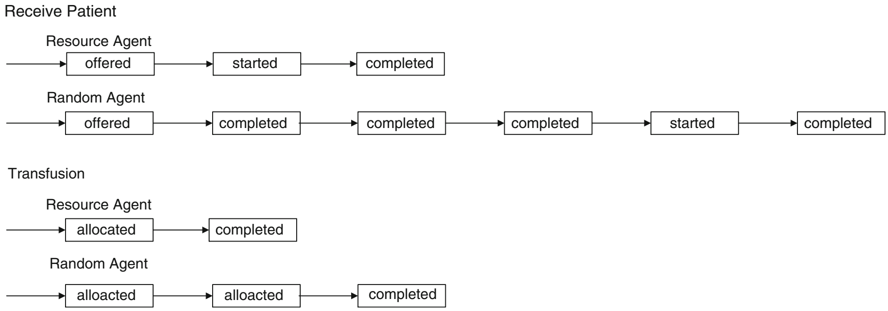


Fig. 14.6 Illustration of a fragment in systems logs, which represents a potential state transition sequence. It is apparent that the random agent cannot make a rational action in workitems where redundant states appear. Compared with random agent, there is no redundant state transition in the trace of resource agent

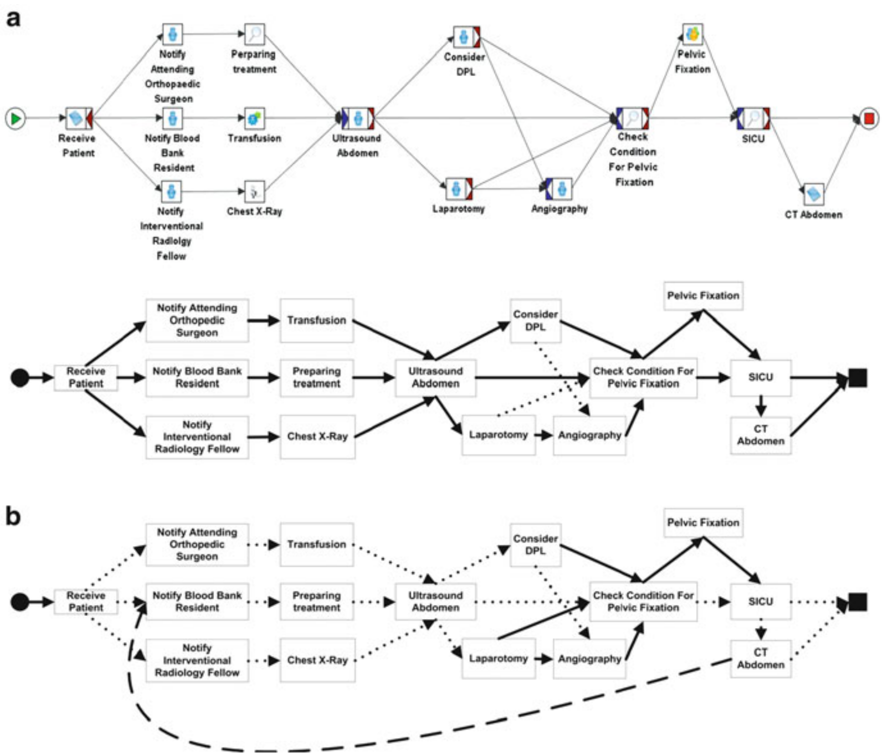


Fig. 14.7 Analytic model *O* and two rediscovered workflow models (*A* and *B*). Model *A* was rediscovered from the human resource centric simulator's logs, while model *B* was rediscovered from the random simulator's logs. *Dashed lines* indicate missing transitions and wrong transition

Table 14.1 Three candidate resource models

Resource model	Resource ID	Responsibility
A	staff_1	Laparotomy, check condition for pelvic fixation, pelvic fixation
	staff_2	Transfusion
	staff_3	CT abdomen, angiography, chest X-ray, consider DPL
	staff_4 staff_5 staff_6	Receive patient, preparing treatment, SICU, notify orthopaedic surgeon, notify blood bank resident, notify interventional radiology fellow, ultrasound abdomen
B	staff_1	Prepare treatment, laparotomy, check condition for pelvic fixation, pelvic fixation,
	staff_2	Transfusion
	staff_3	CT abdomen, angiography, chest X-ray, consider DPL, ultrasound abdomen
	staff_4 staff_5 staff_6	Receive patient, SICU, notify orthopaedic surgeon, notify blood bank resident, notify interventional radiology fellow
C	staff_1	Laparotomy, check condition for pelvic fixation, pelvic fixation,
	staff_2	Transfusion
	staff_3 staff_7	CT abdomen, angiography, chest X-ray, consider DPL, ultrasound abdomen
	staff_4 staff_5 staff_6 staff_8	Receive patient, SICU, notify orthopaedic surgeon, notify blood bank resident, notify interventional radiology fellow

similar with model *O*, only two transitions are missing. Considering the model *B*, we can tell that most of the transition is missing and a wrong transition is created.

With the investigation in model comparison and trace, we believe the results provide supporting evidence that the human resource centric simulator is rational, and it can be used as a basis for a tool to assist people in decision making.

4.2 Resource Agents Workload Analysis

With the positive result in the previous section, we will show the simulation ability of our simulator in a what-if scenario.

Let us consider that an administration department in a hospital is considering a personnel arrangement for a pelvic treatment process. The treatment process they selected is the model *O* in Fig. 14.7, and three candidate resource models for this treatment are available in Table 14.1. *Resource model A* and *resource model B* both contain six human resources, but responsibilities of human resources are adjusted. Compared with *resource model A* and *resource model B*, there are two more human resources added in *resource model C* which intends to relieve workload on individuals

We provided our simulator with the workflow model and three resource models for the simulation. Then, we investigate the simulation results with ProM and estimated time consumption; see Table 14.2. As we can see, resource model B

Table 14.2 Simulation results with three resource models

	Time consumption					
	Workflow instance			Workitem		
Resource model	Min	Ave	Max	Min	Ave	Max
A	53	158	285	2	19.8	58
B	48	106	158	2	14.75	53
C	34	118	245	2	13.92	52

may be the best one for administration department, showing its advantages in both items of time consumption.

5 Conclusions

In this chapter, we have proposed a human resource centric simulator. There are three main contributions in this work.

Firstly, to our best knowledge, this is the first agent-based simulation approach that utilises the HTN-based state transition mechanism to model human resource behaviour in a workflow system. Secondly, we provide a general agent-based simulation architecture that can be integrated with a generic WfMS. This means our simulation architecture can be implemented as a simulation component for any WfMS. Thirdly, a what-if scenario has indicated that our simulator can be potentially used as a decision support tool and is therefore of interest to business analysts.

In our future work, we will extend this simulation system as a hybrid multi-agent system (MAS)/3D virtual world simulation system. Such an interactive visualisation will give extra resource model analysis support for business process analysts and enhances process model communication to stakeholders (Brown 2010).

Acknowledgments This research has been supported by Smart Service CRC in Australia.

References

Aalst WMP, Nakatumba J, Rozinat A (2010) Business process simulation. In: Jv B, Rosemann M (eds) Handbook on business process management, vol 1. Springer, Berlin/Heidelberg, pp 313–338

Agrawal R, Gunopulos D, Leymann F (1998) Mining process models from work-flow logs. Presented at the sixth international conference on extending database technology

Bandara W, Indlska M, Sadiq S et al (2007) Major issue in business process management: an expert perspective. Presented at the 15th European conference on information systems

Biffl WL, Smith WR, Moore EE et al (2001) Evolution of a multidisciplinary clinical pathway for the management of unstable patients with pelvic fractures. Ann Surg 233:843–850

Brown R (2010) Conceptual modelling in 3D virtual worlds for process communication. Presented at the Asia-Pacific conferences on conceptual modelling

Brown R, Rasmussen R (2009) Virtual environment visualisation of executable business process models. In: Rao NR (ed) Virtual technologies for business and industrial applications: innovative and synergistic approaches. IGI Global, p 350 (15 April 2010)

Croce MA, Magnotti LJ, Savage SA (2007) Emergent pelvic fixation in patients with exsanguinating pelvic fractures. J Am Coll Surg 204:935–939

Decker K, Li J (1998) Coordinated hospital patient scheduling. In: Proceedings of the third international conference on multi-agent systems, pp 104–111

Erol K, Hendler J, Nau DS (1994) Semantics for hierarchical task-network planning. University of Maryland at College Park

Hollingsworth D (1995) Workflow management coalition the workflow reference model. WfMC, Winchester

Isern D, Sánchez D, Moreno A (2010) Agents applied in heal care: a review. Int J Medical Inform 79:145–166

Mantovani F, Castelnuovo G, Gaggioli A et al (2004) Virtual reality training for health-care professionals. CyberPsychol Behav 6:389–395

Nicodemo A, Decaroli D, Pallavicini J et al (2008) A treatment protocol for abdomino-pelvic injuries. J Orthop Traumatol 9:89–95

O'Brien PD, Wiegand ME (1998) Agent based process management: applying intelligent agents to workflow. Knowl Eng Rev 13:161–174

Peter D, Manfred R, Klaus K (2000) Clinical workflows- the killer application for process-oriented information system. Presented at the international conference on business information systems

Pollock JL (1999) Planning agents. In: Wooldridge M, Rao A (eds) Foundations of rational agency, 1st edn. Springer, Dordrecht (March 31, 1999)

Recker J, Indulska M, Green P (2007) Extending representational analysis: BPMN user and developer perspectives. In: Alonso G, Dadam P, Rosemann M (eds) Business Process Management – BPM 2007, vol 4714, Lecture Notes in Computer Science. Springer, Brisbane, pp 384–399

Robert GS (2007) Verification and validation of simulation models. Winter simulation conference

Rozinat A, Wynn MT, van der Aalst WMP (2009) Workflow simulation for operational decision support. Data Knowl Eng 68:834–850

Russell S, Norvig P (2003) Artificial intelligence: a modern approach. Pearson Education, Upper Saddle River

Russell N, van der Aalst WMP, Hofstede AHMt et al (2005) Workflow resource patterns: identification, representation and tool support. In: Proceedings of the 17th conference on advanced information systems engineering (CAiSE'05), pp 216–232

Sadiq S, Indulska M, Bandara W et al (2007) Major issues in business process management: a vendor perspective. Presented at the 11th Pacific-Asia conference on information systems: managing diversity in digital enterprises, Auckland

Smith R (2006) Technology disruption in the simulation industry. J Def Model Simul: Appl Methodol 3:3–10

van der Aalst WMP, Hofstede AHM (2005) YAWL: yet another workflow language. Inf Syst 30:245–275

van der Aalst WMP, van Hee KM (2002) Workflow management: models, methods, and systems. MIT Press, Cambridge, MA

van der Aalst WMP, Weijters AJMM, Maruster L (2004) Workflow mining: discovering process models from event logs. IEEE Trans Knowl Data Eng 16:1128–1142

van der Aalst WMP, Reijers HA, Weijters AJMM et al (2007) Business process mining: an industrial application. Inf Syst 32:713–732

van der Aalst WMP, Nakatumba J, Rozinat A et al (2008) Business process simulation: how to get it right? BPM Centre

Wooldridge MJ (ed) (2002) Multi-agent systems: an introduction. Wiley, Chichester

Chapter 15
Evaluating the Synergies of Integrating E-Assessment and Software Testing

Tim A. Majchrzak and Claus A. Usener

1 Introduction

Teaching and learning rely on assessments. They help to measure performance and to keep track of learning success. Manual assessments are very time-consuming for teaching personnel. Therefore, e-learning and in particular e-assessment systems are introduced by higher education institutions to reduce recurrent work. Currently, available systems offer assessment with simple examinations only and do not support tasks that require higher-order cognitive skills (Heywood 2000) to be solved. In the computer science (CS) education, these skills are particularly important. Exercises requiring knowledge to be memorized only account for a fraction of the overall examination.

Our system *EASy* tries to close this gap by providing assessment for the CS and mathematics sciences education. It currently can be used to examine students on solving formal proofs (Gruttmann et al. 2008) and on their skills in formal specification techniques (Usener et al. 2010). In this chapter, we introduce EASy's extension for programing exercises in combination with software testing. Programing should always be followed by testing to reach an adequate level of software quality. Assessing both programing and testing is a complex task. There might be various *correct* solutions to a given task. Correct solutions might differ in quality, e.g., with regard to runtime characteristics. Even imperfect solutions might show valuable approaches.

EASy supports the complete workflow connected to programing exercises. Exercises are disseminated, and solutions are collected via EASy. Syntactic checks of program code are automatically done, whereas semantic checks remain with the tutors. To highlight the importance of testing, we included the test case generator

T.A. Majchrzak (✉) • C.A. Usener
European Research Center for Information Systems (ERCIS), University of Münster, Münster, Germany
e-mail: tima@ercis.de; usener@ercis.de

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_15,

Muggl (Majchrzak and Kuchen 2009). It automatically generates test case files from solutions that students uploaded. In this scenario, these files can be used to improve the solution until the final deadline for handing in an exercise has been reached. Thereby, EASy does not only aid tutors and supersedes circuitous paper-based processing of exercises but also supports students in their programing efforts—testing knowledge has been found to also improve students' programing skills (Marrero and Settle 2005). While the combination of e-assessment and testing is notable as such, we deem it essential to understand students' perception of new ways of assessment. Consequently, we did a survey to find out whether EASy actually helped students and whether they accept the system, or not.

This chapter makes several contributions. Firstly, it introduces our approach of combining e-assessment with software testing. Secondly, it explains the benefits for the CS education. Thirdly, it presents significant quantitative evaluation results. Fourthly, it discusses direction of future research based on qualitative findings. And fourthly, it motivates using e-assessment for higher-order cognitive skills, specifically in the fields of CS and mathematics. This chapter is structured as follows: Sect. 2 sketched the background of our research. Related work is discussed in Sect. 3. Integration details are given in Sect. 4, and evaluation results are presented in Sect. 5. Finally, a conclusion is drawn in Sect. 6.

2 Background

To understand our approach, we give an overview of e-assessment in CS and shortly introduce two software testing techniques that can be used in programing lectures.

2.1 E-Assessment in Computer Science

Assessments are an essential part of every teaching and learning scenario as they help to identify and measure individual learning success (Usener et al. 2010). However, not only the individual learning success can be identified; assessment results can also be seen as an indicator for lecture improvements (Pellegrino et al. 2001). Thus, teachers and learners can gain from assessment results.

After finishing their academic studies, CS students should be able to create and enhance computer systems. To reach this goal, they have to internalize basic computer science knowledge and, beyond that, to develop analytic, creative, and constructive skills. Furthermore, most of the relevant learning objectives cannot be reached by only learning factual knowledge; they require intensive involvement and continuous practice (Usener et al. 2010). This can be achieved by assigning exercises based on lecture content in which students take an active part during the problem-solving process. However, passively consumed information can be transferred to active knowledge particularly by applying it to problems with practical relevance (Dyckhoff 2008).

Having in mind that universities suffer from decreasing resources, low personnel capacities, and mass lectures, organizing formal assessments is very challenging. In light of this, e-assessment—the use of computers for examination purposes (Ruedel 2010)—is a promising possibility to offer formal assessments such as weekly exercises, which accompany the lecture. At the same time, e-assessment can enable new promising didactical concepts like immediate feedback facilitated by fast and automatic processing of workflows (Weicker and Weicker 2005). Current e-assessment systems usually provide assessment functionality for closed question types like multiple choice questions or short text answers (Cook and Jenkins 2010). These question types are not suitable to assess analytic, creative, and constructive skills, which is of importance in the CS education.

2.2 Applicable Software Testing Techniques

A large variety of testing techniques and software quality management practices exist (cf. Pezze and Young 2007). Which technique to choose depends mainly on the stage of software development (Majchrzak 2010). In early stages, testing mainly focuses on component testing; when the software becomes more complete, integration and system testing plays a predominant role. In general, software components are coded first, and afterward test cases are written, even if development and testing processes are parallelized. *Test-driven development* (TDD) changes this order (Astels 2002). In TDD—also called test-first development (Beck 2002, p. 203)—test cases based on specifications are implemented first, followed by the development of a specific software piece. By implementing only the required functionality, TDD keeps program code simple, which leads to better maintainability (Beck 2002, p. 195). Writing tests first and then implementing functionality corresponding to the test cases specified earlier require discipline (Astels 2002, p. 8). But with the possibility to test functionality, directly debugging costs can be reduced since debugging in later stages is more time-consuming due to the rising complexity of the source code (Beck 2002, p. 202). Tests that were written to reveal simple errors often also reveal more severe ones (DeMillo et al. 1978). TDD has been found to support students in judging whether their program is already correct or needs further improvement (Edwards 2003).

Back-to-back testing is a technique used for software systems in which failures are irreparable, like in embedded systems, or fixing goes along with (extremely) high expense. In this iterative strategy, n prototypes of one program are developed independently by different teams. Since all prototypes are based on the same specification, they are supposed to be semantically equal (Roitzsch 2005, p. 376). After each iteration, the prototypes' test results are compared, and differences are passed on to the development teams. The teams are forced to improve their prototype since different behaviors indicate that at least one prototype is erroneous. The process is repeated until all prototypes behave equally. Most other testing methods have different lacks due to their heuristic nature, which back-to-back testing as a *diversifying* test tries to overcome. Back-to-back testing can be, despite the highly increased effort compared to other methods, cost-effective under certain conditions.

3 Related Work

Since we are combining two different tools and thereby two distinct fields of research, related work can be discussed from three positions: e-assessment for programing exercises, automated test case generation, and combination of both.

Regarding e-assessment of programing exercises, there already has been some research done. We give a brief overview of the systems *Praktomat*, *DUESIE*, and *ELP* that differ in their assessment approach. They all provide at least assessment functionality for Java programing language exercises. Praktomat and DUESIE are also able to assess exercises on other object-oriented and functional programing languages. Besides systems for programing exercises in Java, tools to support programing assignments in other languages exist. For example, Bridgeman et al. (2000) describe the Web-based system PILOT that can be used to visualize a variety of computer science concepts. It can also be used to "test" students (Bridgeman et al. 2000), whereas the concrete grading mechanisms are unclear. However, comparing tools for other programing languages and in particular non-object-oriented ones was only helpful if assessment strategies could be transferred.

Praktomat focuses on support for programing workshops of second year students. Krinke et al. (2002) describe the main feature of their system as the ability that students can view and comment other solutions after a first submission and improve it in following iterations. This approach should give students an opportunity to learn from other students' solutions, rethink their own solution, and improve it in a following step. Additionally, static and dynamic tests are used to evaluate a student's solution and can be used as a basic test before handing in a solution. Similar to Praktomat, DUESIE uses static and dynamic test to assess coding style and functionality of a student's solution (Hoffmann et al. 2008). Both systems use external software for static and dynamic checks. Contrary to Praktomat, DUESIE does not provide the possibility to check a solution in advance, which would protect students from solving an exercise by a *trial-and-error* approach (Hoffmann et al. 2008). In both systems, an automated check of every solution is supplemented by a final manual assessment. ELP pursues a different goal: it does not benchmark students' solutions; instead, it provides self-assessment ability. It is developed to help students with no or little programing experience to learn use of basic Java constructs (Truong et al. 2004). Thus, it provides brief *fill-in-the-gap* exercises, each emphasizing only one language feature. Solutions are checked by first simplifying them and by then comparing them with stored example solutions (Truong et al. 2004). Feedback is given directly due to differences between handed in solution and example solutions and should motivate students to improve their work. Of the introduced systems, only ELP checks solutions without human interaction, but it is limited to very small and well-defined problems. The other systems are suited for more complex problems but require manual correction.

A wealth of work has been published on automated generation of test cases (TCG). However, TCG still faces severe limitations (Fewster and Graham 1999, p. 10ff). No general purpose solutions for full automation has been developed, yet.

In this chapter, TCG is a means to an end. Therefore, we will only briefly discuss three closely related approaches. A comprehensive overview of related work is given in Majchrzak and Kuchen [2009, 2011].

The tool of Fischer and Kuchen (2007, 2008) is very similar in concept to Muggl. However, it has been developed for the functional logic programing language *Curry*. It also does not include a *constraint solver*. *Pex* (Tillmann and Halleux 2008) is a tool for the Microsoft Common Language Infrastructure (CLI) that generates test cases for .NET-based programs. Conceptually, it shares many ideas with Muggl such as using dynamic symbolic execution and a constraint solver. In contrast, it does not employ an own virtual machine but utilizes the .NET profiling API. Its constraint solver *Z3* (Moura and Bjoerner 2008) is a satisfiability modulo theory (SMT) solver, which is different to the kind of solver that Muggl uses. *IBIS* (Doyle and Meudec 2003) is based on symbolic execution of Java bytecode and on constraint solving. In contrast to Muggl, it uses a *Prolog* representation of Java `class` files. Unfortunately, no recent progress has been reported on the tool; the described status is incomplete.

For both e-assessment of higher cognitive skills and TCG, hardly any advanced solutions exist. Therefore, it is not surprising that no e-assessment tools that integrate software testing can be identified, which are beyond an experimental state. Nevertheless, some approaches can be compared to our work.

In his work, Ihantola describes a system that automatically generates test data and visualizes the results using *Java PathFinder* (Ihantola 2007). While assessment strategies are discussed, the solution is inherently different to ours since it uses a *model checker*. Test cases follow a predefined scheme (the model); we generate test cases based on the structure of the student's solutions. A recent paper that apparently extends the earlier work described the inclusion of mutation analysis and code coverage (Aaltonen et al. 2010). Despite some reported problems, the approach is very interesting; we identified code coverage as a possible addition for our approach. Muggl tracks coverage anyway, it just needs to be visualized for the students.

Spacco and Pugh (2006) intend to motivate students to write test cases and work in a test-driven manner. They have also developed a prototypic system, called *Marmoset*, which allows students to upload solutions of programing assignments. Marmoset executes exemplary test cases on them. However, Spacco and Pugh focus on didactical aspects of teaching TDD. Jackson and Usher (1997) describe *ASSYST*, which helps tutors to assess programing exercises. Besides other criteria, it checks whether test cases specified by students lead to code coverage in their solutions. However, neither Marmoset nor *ASSYST* is capable of generating test cases. Similarly, *Scheme-robo* can execute test cases on students' solutions, but it cannot generate test cases (Saikkonen et al. 2001). Elbaum et al. describe a system that is meant to provide tutorial functionality to "engage students in learning software testing strategies" (Elbaum et al. 2007). While assessment is part of their work, no automated test case generation is intended. Finally, Goldwasser motivates the inclusion of software testing in the curriculum (Goldwasser 2002). His work focuses on combining programing and software testing and on supporting assessment. As with the previous approaches, no automated test case generation is possible.

None of the described assessment tools with their different assessment approaches provide any possibility of generating test cases on submitted solutions. Other e-assessment tools usually provide only simple exercise types, and if they offer assessment of Java exercises, they use similar methods. A broader overview of these tools can be found in Gruttmann [2010]. Many different approaches for TCG exist, but usually, they are bound to certain preconditions or results are of low value. We found only one work combining e-assessment and TCG. It is based on model checking.

4 Integrating E-Assessment and Software Testing

In the following, we first introduce EASy and Muggl separately before we describe how a combination of both tools is arranged, followed by an application scenario.

4.1 EASy

EASy—an acronym for *E-A*ssessment System—is a Web platform developed with the aim of providing electronic assessment of complex exercises in the CS education (Gruttmann 2010). To be more specific, EASy supports lecturers, tutors, and students during the workflow of electronically providing, solving, submitting, correcting, and returning CS exercises. At the same time, the system tries to keep the postulated analytic, creative, and constructive challenges by implementing new didactical concepts enabled by electronic submission. The system is designed as a Web application to offer time- and place-independent access. Conceptually, each exercise type is designed as a separate module which eases realization of different exercise types, each with its specific requirements. Additionally, effortless extension with new types is enabled.

Currently, EASy provides four different exercise types: mathematical proofs, software verification exercises based on *Hoare Logic*, programing exercises, and a multiple-choice module. Further extensions are planned. The EASy module for mathematical proofs is designed to assess students' ability to solve mathematical problems (Gruttmann et al. 2008). The module offers a variety of strategies to solve a proof. Students successively apply arithmetic and Boolean rules until a proof is solved. Similar in concept, the module for formal specification (Usener et al. 2010) deals with partial correctness proofs using *Hoare Logic* (Hoare 1969). Both modules support students in conducting the exercise and help tutors by, e.g., highlighting correct subproofs.

The programing exercise module of EASy currently supports Java. Each exercise can comprise several Java classes. Students are provided with class skeletons, exercise description, and a set of test settings. They get feedback for their solutions in two ways. Until a fixed deadline is reached, they can upload (improved) solutions and receive automatically generated feedback. The system first checks the code for compiling errors and missing elements like required methods and variables. It also preforms predefined JUnit test cases to tell whether the code fulfills the main demands or not. Students are encouraged to rethink and to continuously revise their solutions (Weicker and Weicker 2005). In a second phase, after the deadline has been reached, their solutions are first checked automatically and the submitted work is annotated with the generated remarks. These automatic tests contain compiling errors, missing classes, and methods as well as violation of coding style guidelines, markings or error-prone code patterns, and complex JUnit test cases. Afterward, the students' solutions are checked by tutors and manually graded. We do not think that grading should be automated completely. However, the evaluation done by EASy relieves tutors of much manual work.

4.2 Muggl

Muggl (Majchrzak and Kuchen 2009) is a tool that automatically generates test cases. Contrasting most other tools that either test randomly or rely on preconditions (such as models), Muggl derives test cases from a program's structure. Muggl includes a custom Java virtual machine (JVM) (Lindholm and Yellin 1999) that symbolically executes Java bytecode. Input parameters are treated as logic variables. These variables take the value of terms during execution; terms are composed of constants, operation symbols, and logic variables. Using an *iterative-deepening* depth-first search, Muggl finds possible path through the *search tree* of all possible computations. Tuples of input parameters and *constraints* that describe possible outputs are saved as *solutions*. They can be used to generate test cases by assigning constant value for expected outputs.

A number of features known from (constraint) logic programing are embedded into Muggl. In particular, it generates *choice points* to keep track of states that it yet has to visit. Moreover, it uses a *backtracking* algorithm that reverts execution to a former state using a *trail*. In general, the number of states to explore is infinite. To achieve a feasible performance, a number of sophisticated techniques are used. In particular, a *constraint solver* (Lembeck et al. 2004) ensures that only states that can possibly be reached are visited. Muggl tries to generate as little test cases as are required to find all defects in a program. A small number of test cases are faster to execute and can more easily be checked by humans. Therefore, Muggl uses a coverage-based elimination algorithm (Majchrzak and Kuchen 2009) to remove redundant test cases. Eventually, Muggl writes JUnit test case files which can be executed directly.

The above introduction of Muggl is extremely short; details are out of scope of this chapter. A detailed description of the status of Muggl and examples can be found in Majchrzak and Kuchen [2011]. Please note that Muggl cannot (yet) generate test cases for arbitrary programs. It still fails for larger programs due to their immense state spaces. It however is suitable for typical programs coded in exercises.

4.3 Steps Toward the Integration

The integration of EASy and Muggl requires two steps. Firstly, both systems need to be equipped with interfaces. Secondly, they have to be integrated.

Due to EASy's modular structure, integrating Muggl is simple. Existing modules hardly have to be changed, but the system has to become able to invoke Muggl. Moreover, EASy has to become capable of processing the test cases generated by Muggl. To guarantee EASy's availability, the invocation of Muggl is encapsulated. Despite using much computational power, enough resources remain for EASy.

Muggl's execution core is independent of its graphical user interface (GUI). It provides interfaces to control the symbolic JVM and retrieve data from it. Since most functionality is not required by EASy, a *wrapper* was implemented as a convenient interface to Muggl. It allows EASy to set a basic configuration, to start and control execution, and to fetch results (either a test case or data on generation failure). The wrapper also aborts execution after a maximum runtime. We wanted to provide students with immediate results and abort execution after 5 s.

4.4 Application Scenario

To evaluate our approach, we used the new EASy module in an undergraduate course on programing. We had to find an exercise that was simple enough to be solved by most students but at the same time offered challenges with regard to style and performance. We specified an algorithm that blurs images, which are delivered as one-dimensional arrays. This is rather easy; nevertheless, including all specified exceptions and elegantly calculating the blurred pixels require a careful implementation. Our exemplary solution is shown in Fig. 15.1.

Students were provided with the exercise and a JUnit test case file for our solution generated by Muggl. They then developed their own solution in a test-driven manner: the goal was to successfully execute the exemplary test case with their program. Once the students had a running program, they could upload it to EASy. They were then provided with another test case generated by Muggl for their own solution. By checking if it was semantically equal to the exemplary test case, students could find out whether their solution was complete. Differences had to be interpreted by them in order to find ways to improve their solutions. This back-to-back testing step could be repeated until students were confident that their solution was correct.

```java
public class Blur {
  public static int[] simpleBlur(int[] image) {
    if (image == null)
      throw new IllegalArgumentException("Null specified.");
    int length = image.length;
    if (length == 0)
      throw new IllegalArgumentException(
        "Cannot blur empty image.");

    // Check whether the image is quadratic.
    int n = length;
    while (n * n != length) {
      n /= 2;
      if (n == 1)
        throw new IllegalArgumentException("Not quadratic.");
    }

    if (n < 3)
      throw new IllegalArgumentException("Sides are too
          short.");

    int[] image2 = new int[length];
    for (int a = 0; a < length; a++) {
      int neighbours = 0;
      long pixel = 0L;

      if (a % n != 0) { // Left pixel.
        neighbours++;
        pixel += image[a - 1];
      }
      if (a >= n) { // Upper pixel.
        neighbours++;
        pixel += image[a - n];
      }
      if (a % n != n - 1) { // Right pixel.
        neighbours++;
        pixel += image[a + 1];
      }
      if (a < length - n) { // Lower pixel.
        neighbours++;
        pixel += image[a + n];
      }
      // Calculate the new value.
      // Round it to get precise results.
      pixel += (8 - neighbours) * image[a];
      pixel = ((pixel * 10 / 8L) + 5) / 10;
      image2[a] = (int) pixel;
    }

    return image2;
  }
}
```

Fig. 15.1 Exemplary solution provided to students in our scenario

5 Evaluation

To evaluate our approach, the task described in Sect. 4.4 was used as an exercise in a first-year lecture on *introduction to programing*. Students were asked to answer a questionnaire. In the following, we present and discuss the findings.

5.1 Survey Results

The questionnaire was made up of four parts: demographics, questions regarding the general usage of EASy, questions concerning testing, and questions about the general attitude toward e-assessment. It was provided along with the exercise; both were voluntarily. The students were free to skip questions. Therefore, some students did not answer every question. Most of the questions were single choice although some had to be answered textually. Of 311 students who applied for the tutorial, more than 250 students tried to solve the exercise, and noticeable 187 students answered the questionnaire. To give some more demographic data, 96 % of the answers came from first year students; only 10 % of the students who answered were female. Most of the students (53 %) were enrolled for *information systems*, 17 % for the major *information science*, and the rest (each 10 %) were equally divided into *geoinformatics*, *mathematics*, and *physics*.

Concerning the general usage of EASy, we first asked whether the students read the systems' documentation, and surprisingly almost 66 % did not read it. The usage of EASy was by choice; thus, we asked how often they used it: more than half of the students (57 %) stated to use the system for every exercise, whereas one third of them used the system at least from time to time, and less than 10 % of them did not use EASy at all. The main reason for using EASy was given as the ability to check functionality of own results in advance (89 %), whereas further answers, like saving of time, measurement of improvement, and others, played marginal roles (each ca. 4 %). Reasons for using EASy only from time to time or not at all did not show a clear tendency. Some students experienced difficulties using EASy (11 %) or found the usage too time-consuming (11 %); in some cases, the tutor did not support students (11 %), and some even did not see any advantage in using EASy (16 %). The other students stated in a free text field that they did not use EASy due to system availability or that due to teamwork another group member handled the use of EASy. Usability of the system seems to be good since less than 9 % graded the usability to be poor (7.5 %) or bad (1 %), more than 56 % remarked usability to be good, and 9 % declared it to be excellent.

Regarding assessment of the students' work, we asked if the first test case generated for their solution was equivalent to the one given in the exercise. Remarkably, 30 % of the students stated that they submitted a solution which caused the generator to provide a test case equivalent to the provided one.

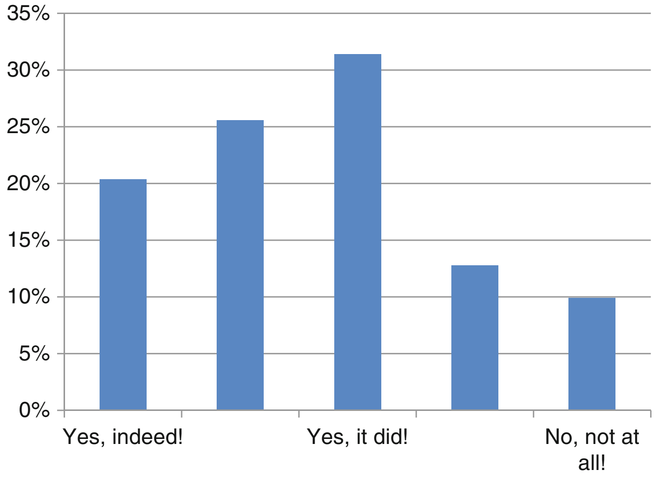


Fig. 15.2 Results to the question: "if a test case failed in EASy, did the failure motivate you to improve your solution?"

The remaining students were asked whether they were able to improve their submission according to the results of test case generation. From the remaining 70 % who did not get the correct test case at the first try, 62 % were able to interpret the generated test case and adapt their solution according to the results. Students who did not succeed were asked to explain the problems they had. Answers are diverse: 23 % did not succeed but felt confident that their solution was correct anyway. Further, 17 % found differences to be marginally different and therefore did not change their solution. Some students (15 %) stated that Muggl had problems to generate an appropriate test suite, which occur if a solution is too complex to generate a test suite in a feasible time. We further asked students to state whether they found it informative to compare test cases generated on their solution with the given test case. More than 63 % said they gained from the comparison. Almost 50% of students who did not gain from the comparison were not able to generate a test case. Focusing on students who succeeded to generate at least one test case, nearly 75 % found the comparison informative. We also asked students if a failed test case shown in EASy motivated them to improve their solution. As illustrated in Fig. 15.2, 77 % of the students were motivated to improve and rethink their solution due to a failed test case.

We finally asked the students to give their opinion toward diagnostic, formative, and summative e-assessment. While students appreciate diagnostic e-assessment for self-assessment purposes (57 % strong support; 32 % enhanced support), the attitude toward formative e-assessment is lower (18 % strong support; 29 % enhanced support). Still, 72 % support formative e-assessment. In contrast, attitude toward summative assessment, such as final exams, indicates a significant difference; more than 31 % strongly refuse summative e-assessment.

5.2 Qualitative Findings

The last section presented quantitative results of our questionnaire, which are adequate to judge our approach; however, qualitative feedback helps us to get indications on how to improve the system. We desist from quoting students since the questionnaire was answered in German; instead, we summarize main tendencies.

We asked students to provide us with suggestions for improvements on the integration of EASy and Muggl. The feedback we got was multifarious. Some students would have wished to get a more detailed overview of Muggl's processing mechanism beforehand; for them, the generated test case was challenging to interpret as it seemed to be a result of a *magical black box*. Some found the task description too detailed and constricting their way of solving the exercise, whereas others would have wished for a more guided exercise. These contradictions may be explained by the different knowledge levels; the lecture is addressed to first year students with different programing backgrounds. Some students would have appreciated an automatic comparison of the generated and the given test case. Of course, this feature would conflict with the fundamental idea of back-to-back testing. We were pleased to notice that students truly enjoyed the exercise and stated they wished to get more exercises in a test-driven manner since the learning effect was steep. This underlines the findings of the previous section.

Furthermore, students were asked to give feedback concerning the handling of the system. We got a lot of positive feedback. Many students praised simplicity, structured presentation of information, the absence of needless features, and consequential the intuitive handling. On the other hand, some complained about small handling issues concerning the navigation bar or a missing multiple file upload. Nonetheless, positive feedback regarding the GUI outweighed the critical comments.

In the quantitative part, we asked for the general attitude toward e-assessment. To give chance to state a more detailed opinion on the usefulness of e-assessment in certain lectures, we asked the students to list lectures and fields of study, which, in their opinion, are feasible for e-assessment support. If students appreciate e-assessment, tendencies can be identified. The highest benefits of e-assessment are seen in natural science subjects with mathematical background where, in their opinion, clear results without ambiguities are common. This endorses Gruttman et al.s' (2008) findings toward using EASy for mathematical proofs. One student even suggested e-assessment for phonology. Mentionable are also some reserved student opinions, which fear the disregarding of individual skills.

We were amazed by many outstanding feedbacks; some even underlined the importance of e-assessment *research*. Although students legitimately complained about the sporadic instability of the system, we could not avoid this with our current setup. From face-to-face feedback of tutors, we found out that quality of students' solution rose due to new concepts enabled by EASy.

5.3 Discussion and Limitations

There are some limitations of our approach that have to be mentioned. As argued by some students, the integration of EASy and Muggl is not yet perfect. We have to provide more information to the students on how test cases are generated. It also has to be ensured that EASy is available with hardly any downtimes and that test cases can be provided for syntactically correct solutions even if they have been implemented very inelegantly. Probably, means of supporting students in comparing their test case to the exemplary one have to be found. However, these limitations can be overcome in future versions of the EASy module.

The survey only gives a first idea of how the new module was perceived. Despite the high number of participants, results should be confirmed by surveys with students from other universities and—ideally—from other countries. Nevertheless, the feedback provided is extremely valuable considering the novelty of our approach.

6 Conclusion and Future Work

In this chapter we presented work on EASy, a system for e-assessment of exercises that require higher-order cognitive skills. We introduced a new module for programing exercises and its integration with automated test case generation. Moreover, we motivated to teach testing techniques such as test-driven development and back-to-back testing by integrating them into exercises. We illustrated our approach and underlined its novelty by highlighting differences to existing work. Additionally, we presented results from a survey with almost 200 students. Results were diverse, but the general approach was perceived very positively.

EASy already comprises of several modules. Future development will include both the development of new modules and improvements to existing ones. The latter work will be based on feedback we received from students. We aim at providing a system that seamlessly integrates into the teaching and learning workflows and that is robust and flexible. Moreover, we will search for ways of including other subjects of study that can also be provided with e-assessment. Eventually, we will investigate in which settings systems such as EASy can be used for mandatory exams.

References

Aaltonen K, Ihantola P, Seppälä O (2010) Mutation analysis vs. code coverage in automated assessment of students' testing skills. In: Proceedings of the OOPSLA '10. ACM, New York, pp 153–160

Astels D (2003) Test driven development: a practical guide. Prentice Hall Professional, Upper Saddle River/London

Beck K (2002) Test-driven development by example. Addison-Wesley, Reading/Harlow

Bridgeman S, Goodrich MT, Kobourov SG, Tamassia R (2000) Pilot: an interactive tool for learning and grading. In: Proceedings of the thirty-first SIGCSE technical symposium on computer science education, SIGCSE '00. ACM, New York, pp 139–143

Cook J, Jenkins V (2010) Getting started with e-assessment. University of Bath, project report

DeMillo RA, Lipton RJ, Sayward FG (1978) Hints on test data selection: help for the practicing programmer. Computer 11:34–41

de Moura L, Bjoerner N (2008) Z3: an efficient SMT solver. LNCS, vol 4963/2008. Springer, Berlin, pp 337–340

Doyle J, Meudec C (2003) IBIS: an interactive bytecode inspection system, using symbolic execution and constraint logic programming. In: Proceedings of the PPPJ '03. Computer Science, New York, pp 55–58

Dyckhoff A (2008) An integrated web-based exercise module. In: Proceedings of the IASTED CATE 2008. ACTA, Anaheim, pp 244–249

Edwards SH (2003) Rethinking computer science education from a test-first perspective. In: Proceedings of the OOPSLA '03. ACM, New York, pp 148–155

Elbaum S, Person S, Dokulil J, Jorde M (2007) Bug hunt: making early software testing lessons engaging and affordable. In: Proceedings of the ICSE '07. IEEE CS, Washington, pp 688–697

Fewster M, Graham D (1999) Software test automation: effective use of test execution tools. ACM/Addison-Wesley, New York

Fischer S, Kuchen H (2007) Systematic generation of glass-box test cases for functional logic programs. In: Proceedings of the 9th PPDP '07. ACM, New York, pp 63–74

Fischer S, Kuchen H (2008) Data-flow testing of declarative programs. In: Hook J, Thiemann P (eds) Proceedings of the ICFP '08. ACM, New York, pp 201–212

Goldwasser MH (2002) A gimmick to integrate software testing throughout the curriculum. SIGCSE Bull 34:271–275

Gruttmann S (2010) Formatives E-assessment in der Hochschullehre—Computerunterstützte Lernfortschrittskontrollen im Informatikstudium. Ph.D. thesis, University of Münster, Münster

Gruttmann S, Böhm D, Kuchen H (2008) E-assessment of mathematical proofs: chances and challenges for students and tutors. In: Proceedings of the CSSE '08. IEEE Computer Society, Washington, pp 612–615

Heywood J (2000) Assessment in higher education: student learning, teaching, programmes and institutions. Jessica Kingsley Publishers, London/Philadelphia

Hoare CAR (1969) An axiomatic basis for computer programming. Commun ACM 12(10):576–580

Hoffmann A, Quast A, Wismüller R (2008) Online-Übungssystem für die Programmierausbildung zur Einführung in die Informatik. In: Proceedings of the DeLFI '08. pp 173–184

Ihantola P (2007) Creating and visualizing test data from programming exercises. Inform Educ 6:81–102

Jackson D, Usher M (1997) Grading student programs using ASSYST. In: Proceedings of the SIGCSE '97. ACM, New York, pp 335–339

Krinke J, Störzer M, Zeller A (2002) Web-basierte Programmierpraktika mit Praktomat. Softwaretechnik-Trends 22(3):S.51–S.53

Lembeck C, Caballero R, Mueller RA, Kuchen H (2004) Constraint solving for generating glass-box test cases. In: Proceedings WFLP '04. RWTH, Department of Computer Science, Aachen, pp 19–32

Lindholm T, Yellin F (1999) The Java virtual machine specification, 2nd edn. Prentice Hall

Majchrzak TA (2010) Improving the technical aspects of software testing in enterprises. Int J Adv Comput Sci Appl 1(4):1–10

Majchrzak TA, Kuchen H (2009) Automated test case generation based on coverage analysis. In: TASE ’09: proceedings of the 2009 3rd IEEE international symposium on theoretical aspects of software engineering. IEEE Computer Society, Los Alamitos, pp 259–266

Majchrzak TA, Kuchen H (2011) Muggl: the muenster generator of glass-box test cases. In: Becker J, Backhaus K, Grob H, Hellingrath B, Hoeren T, Klein S, Kuchen H, Müller-Funk U, Thonemann UW, Vossen G (eds) Working papers no. 10. European Research Center for Information Systems (ERCIS)

Marrero W, Settle A (2005) Testing first: emphasizing testing in early programming courses. SIGCSE Bull 37:4–8

Pellegrino JW, Chudowsky N, Glas R (2001) Knowing what students know: the science and design of educational assessment. National Academy, Washington, DC

Pezze M, Young M (2007) Software testing and analysis: process, principles and techniques. Wiley, Hoboken

Roitzsch EHP (2005) Analytische Softwarequalitätssicherung in Theorie und Praxis: Der Weg zur Software mit hoher Qualität durch statisches Prüfen, dynamisches Testen, formales Beweisen. Monsenstein und Vannerdat

Ruedel C (2010) Was ist E-assessment. In: Ruedel C, Mandel S (eds) E-Assessment: Einsatzszenarien und Erfahrungen an Hochschulen. Waxmann, Münster

Saikkonen R, Malmi L, Korhonen A (2001) Fully automatic assessment of programming exercises. SIGCSE Bull 33:133–136

Spacco J, Pugh W (2006) Helping students appreciate test-driven development (tdd). In: Proceedings of the OOPSLA ’06. ACM, New York, pp 907–913

Tillmann N, de Halleux J (2008) Pex–White box test generation for .NET. In: Proceedings of the 2nd international conference on tests and proofs. pp 134–153

Truong N, Roe P, Bancroft P (2004) Static analysis of students’ Java programs. In: Proceedings of the ACE ’04. Australian Computer Society, Darlinghurst, pp 317–325

Usener CA, Gruttmann S, Majchrzak TA, Kuchen H (2010) Computer-supported assessment of software verification proofs—towards high-quality E-assessments in computer science education. In: Proceedings of the ICEIT ’10. IEEE Computer Society, pp 115–121

Weicker N, Weicker K (2005) Didaktische Anmerkungen zur Unterstützung der Programmierlehre durch E-Learning. In: Haake JM, Lucke U, Tavangarian D (eds) DeLFI 2005: 3. Deutsche e-Learning Fachtagung Informatik. GI e. V., Bonn, pp 435–446

Chapter 16
Multiple-View Architecture Model for Distributed Real-Time Systems Using MARTE

Eduardo Augusto Silvestre and Michel dos Santos Soares

1 Introduction

Currently, the object-oriented (OO) paradigm is the most common approach for modeling software systems. This includes object-oriented programming languages, methods, and modeling languages. Among all OO notations, UML (OMG 2010a) is the most used one, despite the many well-known issues observed both in theory and practice, such as ambiguity (Beneken et al. 2003) and lack of defined semantics (André et al. 2007, Dobing and Parsons 2006).

Historically, OO methods such as ROOM (Selic et al. 1994) have been applied to model real-time systems. With the advent of UML as the de facto modeling language, UML has been applied to model real-time systems in a variety of domains. However, UML alone is not considered sufficient to model real-time systems for many reasons. The representation of time constraints with UML is poor, as there is no syntax for the expression of time constraints beyond time-dependent conditions in actions (Graf et al. 2006). Behavior diagrams, such as the sequence diagram, cannot represent time constraints effectively, as they are essentially untimed, expressing only chronological order (André et al. 2007). UML lacks the ability to express nonfunctional properties (Demathieu et al. 2008). In addition, the many scenarios described by different sequence diagrams hide important information, such as resources to be shared and overcomplex objects (Soares et al. 2008).

E.A. Silvestre (✉)
Federal Institute of Triângulo Mineiro (IFTM), Patrocínio, Brazil
e-mail: eduardosilvestre@iftm.edu.br

M.d.S. Soares
Federal University of Uberlândia (UFU), Computing Faculty (FACOM), Uberlândia, Brazil
e-mail: michel@facom.ufu.br

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_16,

In order to solve these issues, the profile SPT (UML Profile for Schedulability, Performance, and Time) was proposed by OMG (2005). SPT was applied to model real-time systems in many domains (Baar et al. 2004, Medina et al. 2004), but it was not well accepted in general as it lacked important modeling power and was considered hard to apply. Therefore, significant suggestions for improvements and consolidation (Demathieu et al. 2008) were proposed. These include new requirements for specifying both software and hardware elements, alignment with UML 2, and modeling of additional nonfunctional properties such as power consumption and memory size (Demathieu et al. 2008).

A new UML profile to design real-time systems was proposed: MARTE (*Modeling and Analysis of Real-Time and Embedded systems*) (OMG 2010d), with focus on the UML version 2. MARTE is an evolution over SPT. MARTE is a recently adopted standard profile that specializes UML by adding concepts for modeling and analysis of real-time and embedded systems. The practical application of MARTE is still very incipient (Demathieu et al. 2008), but some results have been published in domains such as system-on-chip (Quadri et al. 2010) networks (Elhaji et al. 2011) and in the development of software product lines (Belategi et al. 2010).

The objective of this chapter is to propose a multiple-view software architecture model to be applied to the design of distributed real-time systems, in which MARTE is the main modeling language. The architecture should be seen as complementary to other architectures, not as a replacement, as the focus is to describe only the most relevant views regarding the design of distributed real-time systems.

2 The MARTE Profile

As illustrated in Fig. 16.1, MARTE is organized around three main packages. The MARTE Foundations package defines concepts for real-time and embedded systems. These concepts cover the modeling of generic applications and platform artifacts. The Foundations package is composed of the following sub-packages: Core Elements (CoreElements), Nonfunctional Properties Modeling (NFPs), Time Modeling (Time), Generic Resource Modeling (GRM), and Allocation Modeling (Alloc). These foundation concepts are refined for design purpose into the MARTE Design Model package and for analysis purpose into the MARTE Analysis Model package (Mraidha et al. 2008).

The MARTE Design Model package provides support required for a variety of activities, from specification to detailed design of real-time systems. The MARTE Design Model is composed of four sub-packages. The Generic Component Model (GCM) package presents additional concepts to address the modeling of artifacts in the context of real-time systems. The GCM extends the UML component model by adding two specializations of ports: message ports and flow ports. Message ports support a request/reply communication paradigm. Messages that flow across ports represent operation calls or signals. Flow port enables data flow-oriented communication between components. The messages that flow across ports represent data

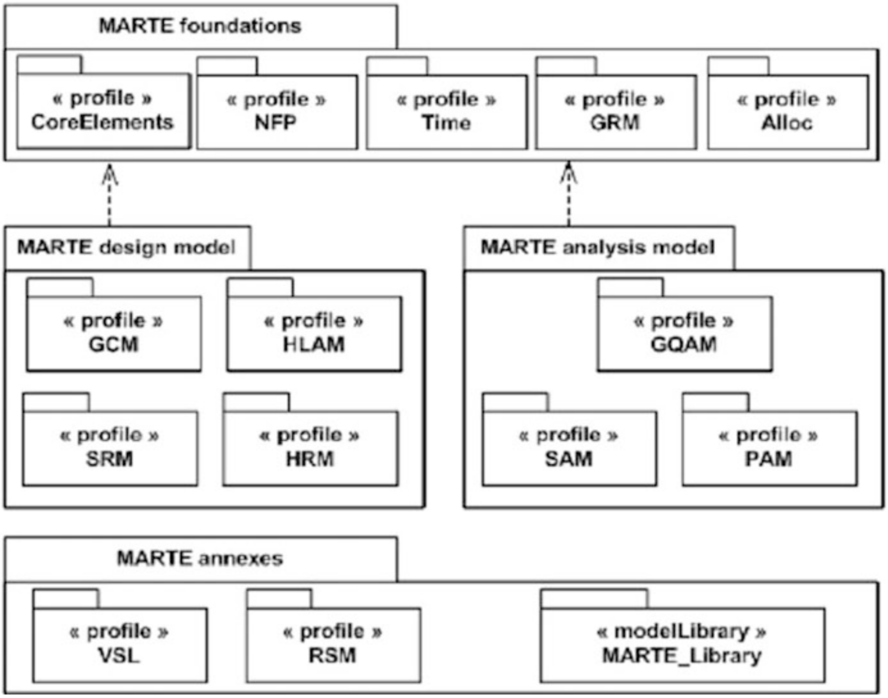


Fig. 16.1 The MARTE profile (OMG 2009)

items. Both pull and push formats are supported by MARTE. The High-Level Application Modeling (HLAM) package provides high-level modeling concepts to deal with real-time and embedded features modeling. The HLAM package allows to specify temporal properties of calls. HLAM can model active objects (active components), i.e., entities that provide their own thread of control. The Detailed Resource Modeling (DRM) package provides a set of detailed resources for modeling both software and hardware platforms by specializing the concepts defined within the GRM. DRM is divided into two sub-packages. The Software Resource Modeling (SRM) package focuses on modeling of application programming interfaces of software multitasking platform and defines features for parallel execution and interaction between concurrent contexts. The Hardware Resource Modeling (HRM) package focuses on modeling hardware platform through different views and detail levels and defines constructs such as nodes and devices.

The MARTE Analysis Model package offers concepts for the analysis of models. The analysis of models can detect problems early in the development life cycle and is useful to reduce cost and risk of software development. The MARTE Analysis Model is composed of three sub-packages. The Generic Quantitative Analysis Modeling (GQAM) package supports generic concepts for types of analysis based on system execution behavior, which may be represented at different levels of detail. The GQAM includes specialized domains in which the analysis is based on the software behavior, such as performance, schedulability, and availability. GQAM is used to specify Schedulability Analysis Modeling (SAM) and Performance Analysis Modeling (PAM), offering facilities to annotate models with information required to perform schedulability or performance analysis. The SAM package provides specific concepts for schedulability in the context of real-time architectures. The PAM package describes the analysis of temporal properties, including a statistical measure. The PAM techniques include simulations, extended queueing models, and discrete-state models such as stochastic Petri nets.

MARTE Foundations, the main package, defines features in the context of software architecture. The Causality package, in CoreElements, describes the basic elements for behavioral modeling and their run-time semantics. The objective of the Causality package is to show execution at run time. This model is used as a basis for any dynamic model description associated with the MARTE profile.

Nonfunctional requirements are fundamental when designing software architectures. For this reason, a specific package was included in MARTE to deal with nonfunctional requirements. The package NFPs provides the capability to describe several types of values related to physical quantities, such as time, mass, and energy. The package deals with how NFPs are described and the relationships between different NFPs. NFPs also provide relevant characteristics to deal with requirements such as availability, reliability, safety, integrity, and maintainability, as, for instance, throughput, delays, overheads, scheduling policies, deadline, or memory usage. Furthermore, MARTE uses the stereotypes from the Value Expression Language (VSL) to specify constraints, properties, and attributes particularly related to nonfunctional requirements.

The Time package has concepts for modeling multiple-time base models and concepts for accessing time structure. In addition, it defines three types of time: causal (concerned about instruction precedence), clocked (adds a notion of simultaneity), and physical (precise accurate modeling or real-time duration values).

The GRM package offers concepts to model a general platform for executing real-time embedded applications, such as modeling and executing platforms at different levels of details and modeling both hardware and software platforms. The GRM is divided into packages. The ResourceCore package defines basic elements and their relationships. The ResourceTypes package defines fundamental types of resources. The ResourceManagement package defines specific management resources and their associated services. The GRM is planned to be used not only for further extension in software and hardware platform models, or in the analysis models of this specification, but also as a way to describe resources and platform architectures at a high abstraction level, when design choices and analysis techniques to use for verification are probably still undecided.

The development of distributed real-time systems includes resource allocation decisions at multiple levels of refinement. Current trends to software integration make these decisions tightly coupled to the software architecture. Within MARTE, the Alloc package model the allocation of elements onto the available resources (the execution platform).

3 Architectural Views of Distributed Real-Time Systems Using MARTE

The proposed architecture is based on multiple views, inspired by previous works (Kruchten 1995, Soni et al. 1995). The ANSI/IEEE 1471–2000 architecture reference framework also specifies that an architecture is organized by multiple views (Maier et al. 2004). The chosen views in this research are Process, Time, Resources, and Allocation of Resources. These views are related to distributed real-time systems and are not well described in previous works. The views proposed in this chapter should be used together with other multiple-view models of architecture (they are not mutually exclusive).

The Process View is used because it captures concurrency, synchronization, and nonfunctional requirements aspects of the design, such as performance and availability, and it addresses issues of system's integrity and fault tolerance (Kruchten 1995). These are important to be represented in distributed real-time systems. The Time View is applied to represent an incomplete feature in UML diagrams: the time. The time constraints are the main concern in distributed real-time systems. The Resource Modeling View represents a description of resources. The definition of resources are important to deal with concurrency, distributed elements, and parallelism. The last view, the Allocation Modeling View, represents the allocation of functional application elements onto the available resources. In many cases, the resources are not available; thus, a specific mechanism has to manage the allocation of resources.

The introduction of the MARTE profile to each one of these views is described in the following subsections.

3.1 Process View

The Process View must treat events, concurrency, and synchronization aspects of the design. The CoreElements package defines three main stereotypes in the context of Process View. The Configuration stereotype is a set of active system elements, the Mode stereotype identifies an operational segment within the system, working in a given mode may imply that a set of system entities are active, and the ModeBehavior stereotype specifies a mutually exclusive set of modes.

The NFP package defines the stereotype NfpConstraint which supports textual expressions to specify assertions regarding, for instance, performance and scheduling. It also include their relationship to other features by means of variables, mathematical, logical, and time expressions. Besides, the VSL package provides stereotypes to represent quantities (TupleType, ChoiceType, CollectionType) and intervals of values (IntervalType) provided in NFP package.

The GCM package provides concepts for modeling message ports and extended ports. Some of the most important stereotypes are the DataPool stereotype, which is used to specify the storing policy, the FlowPort stereotype, which specifies the nature of the flow, the ClientServerPort stereotype, which provides a mechanism for specifying provided and required behavioral features, and the DataEvents stereotype, which treats the received messages.

The Alloc package defines some constructions related to real-time systems. The most important constructions are the stereotypes. Assign is a stereotype for associating elements from a logical context (application model elements), to named elements described in a more physical context (execution platform model elements). NfpRefine is a stereotype for associating one abstract model element to refined model elements.

The HLAM package defines diagrams and stereotypes for concurrency treatment specifically for issues about concurrent systems. In order to handle this feature, the MARTE profile uses the RtUnit stereotype. MARTE also defines stereotypes to concurrency types (CallConcurrencyKind, ConcurrencyKind, and SynchronizationKind).

Among the most important stereotypes of the GQAM package are the stereotypes related to communication (GaCommChannel and GaCommHost) and representation of resources (GaResourcesPlatform). The main stereotypes defined by SAM are used to analysis shared resources (SaSharedResource) and to measure schedulability, interrupt overheads, and utilization of scheduling processing (SaExecHost). The main stereotypes defined by PAM are PaRunTInstance, PaStep, and PaCommStep.

The concepts of time, resources, and resources allocation related to process view are described in the next sections: Time View, Resource Modeling View, and Resource Allocation View, respectively.

The DRM package defines several constructions related to real-time systems. The GRM is specialized by the DRM. It is divided into two sub-packages: SRM and HRM. The package SRM defines several stereotypes for software modeling. Among them are stereotypes for modeling interruptions, synchronize resources, policies to access a resource, and modeling concurrency resources.

The concepts of time related to process view are described in Time View (next section).

3.2 Time View

Even within UML 2.0 and the new time diagram, the notion of time is not clearly defined in UML. Within UML, the model of time is oversimplistic. As this is critical, the UML specification (Chap. 13, Superstructure) advices the use of a more sophisticated model of time for distributed systems (OMG 2010b). The MARTE profile presents time in a more precise and clear manner. Both discrete and continuous models of time are handled in MARTE. Clocks, which can be chronometric or logical, are used to access time structure. A chronometric clock implicitly refers to physical time, and a logical clock mainly addresses concrete instant ordering.

A clock is a finite or infinite set of instants and may represent a timed event and instants of its occurrences. A clock has a unit, and the instants can have a label. These instants in a clock are totally ordered for discrete time clocks; thus, they can be indexed by natural numbers. A time structure is composed of a set of clocks with the precedence relation between them. Precedence is a binary relation on clocks. From this relation, the following new relations can be derived: coincidence, strict precedence, independence, and exclusion.

MARTE describes extensions required to support the concepts of time. Some concepts result in new stereotypes, others specialize stereotypes defined for NFPs modeling, and still others need no extensions at all. The stereotypes ClockType and TimedDomain represent the concepts related to the time structure introduced in the Time domain view. The stereotypes TimedValueSpecification makes reference to Clocks. The Time Modeling package introduces two stereotypes specializing the NfpConstraint stereotype: TimedConstraint and ClockConstraint. The TimedEvent stereotype represents events whose occurrences are explicitly bound to clocks. And finally, the TimedProcessing stereotype represents activities that have known start and finish times or a known duration and whose instants and durations are explicitly bound to clocks.

3.3 Resource Modeling View

The central concept of GRM is the notion of a Resource. A Resource represents a physically or logically persistent entity that offers one or more ResourceServices. Resources and their services are the available means to perform the expected duties and/or satisfy the requirements for which the system under consideration should be designed.

The GRM package defines how the elements of the domain model extends metaclasses of the UML metamodel to represent resources. The most important definition for modeling resources in GRM is the stereotype Resource. It provides representation of generic resources from a holistic system wide perspective. The Resource stereotype aid represents mechanisms for connecting and delivering data, means to transport information from one location to another, concurrency of resources, liberation of resources, usage of the resource, scheduling policies, and different forms of memory.

3.4 Resource Allocation View

Allocation comprise both spatial distribution and temporal scheduling aspects in order to map various algorithmic operations onto available computing and communication resources and services. MARTE use the word allocation rather than deployment (as in UML) since deployment implies a physical distribution whereas allocation can also be logical. An example of allocation is the scheduling of processes in a processor (Mallet and Simone 2008). The allocation mechanism proposed by MARTE is very close to the structure allocations of SysML (the Systems Modeling Language OMG 2010c) because it allocates logical parts to more physical ones (André et al. 2007). However, MARTE makes it explicit that both the logical and the physical parts could be either of a behavioral or structural nature.

The Alloc package defines stereotypes for allocation modeling. Two of the most important stereotypes are Allocate and Allocated. Allocate is used to identify what can be allocated, the logical view (which can be behavior or structure), and what can serve as a target of an allocation, the physical view (which can be a resource or a service). Allocated applies to any named element that has at least one allocation relationship with another named element.

4 Conclusion

The MARTE profile was proposed to solve some UML issues related to the design of real-time systems. MARTE brings interesting notions to UML in order to model aspects such as resource allocation and nonfunctional requirements. Special attention was given to time constraints, in which three types of time representation are proposed: causal (concerned about instruction precedence), clocked (adds a notion of simultaneity), and physical (precise accurate modeling or real-time duration values).

In this chapter, a multiple-view architecture using the UML profile MARTE is proposed. Four main views were identified, and for each one, MARTE can be used as modeling language together with UML. For instance, the MARTE stereotype makes semantics of time more clear than it is within UML. The architecture is complementary to other architectures, not a replacement. The reason is that the focus is to describe the most important views regarding the design of distributed real-time systems. Therefore, the architecture should be combined with other multiple-view architecture models in order to describe other views not mentioned in this chapter, such as the use case view. As for future research, one idea is to jointly apply MARTE and SysML to a variety of distributed real-time systems in domains such as traffic control and manufacturing.

References

André C, Mallet F, de Simone R (2007) Modeling time(s). In: ACM-IEEE (ed) Proceedings of the 10th international conference on model driven engineering languages and systems (MODELS '07). Springer, Berlin/New York, pp 559–573

Baar T, Strohmeier A, Moreira AMD, Mellor SJ (eds) (2004) UML 2004 – the unified modelling language: modelling languages and applications. 7th international conference. Lecture notes in computer science, vol 3273. Springer, Berlin/New York

Belategi L, Sagardui G, Etxeberria L (2010) MARTE mechanisms to model variability when analyzing embedded software product lines. In: Bosch J, Lee J (eds) Software product lines: going beyond. Lecture notes in computer science, vol 6287. Springer, Berlin/Heidelberg, pp 466–470

Beneken G, Hammerschall U, Broy M, Cengarle MV, Jürjens J, Rumpe B, Schoenmakers M (2003) Componentware – state of the art 2003. In: Proceedings of the CUE workshop Venedig

Demathieu S, Thomas F, André C, Gérard S, Terrier F (2008) First experiments using the UML profile for MARTE. In: Proceedings of the 2008 11th IEEE symposium on object oriented real-time distributed computing, Orlando, pp 50–57

Dobing B, Parsons J (2006) How UML is used. Commun ACM 49(5):109–113

Elhaji M, Boulet P, Tourki R, Zitouni A, Dekeyser JL, Meftali S (2011) Modeling networks-on-chip at system level with the MARTE UML profile. In: M-BED'2011, Grenoble

Graf S, Ober I, Ober I (2006) A real-time profile for UML. Int J Softw Tools Technol Transf 8:113–127

Kruchten P (1995) The 4+1 view model of architecture. IEEE Softw 12(6):42–50

Maier MW, Emery D, Hilliard R (2004) ANSI-IEEE 1471 and systems engineering. Syst Eng 7:257–270

Mallet F, de Simone R (2008) MARTE: a profile for RT/E systems modeling, analysis–and simulation? In: Proceedings of the 1st international conference on simulation tools and techniques for communications, networks and systems, simutools '08, Marseille. pp 43:1–43:8

Medina JL, Harbour MG, Drake JM (2004) The UML profile for schedulability, performance and time; in the schedulability analysis and modeling of real-time distributed systems

Mraidha C, Tanguy Y, Jouvray C, Terrier F, Gerard S (2008) An execution framework for MARTE-based models. In: Proceedings of the 13th IEEE international conference on engineering of complex computer systems. IEEE Computer Society, Los Alamitos, pp 222–227

OMG (2005) UML profile for schedulability, performance, and time, version 1.1. Technical report formal/2005-01-02, OMG

OMG (2009) UML profile for MARTE: modeling and analysis of real-time embedded systems version. Technical report formal/2009-11-02, OMG

OMG (2010a) OMG unified modeling language (OMG UML) infrastructure, version 2.3. Technical report formal/2010-05-03, OMG

OMG (2010b) OMG unified modeling language (OMG UML) superstructure, version 2.3. Technical report formal/2010-05-03, OMG

OMG (2010c) Systems modeling language (SysML) – version 1.2

OMG (2010d) UML profile for MARTE: modeling and analysis of real-time embedded systems—version 1.1

Quadri IR, Yu H, Gamatié A, Meftali S, Dekeyser JL, Rutten É (2010) Targeting reconfigurable FPGA based SoCs using the MARTE UML profile: from high abstraction levels to code generation. Int J Embed Syst :18

Selic B, Gullekson G, Ward PT (1994) Real-time object-oriented modeling. Wiley, New York

Soares MS, Julia S, Vrancken J (2008) Real-time scheduling of batch systems using petri nets and linear logic. J Syst Softw 81(11):1983–1996

Soni D, Nord RL, Hofmeister C (1995) Software architecture in industrial applications. In: Proceedings of the 17th international conference on software engineering, ICSE '95, Seattle

Chapter 17
Applying Psychology to Facilitate Participation in Conceptual Modelling

Simon McGinnes

1 Introduction

Conceptual modelling skills are fundamental for information systems (IS) professionals and useful in the design of databases, programmes and user interfaces. Conceptual modelling techniques help developers and end users build a shared understanding of the application domain. Most conceptual modelling techniques use diagrams, which are thought "to facilitate communication between analysts and users" (Parsons and Cole 2005). However, content and rigour have historically been emphasised rather than ease of representation; "descriptions should be stated in a formalism with unambiguous syntax which can be understood by a suitable processor" (Loucopoulos and Zicari 1992).

The term *conceptual model* is used here to mean any model produced mainly to define mental concepts (as opposed to software structures, for example). The concepts in question relate to things that an IS needs to store data *about* and are defined as entity types or object classes in techniques such as entity-relationship (ER) and UML class diagrams. This excludes techniques not primarily intended for definition of mental concepts, such as the "conceptual models" of Soft Systems Methodology. The focus in this chapter is on conceptual models used for communication with, or produced by, individuals from the continuum of end users and other novice modellers. Unlike the diagrams that IT specialists construct for their own purposes, these models must be easily understandable by nonexperts. A substantial literature exists on the first type of model (e.g. Moody 2009), but relatively little attention has been paid to the use of conceptual models by nonexperts.

It has long been recognised that conventional conceptual model diagrams can be off-putting for nonexperts (Bansler and Bødker 1993). Even when models are

S. McGinnes
School of Computer Science and Statistics, Trinity College, Dublin, Ireland
e-mail: Simon.McGinnes@tcd.ie

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_17,

logically correct, they may still be misunderstood; the more formal a model becomes, the less useful it seems for communication. It may be unclear for whom models are being developed (Moody 2004). More understandable models could also be helpful for IT professionals, many of whom are weak modellers (Venable 1996). Modelling requires substantial expertise and is taxing for nonexperts, which limits their speed, accuracy and flexibility.

Today's modelling methods are descended from techniques that emerged in a very different technological landscape. Their "box-and-arrow" notations were intended for pencil-and-paper modelling, not the visual richness that software tools can provide. Although modelling has moved on, with advances such as domain-specific modelling and model-driven development, the same kinds of notation continue to be used. In any sphere, once-useful assumptions and methods can threaten effectiveness if there is a failure to adapt to new circumstances. We argue that conceptual modelling could usefully evolve by reducing the complexity of model representations. Errors are costly and it is helpful if end users can check models against their own knowledge, but this is impeded if models are couched in complex, cryptic notations. Arguably, the complexity of model representations has led to a perception that conceptual models are necessarily hard to understand. Alternative representations could make models more accessible to nonexperts who need to understand them or who seek to produce models themselves.

2 Psychological Analysis

A major goal of conceptual modelling is communication. To know how well communication is achieved, we must ask how models are perceived and understood. This is the realm of psychology, which can help us understand cognitive strengths, such as visual recognition, as well as the limitations of attention and short-term memory. Psychology elucidates the relationship between knowledge and mental models (Johnson-Laird 2005); it can help explain the demands of modelling and why it is difficult for nonexperts. Psychology has always been an important reference discipline for IS, but in conceptual modelling, it is pivotal (Ramesh et al. 1999); "real-world modeling practices can be informed by a deep understanding of cognitive facts" (Veres and Mansson 2005). Psychology tells us that, for a diagram to be understood, sensory perceptions must be associated with memories (Fig. 17.1). Diagram elements stimulate associative recall, creating conscious meaning. If elements are recognisable, then interpretation can be fast and unconscious. If they are less recognisable, interpretation must use a slower, conscious, sequential process which is cognitively intensive. This is why "decisions regarding the presentation of conceptual models are far from trivial and should be approached with as much care as decisions on their content" (Moody 2004).

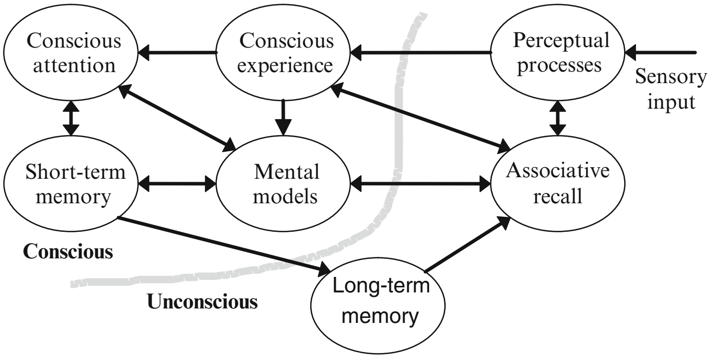


Fig. 17.1 Interaction of cognitive processes during perception of a model

2.1 Psychological Themes

Below we address several psychological themes in relation to conceptual modelling. Each suggests ways of making models easier to understand. The themes are derived from relatively uncontroversial psychological knowledge and represent distinct but related aspects; hence, some conclusions recur in different contexts.

2.1.1 Pre-attentive Processing

Pre-attentive processing is the automatic interpretation of sensory inputs, which allows information to be conveyed efficiently using variations in colour, shape and position (Schweizer 2001). It allows certain sensory data to be processed effortlessly, facilitating rapid visual searching; it is the mechanism that causes the single red word in a page of black text to "pop out" (Hershler and Hochstein 2005). Graphs exploit this cognitive ability; data can be easier to interpret when presented graphically than in text form (Fig. 17.2). Pre-attentive processing may be used only for single features, since varying features *in combination* force slower, conscious, sequential processing to be used.

The use of pre-attentive processing could make interpretation of conceptual models quicker and easier. However, conventional modelling techniques limit pre-attentive processing for nonexperts, because symbols are used without variation (e.g. boxes to denote entity types). To allow pre-attentive processing, visual elements should be varied such that nonexperts can distinguish them automatically and unconsciously. Examples of this might include use of recognisable iconic images, variation in colour or the use of positioning to convey information.

2.1.2 Mental Models

A mental model is our internal representation of a situation, providing a framework which helps us rationalise and predict. Mental models are thought to resemble the

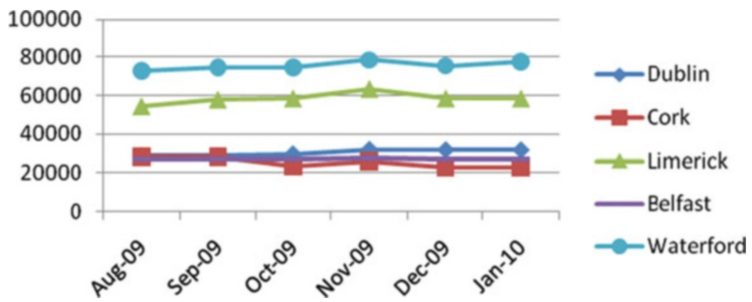


Fig. 17.2 Information presented in both textual and graphical form

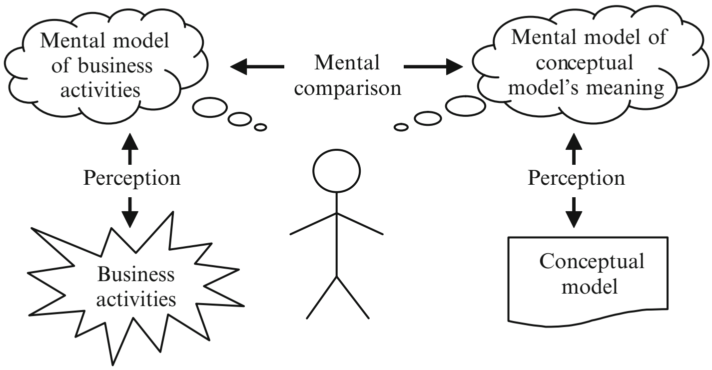


Fig. 17.3 Translating between conceptual and mental models

situations they represent; this is *isomorphism* (Johnson-Laird 2005). A conceptual model is, in some ways, merely a documented mental model; end users have mental models of their business, and the IS designer needs a similar understanding. The conceptual model is a way of helping the designer acquire a suitable mental model.

The conceptual model's role in this process means that end users must be able to compare conceptual models with their own mental models (Fig. 17.3). Ideally, rapid, at-a-glance comparison is needed. An expert can interpret models quickly and unconsciously, because the skill of interpretation has become internalised. But novices must translate each part of a model separately, in a slower, conscious process. As a result, complexity can easily overwhelm a nonexpert if the model looks different from their own conception of the situation. To illustrate the mental effort required, imagine a learner driver who remembers how to change gear only as a sequence of discrete actions. This kind of thinking involves a heavy cognitive load and has been likened to "using a helicopter to hold up a clothes line" (Sowa 1984).

Clearly, conventional models do not visually resemble the domains they describe; they are abstract statements of fact. The challenge therefore is to minimise cognitive load by making models more intuitively evocative of what is modelled. This might be done, for example, by using background images and choosing recognisable images to represent entity types. Research into the visual processing of cartoons shows that only a small portion of the information in an image is required for recognition, since the brain's perceptual system fills in details automatically. As in graphic design, a picture needs only to be rich enough to convey the essential information. Domain-specific modelling (DSM) offers some potential here, since models can be constructed from familiar domain concepts; however, quick recognition relies on consistent visual resemblance, which is unusual in DSM (Kelly and Tolvanen 2008).

2.1.3 Interaction of Sensory Modalities

In the nervous system, inputs from different senses are initially processed independently, but the brain's cognitive systems interact when cues of different types are present. This interaction can improve recognition. For example, combining images with words can be superior to the use of either modality alone (Medin and Rips 2005; Siau and Tan 2005). Conceptual modelling techniques can capitalise on this effect by augmenting text with recognisable images and using other sensory channels, if practicable (Chen 1999).

Meaning can also be reinforced by the positioning of items. Distinctive layout improves learning and can convey useful information rapidly, with minimal cognitive load. For example, proximity of elements automatically suggests a relationship between them (Koffka 1999). Consistent positioning helps associate elements and their meaning with specific locations; this is the "spatial arrangement mnemonic" (Bellezza 1983). Conceptual modelling techniques can exploit these cognitive mechanisms by consistently placing elements so that the viewer becomes accustomed to their locations; elements should therefore not change position unless absolutely necessary.

One logical consequence of this advice is that the practice of reorganising diagrams should be avoided. As models evolve, the use of lines to connect elements normally makes it necessary to move elements in order to avoid ugly "relationship spaghetti". If elements should not be moved, this implies that some notation other than lines should be used to denote relationships between elements.

2.1.4 Association and Reinforcement

Meaning is created when an observer receives sensory cues and recalls prior experience. For this to happen, the cues must have well-defined and distinct mental associations (in other words, they must be meaningful to the observer). For example, hearing the word "dog" may elicit the recall of mental images of dogs. Meaning is reinforced when each cue is consistently linked to a particular recalled experience. The effect is extinguished when the opposite occurs; when a given cue is associated ambiguously with different recalled experiences at different times.

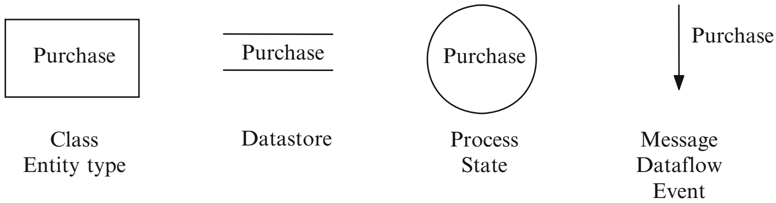


Fig. 17.4 Symbols for different aspects of the concept *purchase*

For end users, conventional modelling methods tend to extinguish associations because they use the same symbol to represent every end-user concept. For example, in an ER diagram, the entity types *purchase, customer* and *product* would all be represented using an identical symbol (a rectangle). The lack of a 1-1 correspondence between concept and symbol means that mental associations cannot be formed, which makes it more difficult to interpret models. As a result, even experts expend more cognitive effort than necessary in interpreting models (McGinnes and Amos 2001).

The logical conclusion is that each end-user concept (entity type) in a model should have a unique, meaningful symbol of its own (Moody 2009). For example, the entity types *purchase, customer* and *product* would each have a different symbol. This might be counterintuitive to IT professionals who are used to using the same symbol for all entity types. However, they should remember that, while *entity type* is a familiar concept for IT professionals, it is not for end users. Using the same symbol for every entity type is therefore unhelpful for end users.

Conventional IS methodologies tend to further complicate matters by representing end-user concepts with multiple symbols. Figure 17.4 shows aspects of the concept *purchase* as they might be depicted in various modelling techniques. Each symbol represents some perspective on *purchase* rather than the concept itself; several symbols have multiple meanings. This might be fine for an IT specialist but, from perceptual and cognitive perspectives, it is a poor choice for end users. Instead, the end-user concept *purchase* should consistently be depicted using a single, unique symbol, even if different perspectives are intended. As before, IT specialists should be aware that the differences between system concepts like *entity type*, *process* and *state* are important and obvious to IT professionals, but rarely apparent or meaningful to end users. On the other hand, an end user would view purchases as entirely different in kind from customers and would therefore not expect the two concepts to be represented using the same symbol.

2.1.5 Short-Term Memory and Chunking

Attention and short-term memory have limited capacity and are easily overloaded; an individual may attend consciously to only a few items at a time (Bannert 2002). The nervous system filters and manipulates perceptions, limiting the need to focus

Table 17.1 Chunking strategies offered by popular modelling techniques

Technique	Summarising	Partitioning	Filtering
UML class diagrams	Inheritance	Aggregation; package diagrams	None
ER diagrams	Supertypes	None	None
Dataflow diagrams	None	Decomposition	None
State diagrams	Nested states	None	None

simultaneously on many items. For example, items which are close or visually similar are mentally grouped; each group may then be attended to individually. This process of *chunking* happens automatically and unconsciously (Gobet 2005).

Chunking creates hierarchies, since each idea can contain other ideas and so on. For example, an international phone number naturally forms a three-level hierarchy: the international code (international access code and country code), the city or area prefix and the remainder (exchange and subscriber numbers). Structuring ideas in a hierarchy is thought to mimic the way experts organise their own knowledge and can assist in comprehension and recall. If a "chunk" is linked to existing knowledge (in long-term memory), this can also help since it avoids the need to hold details about that chunk in short-term memory.

The main chunking strategies are summarising, partitioning and filtering, and these can operate at both perceptual and conceptual levels. We can hear a sound as a single entity or attend separately to its components. Similarly, we can think about a concept as a single entity or as several ideas. Like "divide and conquer" problem-solving, chunking can make a complicated situation more tractable, but it can also result in poorly integrated, partial solutions. Therefore, it is important to have a way of recreating a holistic view of the situation when chunking is used.

In conceptual modelling, there is a need to hide or reduce complexity (Moody 2004), and chunking offers possible solutions. Models routinely contain hundreds of items and can easily overwhelm nonexperts. But popular modelling techniques offer only limited support for chunking (Table 17.1). The *summarising* strategies (all forms of generalisation) are useful, but do not simplify the model's appearance and cannot be applied dynamically; they require the model's structure to be altered. This limits their utility since structural changes require effort to make and may have unwanted implications when a model is used later. The formal *partitioning* strategies (aggregation and decomposition) also require structural change and therefore have similar disadvantages. No native *filtering* strategies are offered at all.

In conceptual modelling, a popular alternative is to partition models informally into subsets. This can help make a large model more accessible, but it also has disadvantages. First, it is not natively supported by modelling techniques and so must be handled "manually"; managing the parallel evolution and reintegration of model subsets can be difficult (Ramesh and Dennis 2002). Also, the notations lack any overview of model subsets, so it is difficult for nonexperts to gain a holistic view. Finally, because lines are used to denote relationships, off-page connectors are usually required. This undermines chunking for nonexperts because each subset's meaning can be obtained only by consulting the meaning of linked subsets, which the nonexpert is unlikely to be able to retain mentally.

The challenge for conceptual modelling techniques is to exploit chunking to make models easier to understand. Modellers should be able to employ the full range of strategies flexibly, during modelling, without needing to structurally reorganise models. Various methods are possible. Where "chunks" of a model can be linked to basic-level concepts already familiar to the observer, such as people, organisations and locations, this can help reduce the cognitive effort required to interpret the model. *Summarising* can be achieved by showing only the main concepts in a model (where the user can control what "main" means) or by hiding nested concepts. *Partitioning* can be achieved by allowing model subsets to be displayed understandably without needing off-page connectors. *Filtering* can be achieved by allowing the user to select what kind of concepts to display; for example, a user might choose to show only entity types that correspond to activities.

2.1.6 Concept Instability and Evolution (and Group Consensus)

Mental categories seem to be fuzzy. We might believe we understand a category but struggle to state rules for category membership. Some things seem more typical of their categories than others and some properties more important (Burek 2005). Categories may be defined in different ways: intensionally, extensionally or in terms of exemplars. Context also has an influence; the tendency to classify things differently in different contexts is termed *concept instability*.

The connectionist view suggests that abstract categories are not a primary mechanism in cognition. Instead, the perceptual system matches patterns; one thing evokes another on the basis of similarity, without the need for internal representations. Artificial neural networks demonstrate this property; they can categorise following exposure to known patterns, without rules or conceptual structures. According to this view, abstract categories are conscious, verbal phenomena which emerge only when we try to define or reason about them. On the other hand, a small number of concrete, basic-level categories may be innate, including people, activities, places and so on (Kapros and McGinnes 2011). Couching models in terms of such basic-level categories may therefore be helpful.

All of this is problematic for conceptual modelling techniques, which implicitly assume that end users' business concepts are pre-existing and crisp. It is as if concepts are brought fully formed to the modelling exercise and need merely to be documented. Experienced modellers recognise that idea as unrealistic, because end users' understanding of their own concepts often evolves substantially during modelling, which is generally a learning experience for all concerned. Yet, modelling techniques do not provide support for the evolution of understanding; it is difficult to explore ideas by continually restructuring a model.

One way that modelling techniques can support exploration is by tolerating incompleteness and incorrectness, so that models can be captured even if elements are partially or badly defined. Another is by reducing the effort required to change models. The challenge is to allow modellers to refine models at their own pace and without undue effort, until a satisfactory state is reached. This advice may seem irrelevant to IT professionals, since experienced modellers make few changes to models; they typically model rapidly and accurately. However, when novices participate in modelling, frequent revision is usually necessary and the cost of change is then much more important (McGinnes 2000).

Ease of change and error tolerance are important for another reason; most models are consensus views that combine the inputs of many individuals. Effective groups seek compromises rather than optimal solutions and are most productive and creative in the absence of judgement and editing. Participants are more likely to contribute if the aim is exploration and learning rather than unanimity (Zander 1994). In modelling, reconciling the viewpoints of different end users can require substantial effort, especially at early stages when alternatives must be explored. Experimentation is usually necessary. When the modelling technique hinders experimentation and demands rigour, this prevents the tolerance and lack of editing that the group needs to function well. The challenge is to actively support the informal process of consensus building. Modellers should be able to experiment during this process, not forced to address syntactical and semantic correctness too early.

2.1.7 Progressive Learning

Another way in which nonexpert modellers learn is by progressively coming to understand the modelling technique, in an iterative process, by making mistakes (McGinnes 2000; Leung and Bolloju 2005). This takes both time and cognitive effort (Moody 2004; Venable 1996). While experts can readily fix their models, nonexperts find it much more difficult; lacking a suitable mental framework, they fail to recognise errors quickly or easily (McGinnes 2000).

The challenge for modelling techniques is to support iterative learning. One way is for the modelling tool to highlight errors and assist in their resolution. Syntactic errors can easily be dealt with this way. If enough intelligence is built in, this can extend to pointing out possible semantic errors (Kapros and McGinnes 2011). But perhaps more powerful is to nudge modellers in a helpful direction. Intelligent defaulting can help modellers make good choices (McGinnes and Amos 2001). Errors can also be reduced by removing the potential for inconsistency, since redundancy in a model is typically confusing and the resulting inconsistencies often go unresolved (McGinnes 2000). Conventional techniques offer opportunities for this type of error. For example, in ER modelling, a relationship is equivalent to the presence of a foreign key; novices may therefore define foreign key attributes that contradict relationships. If the notation made it impossible to model inconsistently, this type of error would be avoided.

2.1.8 Attention, Arousal, Personal Relevance and Set

Arousal is the level of physiological activity, correlated roughly to the degree of alertness. Arousal is necessary for attention; too little or too much is counterproductive. It is increased by stimulation from novel events, movement, loud sounds, bright lights and personal responsibility. It is decreased by monotonous tasks, repetitive stimuli and a lack of personal responsibility, which can lead to social loafing. In conceptual modelling, participants need to maintain attention for sustained periods. But modelling normally uses static, monochrome visuals, resulting in diminished attention, concentration and performance. It is easy to become bored and distracted, especially in group work where individual responsibility is diminished.

The modeller can help sustain attention by using colourful or animated model representations, introducing novel events into modelling sessions and initiating periodic changes in task and posture. Placing responsibility on individual group members can reduce the risk of social loafing. Also, since personally relevant information is more interesting and retained better than other information, attention can be supported by focusing on subject matter in which participants have a particular interest (or by choosing participants to suit the subject matter).

Set is a general term referring to a predisposition that leads one to view a situation in a particular way. Sets can strongly affect the outcome of cognitive tasks such as problem-solving and system design (Adams and Avison 2003). For example, users may view models (as well as requirements and designs) as "technical" and more relevant to the designer's work than to their own (Bansler and Bødker 1993). The result can be confusion; system designers persist with incorrect models, in the mistaken belief that they reflect the end user's reality, while users go along with models they understand poorly (Topi and Ramesh 2002).

This type of set may be reduced by increasing the model's relevance for the end user and avoiding technical jargon in favour of the user's own concepts and language. Rather than abstract, technical concepts such as *class, entity* and *property*, it is better to use concrete categories familiar to the user such as *person, document* and *place*. Also, concrete feedback can be provided (e.g. in the form of prototypes derived from models), helping users to appreciate the consequences of modelling decisions. The overall aim is to help end users develop a sense of ownership in their models. However, this may require a change in set by IT professionals, who may see modelling as a technical process that they must regulate (Bansler and Bødker 1993). They would need to view modelling instead as more of a user-controlled process.

Another relevant mental set is *set of function*. This set prevents us from recognising the tools needed to solve a problem if they must be used in nonobvious ways. In conceptual modelling, experts learn modelling clichés to represent complex situations, whereas novice modellers lack knowledge of these modelling "tricks". Education can help, but in-depth training may be impractical. The challenge is therefore to redesign the modelling task to be more intuitively accessible and so that solutions to modelling problems are more obvious to nonexperts.

Table 17.2 Evaluation of ER modelling (Information Engineering notation)

Theme/guideline	To what extent is guideline observed?[a]
Pre-attentive processing. Allow rapid, unconscious interpretation of models	(3) Prevented by the use of identical symbols for entity types
Mental models. Avoid mental translation by making the model resemble the domain	(3) Diagram does not resemble the domain; it is an abstract statement of fact
Interaction of sensory modalities. Facilitate recognition by combining text and images and by positioning items consistently	(3) Symbols do not match text (e.g. “person” shown as rectangle, not as person); use of line connectors causes items to move
Association and reinforcement. Support association by representing each end-user concept with one and only one symbol.	(3) Association is prevented as symbols are reused (e.g. “invoice” and “employee” have different meanings but identical symbols)
Short-term memory and chunking. Offer dynamic summarising, filtering and partitioning, and link to familiar concepts	(2) No dynamic chunking; no filtering; use of informal subsets is onerous; entity types not linked to concepts familiar to end user
Concept instability and evolution (*and group consensus*). Allow models to be restructured easily and quickly	(2) Structural change is onerous, due particularly to use of lines; this makes experimentation impractical (depends partly on tool)
Progressive learning. Avoid preventable errors; tolerate and assist in resolving others; nudge modeller towards good solutions	(2) Errors rarely tolerated (depends on tool); notation allows inconsistency; unable to detect semantic errors or suggest good choices
Attention, arousal, personal relevance and set. Vary task, colour, etc.; give users control; avoid IT terms; give feedback	(2) Monochrome notation, little variety; high skill requirement prevents end-user control; IT jargon used (entity, relationship, etc.)

[a]1 = guideline is strongly observed, 2 = partially observed, 3 = not observed

3 Conclusions and Further Research

This chapter has introduced several psychological themes which can guide the representation of models and the modelling process. The focus has been on understanding and communication for nonexperts. Matters of technical content have been ignored; they are addressed comprehensively in the literature, and it is assumed that changes in representation or process are made without detriment to technical content. Table 17.2 illustrates application of the guidelines to analyse ER modelling. The results suggest that alternative representations for ER models, with appropriate tool support, could facilitate participation by nonexperts.

The guidelines, and others derived from psychological thinking, have been used to formulate alternative model representations and supporting tools (Kapros and McGinnes 2011). In experiments, the use of these alternative representations and tools was associated with substantially improved model quality, in models produced by end users and novice modellers, and with increased productivity for expert and nonexpert modellers (McGinnes and Amos 2001). The relative contributions to these results of individual guidelines and tool functions are unclear, and further research is planned to test specific aspects.

References

Adams C, Avison D (2003) Dangers inherent in the use of techniques: identifying framing influences. Inf Technol People 16:203–234

Bannert M (2002) Managing cognitive load-recent trends in cognitive load theory. Learn Instr 12:139–146

Bansler JP, Bødker K (1993) A reappraisal of structured analysis: design in an organizational context. ACM Trans Inf Syst (TOIS) 11:165–193

Bellezza FS (1983) The spatial-arrangement mnemonic. J Educ Psychol 75:830–837

Burek P (2005) Essentialized conceptual structures in ontology modeling. Springer, Berlin, pp 880–886

Chen P (1999) From ancient Egyptian language to future conceptual modeling. Concept Model: Lect Notes Comput Sci 1565:56–64

Gobet F (2005) Chunking models of expertise: implications for education. Appl Cogn Psychol 19:183–204

Hershler O, Hochstein S (2005) At first sight: a high-level pop out effect for faces. Vis Res 45:1707–1724

Johnson-Laird P (2005) Mental models and thought. In: Holyoak K, Morrison RG (eds) The Cambridge handbook of thinking and reasoning. Cambridge University Press, New York, pp 185–208

Kapros E, McGinnes S (2011) Exploring conceptual data independence with a prototype adaptive information system. CAiSE 2011, London

Kelly S, Tolvanen JP (2008) Domain-specific modeling: enabling full code generation. Wiley, Hoboken

Koffka K (1999) Principles of gestalt psychology. Psychology Press

Leung F, Bolloju N (2005) Analyzing the quality of domain models developed by novice systems analysts

Loucopoulos P, Zicari R (1992) Conceptual modeling, databases, and CASE: an integrated view of information systems development. Wiley, New York

McGinnes S (2000) Conceptual modelling: a psychological perspective. Ph.D. thesis, Department of Information Systems, London School of Economics, University of London

McGinnes S, Amos J (2001) Accelerated business concept modeling: combining user interface design with object modeling. In: Harmelen MV, Wilson S (eds) Object modeling and user interface design: designing interactive systems. Addison-Wesley, Boston, pp 3–36

Medin DL, Rips LJ (2005) Concepts and categories: memory, meaning, and metaphysics. In: Holyoak K, Morrison RG (eds) The Cambridge handbook of thinking and reasoning. Cambridge University Press, New York, pp 37–72

Moody DL (2004) Cognitive load effects on end user understanding of conceptual models: an experimental analysis. In: Benczúr A, Demetrovics J, Gottlob G (eds) ADBIS 2004, vol LNCS 3255. Springer, Heidelberg, pp 129–143

Moody DL (2009) The "physics" of notations: toward a scientific basis for constructing visual notations in software engineering. IEEE Trans Softw Eng 35:756–779

Parsons J, Cole L (2005) What do the pictures mean? Guidelines for experimental evaluation of representation fidelity in diagrammatical conceptual modeling techniques. Data Knowl Eng 55:327–342

Ramesh V, Dennis AR (2002) The object-oriented team: lessons for virtual teams from global software development. IEEE, pp 212–221

Ramesh V, Parsons J, Browne G (1999) What is the role of cognition in conceptual modeling? A report on the first workshop on cognition and conceptual modeling. Concept Model:272–280

Schweizer K (2001) Preattentive processing and cognitive ability. Intelligence 29:169–186

Siau K, Tan X (2005) Improving the quality of conceptual modeling using cognitive mapping techniques. Data Knowl Eng 55:343–365

Sowa JF (1984) Conceptual structures. Addison-Wesley, Reading

Topi H, Ramesh V (2002) Human factors research on data modeling: a review of prior research, an extended framework and future research directions. J Database Manage 13:3–19
Venable JR (1996) Teaching novice conceptual data modellers to become experts. In: Proceedings of the international conference on Software engineering: education and practice, pp 50–56
Veres C, Mansson G (2005) Cognition and modeling: foundations for research and practice. J Inf Technol Theor Appl 7:93
Zander A (1994) Making groups effective. Jossey Bass, San Francisco, pp 185–236

Chapter 18
The Diffusion of Agile Software Development: Insights from a Regional Survey

David Bustard, George Wilkie, and Desmond Greer

1 Introduction

The social science literature contains a substantial body of work on *innovation diffusion* (Rogers 2003; Mahajan and Peterson 1985; Moore and Benbasat 1991), which is the process by which new ideas become established. The pattern of adoption is typically summarized by two curves (Fig. 18.1), one showing market share growing to saturation (*S*-curve) and the other classifying the type of adopters involved in each phase (bell-shaped Gaussian curve): *innovators*, *early adopters*, *early majority*, *late majority,* and *laggards*. Further, in considering technology innovation, such as agile development, Moore argued that for adoption to occur, a transition has to be made from the early adopters who explore new technology routinely to the early majority who wait pragmatically until there are clear advantages to adoption. This transition he calls "crossing the chasm" (Moore 1999).

In estimating progress along the adoption curve, it seems safe to suggest that agile development is now comfortably on the other side the chasm. Arguably, this began 10 years ago, with the collaboration that led to the "agile" concept, the development of the Agile Manifesto, and the definition of the associated 12 Agile Principles (Beck et al. 2001). The growth of agile use has been substantial since then, but estimates of adoption levels remain surprisingly low. Vijayasarathy and Turk, for example, describe their survey in 2008 (Vijayasarathy and Turk 2008) as a contribution to "bridging the chasm." Also, while Ambler has been asserting that the chasm has been crossed since 2006 (Ambler 2011), his conclusions about the current extent of adoption seem modest (Ambler 2010a). The underlying issue here

D. Bustard (✉)
University of Ulster, Coleraine BT52 1SA, UK

G. Wilkie
University of Ulster, Jordanstown BT37 0QB, UK

D. Greer
Queen's University, Belfast BT7 1NN, UK

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_18,

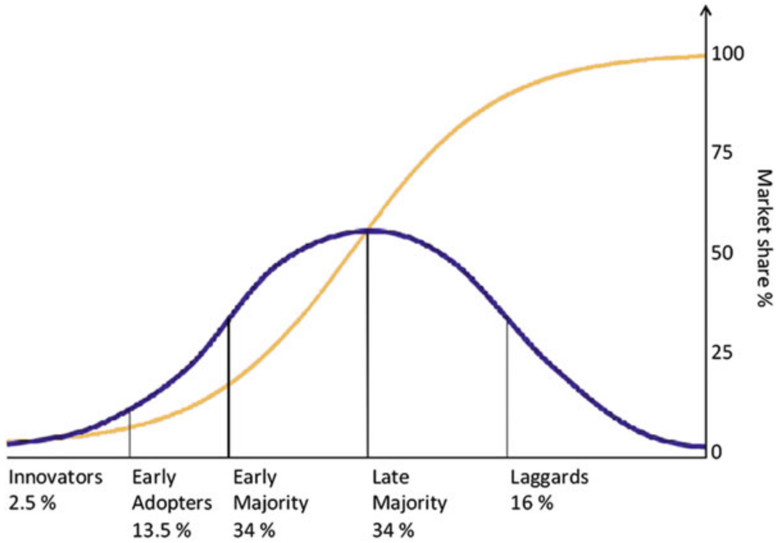


Fig. 18.1 Innovation diffusion curves

is that most surveys are based on self-selecting participants, so the results cannot easily be generalized to the full population of software development organizations. For example, Vijayasarathy and Turk note that their respondents were "knowledgeable and/or interested in agile development … more representative of early adopters and not of all developers."

The purpose of this chapter is to present the results from an agile survey that is distinctive in attempting to provide comprehensive coverage of a single geographic region, rather than using volunteers from a wider area. The survey was commissioned by Invest Northern Ireland (INI), an economic development agency, to analyze the capability and capacity of the Northern Ireland software industry to engage in agile software development. In addition to clearer adoption information, the survey was also expected to provide insights into the current practice of agile development. The next two sections of this chapter describe the design of the survey and make observations on the results obtained. This chapter concludes with some general comments on agile diffusion and suggestions for future work.

2 Survey Design

Pfleeger and Kitchenham (2002) define a survey as "a comprehensive system for collecting information to describe, compare or explain knowledge, attitudes and behavior." The aims of the Northern Ireland survey were to (1) *describe* the extent of agile software development among organizations in the region, (2) *compare* the position in Northern Ireland with other parts of the world, and (3) *explain* any apparent anomalies in the findings that emerged.

The first step was to identify the companies that might participate. The goal, as indicated, was to have comprehensive coverage of the region. This was achieved by building the survey list around the software development companies that had either engaged with INI (the economic development agency) or who had joined Momentum, the trade association for the ICT industry in Northern Ireland. Fifty-six companies were identified in this way.

The next stage was to decide how the survey would be performed. Given that there were three researchers involved (the authors) and the companies to be surveyed had a relatively narrow geographical spread (many in the larger Belfast area), it was a straightforward decision to gather information face-to-face, through structured interviews. The direct contact approach would also make the size and wording of the associated questionnaire less critical and help achieve (1) a high response rate, (2) the full completion of each questionnaire, and (3) consistency in the interpretation of each question.

A number of research questions influenced the design of the questionnaire. The first was the overall driver for the study, namely, *what is the capability and capacity of the Northern Ireland software industry to engage in agile software development*? A set of related secondary questions included are as follows:

1. *What does it mean to adopt an agile approach to software development* (to clarify what is expected in the software process and detect any evolving changes in the concepts and practices involved)?
2. *To what extent is agile adoption in Northern Ireland indicative of agile adoption worldwide* (to assess the significance and generality of the results)?
3. *What are the main inhibitors and accelerators for agile adoption* (to assess likely changes in adoption levels)?
4. *To what extent does the agile approach help or hinder individual aspects of the software development process* (to clarify the specific contributions of the agile approach to the development process, identifying any current inhibitors)?
5. *In what ways is the quality of the software product affected by an agile approach* (to clarify the specific contributions of the agile approach to software quality, identifying any current inhibitors)?

The resulting questionnaire had two main sections: one gathering summary information about the organization and the other capturing details of its experience with agile software development. The questions on the organization asked how long the company had been in operation, its business area (including types of software produced), its autonomy (independent or part of a larger organization), its size (technical staff and overall numbers), and any conformance to external standards, such as ISO 9001 or CMMI (Chrissis et al. 2011).

The number of technical questions posed depended on the agile experience of the organization. Four categories were defined: (A) *no direct experience so far*, (B) *some experimental use*, (C) *routine use for some projects*, and (D) *general approach taken across the organization*. For category A companies, with no direct experience of agile development, it was simply a matter of documenting the reasons for noninvolvement, with some prompts offered, such as (1) satisfaction with current practices, (2) no one having sufficient interest, and (3) preexisting negative impressions of the approach. The third response was followed up with additional questions to clarify the issues involved. Similarly, for category B organizations, with just experimental experience of agile development, the focus was simply on determining details of the experiments and their results.

Twenty questions were posed to those companies in category C or D, which used agile development routinely. The first 12 questions established when agile use began; why and how it was introduced; the extent of its use; involvement in specific methodologies (e.g., Scrum (Schwaber and Beedle 2008), XP (Shore and Warden 2007), DSDM (Tudor and Tudor 2010)); selection of team members, team sizes, and locations; and the number of projects completed so far. More detailed questions then followed. These covered the following:

- An assessment of the advantage or disadvantage in using agile development across over 30 individual software engineering activities, including aspects of requirements analysis, system design, project management, and the satisfaction of developers, clients, and managers
- The perceived impact of agile development on the software attributes cited in the ISO 9126 quality model (functionality, reliability, usability, efficiency, maintainability, and portability) (ISO/IEC 9126-1 2001)
- The level of agreement or disagreement with the 12 Principles of Agile Development (e.g., 7. *Working software is the primary measure of progress*) (Beck et al. 2001)
- The level of experience, if any, with the 12 practices of XP (Shore and Warden 2007)
- Details of any specific tools used in the support of agile development

As is evident from the description above, a substantial amount of information was gathered at each interview. It was estimated that for a company involved in agile development, completing the questionnaire would take roughly an hour; for others, it would be much shorter. It was recognized that having a questionnaire of this length would be daunting for some organizations, which could lead to nonparticipation. This seemed particularly likely for organizations that had yet to consider the agile approach in detail. For those actively engaged in agile development, there was some evidence from earlier informal interactions that companies would be relatively enthusiastic about presenting what they had achieved. Additional incentives included are as follows:

1. Emphasizing the confidentiality of the study: “Please note that all information supplied will be held in accordance with the data protection act 1998. The information will be used and published in the form of consolidated statistical survey data only.”

2. Making clear that although INI was interested in the detailed responses from each company, the front page of the questionnaire allowed for an opt-out from this arrangement.
3. Offering those participating in the survey a full copy of the resulting consultancy report, which would not be made public for at least a year.

To help ensure that there was sufficient reflection on the answers provided, it was decided to send out the questionnaire (electronically) in advance of an interview, with a request to draft responses as far as possible. At the interview, the answers would then be confirmed, additional notes taken, and any missing sections completed.

As a check on the suitability of the questionnaire, it was piloted with three companies with whom the authors had worked previously. No significant changes were required.

3 Survey Implementation, Results, and Observations

This section describes the implementation of the survey in more detail and comments on both the process involved and the results obtained.

3.1 Survey Implementation

The survey was carried out over a 4-month period, leading up to the production of a consultancy report in March 2010. To give authority to the survey, the initial invitation to participate was issued by INI, using email directed at the CEO, CIO, or senior representative in each organization. Participation was, however, optional. About half of the 56 companies contacted agreed to be interviewed. Others made it clear that they did not want to take part and some simply did not reply. For the latter group, further attempts were made to establish contact. In some cases, there had been organizational changes and a new contact had to be found. A small number of organizations agreed to complete the questionnaire by telephone and two returned the questionnaire without an interview or discussion. One of the companies that failed to respond had moved its main software development activity out of Northern Ireland so they were excluded from the study. Another large public body argued that they developed all significant systems through third parties and so should not be included in the survey. This was agreed, though later it was recognized that such bodies have a significant impact on agile adoption in those companies where they place contracts because, typically, the projects are fixed term and specify project management processes (usually Prince2)—both of which tend to inhibit agile use.

Table 18.1 Broad levels of experience in agile use

Response	No. of companies
A: No direct experience so far	6
B: Some experimental use	7
C: Routine use for some projects	15
D: General approach taken across the organization	9

3.2 Survey Results

In the end, 37 questionnaires were completed in a form suitable for analysis. The companies contributing ranged in size from 7 to 770 development staff and covered a wide range of business areas, including aerospace, finance, telecommunications, oil and gas, and government. One characteristic of the region is that it has a relatively large proportion of SMEs, with about two thirds having fewer than 50 staff.

The broad level of experience with agile development reported by the 37 companies surveyed is summarized in Table 18.1, using the four categories explained in Sect. 2.

The six companies in category A had no experience of agile techniques. In each case, the company was satisfied with its current development approach and had no plans to change in the foreseeable future. One of these companies felt that their current approach to requirement development, approval, and change was better suited to the fixed-price contract work in which they were involved; also, they did not find it necessary to demonstrate the "product" to their customers on a regular basis in order to show project progress. Two of these companies did employ prototyping: one to help clarify requirements and the other to aid proof of concept within the development team.

Across the seven companies in category B that had experimented with agile techniques, the main reasons given for considering an agile approach were as follows (number of responses shown in brackets):

- Client requests to integrate with their business/project processes (2)
- Issues with requirements (2), either to reduce the pressure of voluminous detailed requirements up-front or be better able to cope with an anticipated large volume of change requests
- To support frequent releases to help manage the product strategy (product roadmap) (1)
- Reaction to the heavyweight traditional approaches and desire to operate more efficiently (2)

Four of these category B companies had started experimenting with agile techniques during late 2008 or 2009, while for the other three, the experiments were conducted much earlier in the period 2000–2005. In all seven cases, routine or widespread adoption of agile approaches had not yet happened. For two of the companies, the main reason given was that as their customers became more engaged, so their ability to change and extend their requirements increased in often undesirable ways. Interestingly, one of the fully adopting companies found it difficult to experiment with agile techniques on a small scale, arguing that a whole-hearted commitment and high level of discipline was required to realize the main benefits from the approach, necessitating a more complete adoption strategy.

For category C and D companies, reporting routine use of agile techniques, the main reasons cited for adoption were as follows:

- Driven by developer interest (8).
- Influence from a partner/parent/sister organization and in some cases to facilitate better interaction between these co-operating groups (6).
- For competitive advantage, such as a need to meet tight deadlines or provide frequent releases—some companies could see competitors using agile techniques able to release new features in a much shorter timeframe (6).
- To improve requirement development (2).
- Client demand (2).

The main factor was "developer interest," which has been a significant influence in the rise of the agile movement. The role of large companies is also important in spreading the approach globally because they influence competing organizations and subcontractors, as well as their own individual sites.

The nature of the client contract appears to have an important bearing on whether or not an agile approach is used. Many companies (at least 5) in category C used an agile approach if the client contract permitted it. In some cases, even on fixed-price contracts, an agile approach was taken, with internal iterative development but no release of intermediate products.

From the 24 companies using agile development routinely (categories C and D), 11 were using Scrum, 8 a combination of Scrum with XP, and the remaining 5, various mixtures of Scrum, RUP, and DSDM. Overall, Scrum emerged clearly as the dominant framework for the agile approach.

The level of experience with agile development varied significantly across those using it routinely. Two companies had taken an agile approach in one form or another for over 10 years, but over a third of the respondents were still in the first 18 months of routine use. Reflecting that range, roughly 10 % of companies had completed over 50 agile projects while 40 % had 5 or fewer.

Project teams ranged in size from 2 to 30 developers, with an average size of around 5 to 6. Ten companies reported development teams located in the same room, while 14 had some teams that were separated within a building or distributed geographically across different regions of the world.

To assess the effectiveness of the agile process, each company was asked to rate, on a scale of −2 (much worse) to +2 (much improved), various process-related aspects of their agile approach against previous experience with non-agile development. The questions related to activities with requirements, design, coding, testing, maintenance, and overall developer productivity. The results, summed across all 24 respondents (value range −48 to +48), are presented in Fig. 18.2.

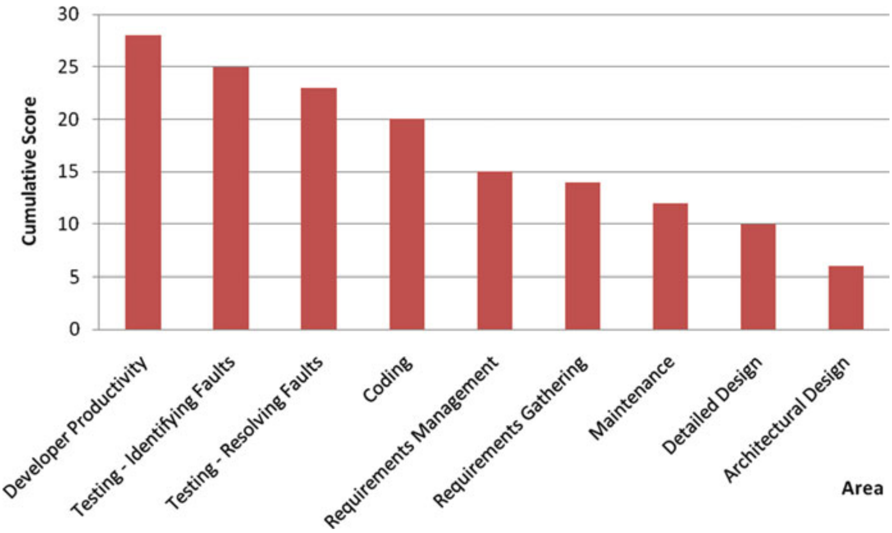


Fig. 18.2 Perceived process benefits of agile approach

It is significant to note that all of the cumulative totals are positive, meaning that agile approaches are considered an improvement over non-agile approaches in all of process areas assessed. These benefits were shared by developers, customers, and management. The highest ranked aspect is developer productivity, but agile approaches also assist testing, particularly through test-driven development. Several companies commented that this had been their area of greatest gain.

The lowest-ranking activities related to design—particularly architectural design, where 13 companies felt that there was no improvement over traditional approaches, with some reporting that teams found design more difficult.

A similar pattern emerged in the assessment of perceived improvement to the software product through agile development, as summarized in Fig. 18.3. As might be expected, product functionality and usability are the two qualities most often enhanced through the use of highly iterative agile approaches, where the focus is very much on the value of the developed product.

3.3 Observations

This section presents a number of observations on the survey results in relation to the general question of the level of agile adoption in Northern Ireland.

3.3.1 What It Means to Be "Agile"

One issue in designing a survey to assess agile adoption is to decide what the term means, as no agreed definition has yet emerged (Abrahamsson et al. 2009). Ambler,

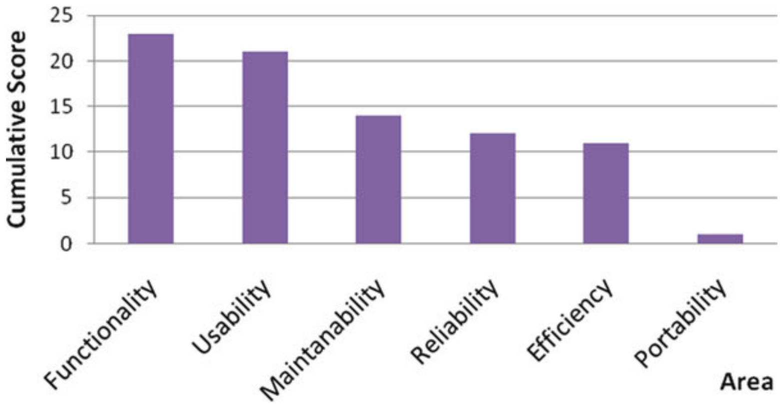


Fig. 18.3 Perceived product benefits of agile approach

for example, recently proposed five criteria (Ambler 2010b) for an agile team to be classified as "agile," but these are debatable and insufficiently precise for use in an empirical study:

1. Produce working software on a regular basis.
2. Do continuous regression testing (and better yet, take a test-driven development approach).
3. Work closely with their stakeholders, ideally on a daily basis.
4. Be self-organizing within an appropriate governance framework.
5. Regularly reflect on how they work together and then act to improve on their findings.

Because of this difficulty, it was decided that rather than produce a definition in advance, information would be gathered in the survey to help create it. This involved seeking each company's views of the standard principles of agile development and noting their associated activities in pursuit of those principles, prompted by the practices defined for XP. In this way, the survey could contribute to an understanding of what it means to be "agile" and, at the same time, propose a practical definition to use in the interpretation of the data.

The results, however, suggest that this level of concern was unnecessary. The vast majority of the companies who adopted agile development did so within a standard agile framework—typically Scrum. The few remaining companies had their own process, but in each case, it involved similar rapid cycles of development and the usual range of associated practices. Interviews provided direct assurance of the depth of agile understanding and activity in each case.

3.3.2 Level of Agile Adoption in Northern Ireland

Overall, the level of adoption of an agile approach to software development was much greater than the authors were expecting, based on earlier impressions gained from a range of interactions with local companies (McCaffery et al. 2004; Bustard and Sterritt 2008). The surprise is explained by the fact that over a third of those reporting routine use of agile development had only adopted it within a period of 18 months immediately before the survey.

From the interviews, it was clear that for many, agile development had become embedded as standard practice, illustrated by companies having the flexibility to adjust its use according to the type of project involved.

In innovation diffusion terms, adoption in Northern Ireland seemed very close to the middle position of the adoption curve. Specifically, 24 of the original 56 companies contacted seemed to be committed agile users, which approximates to 50 %, allowing for (1) the exclusion of some companies because they were not involved directly in software development and (2) the possibility that some of those who did not respond had involvement with agile work. This was known for one (large) company, in which a team had been using an agile approach for at least 2 years.

3.3.3 Generalizing the Results

Roughly half of the 37 companies surveyed were multinational, though a large number of the remaining indigenous companies operated worldwide, with some having offices and/or development facilities in other countries. This breadth of activity suggested that the results obtained for Northern Ireland could have wider significance. Similar studies in other comparable regions would be needed to resolve the matter, but in the meantime, it was considered useful to make some comparisons with other existing survey results. A quantitative comparison with Ambler's surveys (Ambler 2011) is problematic because of the different approach taken in each case, but it is encouraging to note broad similarities between his latest survey results in 2008 (Ambler 2008) and the results obtained here (Table 18.2). The tentative implications are that (1) the level of agile adoption in Northern Ireland is in reasonable step with the rest of the world, and in the other direction, (2) worldwide adoption of the agile development paradigm is at or around the peak level, as found in this study.

4 Conclusion

Getting an idea adopted, even if it is a very good idea, is a slow process. Agile software development is now well established, but there is still value in performing regular surveys of its adoption level, both to chart its progress and to monitor any

Table 18.2 Comparison with Ambler's 2008 survey

Observation	Ambler (%)	INI (%)
Respondents with agile projects beyond the pilot project phase	82	78
Respondents reporting improved productivity using agile teams	82	83
Respondents reporting improvements in the quality of software	77	60[a]
Respondents reporting improvements in business stakeholder satisfaction	78	79[b]
Respondents reporting a reduction in the overall cost of system development	37	29

[a]This figure is approximate. The precise percentage varies according to the specific "product quality" considered, with the closest comparison with Ambler's notion of software quality likely to be either "reliability" or "maintainability"
[b]This statistic refers to improvements on "customer engagement" which can be correlated with customer satisfaction

evolving views of its principles, techniques, and tools. This chapter describes such a survey covering software development companies in Northern Ireland. The work is distinctive in (1) attempting to provide a comprehensive treatment of a particular region, rather than relying on the voluntary completion of questionnaires from a range of sources, and (2) using structured interviews to ensure that the information gathered is complete and consistent.

The results suggest that agile development is now a mainstream software engineering practice, with approximately 50 % of companies using it routinely in commercial projects and all reporting a range of significant benefits. Comparing the results with surveys by Amber suggests that the Northern Ireland position is similar to that in other parts of the world where software development is a significant commercial activity. The implication is that the adoption of the agile approach now seems to have reached the midpoint of the adoption curve. This has consequences for companies that provide consultancy and training services for industry but also for academic institutions that prepare graduates for employment in the computing industry.

This chapter has considered one aspect of the survey data gathered, but there are others areas to be analyzed such as adherence to specific principles and practices and the role of tools in the development and maturation processes. Future research will consider such aspects and also return to the companies contacted through the survey to monitor their ongoing evolution.

Acknowledgements This work draws on the results of a survey commissioned by Invest Northern Ireland in 2010. The authors are very grateful to the companies who participated and particularly those who gave their time generously in helping to complete the questionnaires.

References

Abrahamsson P, Conboy K, Wang X (2009) Lots done, more to do: the current state of agile systems development research. Guest editorial in special issue: agile processes in software development. Eur J Inf Syst 18(4):281–284

Ambler SW (2008) Agile adoption rate survey results: February 2008. www.ambysoft.com/surveys/agileFebruary2008.html. Accessed Apr 2011

Ambler SW (2010a) Where is the proof. www.agilemodeling.com/essays/proof.htm. Accessed Apr 2011

Ambler SW (2010b) Agile software development. Encyclop Softw Eng 1(1):29–46

Ambler SW (2011) Surveys exploring the current state of information technology practices. www.ambysoft.com/surveys. Accessed Apr 2011

Beck K et al (2001) Manifesto for agile software development. http://agilemanifesto.org/. Accessed Apr 2011

Bustard DW, Sterritt R (eds) (2008) Proceedings of agility in the software process (ASP) in engineering of computer-based systems, Belfast, Northern Ireland. IEEE Press, pp 389–444

Chrissis MB, Konrad M, Shrum S (2011) CMMI for development: guidelines for process integration and product improvement, 3rd edn, SEI series in software engineering. Addison Wesley, Upper Saddle River

ISO/IEC 9126-1 (2001) Software engineering product quality – Part 1: Quality model. International Organisation for Standardisation, Geneva

Mahajan V, Peterson RA (eds) (1985) Models for innovation diffusion, vol 48, Quantitative applications in the social sciences. Sage Publications, Beverly Hills

McCaffery FH, Wilkie FG, McFall DJ, Lester NG (2004) Northern Ireland software industry survey. In: Proceedings of fourth international SPICE conference on process assessment and improvement, Lisbon, Portugal, pp 159–161

Moore G (1999) Crossing the chasm. Harper Business, New York

Moore GC, Benbasat I (1991) Development of an instrument to measure the perceptions of adopting an information technology innovation. Inf Syst Res 2(3):192–222

Pfleeger SL, Kitchenham BA (2002) Principles of survey research. Part 1 – Turning lemons into lemonade. ACM SIGSOFT Softw Eng Notes 26(6):16–18

Rogers EM (2003) Diffusion of Innovations, 5th edn. Free Press, New York

Schwaber K, Beedle M (2008) Agile software development with SCRUM. Pearson Education

Shore J, Warden S (2007) The art of agile development. O'Reilly Media, Beijing/Sebastopol

Tudor DJ, Tudor IJ (2010) The DSDM Atern student workbook, 2nd edn. Galatea Training Services Ltd., Knutsford

Vijayasarathy LR, Turk D (2008) Agile software development: a survey of early adopters. J Inf Technol Manage XIX(2):1–8

Chapter 19
An MAS-Based Implementation for Semantic Web Services Composition

Jun Shen, Ghassan Beydoun, and Graham Low

1 Introduction

With the increasing popularity and growth of Web services, many researchers have been interested in developing effective e-service or e-business applications based on existing components for agent-based systems (Negri et al. 2006). In a multi-agent system (MAS) composed of a heterogeneous collection of agents with distinct knowledge bases and capabilities, coordination and cooperation between agents facilitate the achievement of global goals that cannot be otherwise achieved by a single agent working in isolation (Wooldridge et al. 2000). The unique characteristics of a MAS have rendered most standard systems development methodologies inapplicable, leading to the development of agent-oriented software engineering methodologies (Tran et al. 2006a; Beydoun et al. 2009).

However, along with an increasing number of Web services developed in agent-oriented decentralised environments, it's essential to consider the quality of service (QoS), such as response time, for agents when running business processes. It is obvious that the dynamics and heterogeneity of distributed services are extremely important to both service requestors and service providers. Nevertheless, most recent research is predominantly syntactic and has not truly incorporated semantic ontology approaches for service description and composition within a realistic business context. The discovery and integration of a new service into an existing infrastructure is yet to be fully automated and currently requires significant human effort. As a result, it's problematic that traditional methodologies can effectively and autonomously conduct service discovery and composition in a complex

J. Shen (✉) • G. Beydoun
Faculty of Informatics, University of Wollongong, Wollongong, NSW, Australia
e-mail: jshen@uow.edu.au; beydoun@uow.edu.au

G. Low
Australian School of Business, University of New South Wales, Sydney, NSW, Australia
e-mail: g.low@unsw.edu.au

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_19,

dynamic environment. Moreover, the QoS specifications proposed in the literature (e.g. Papaioannou et al. 2006; Ran 2003; Tsesmetzis et al. 2006) are yet to agree on common defining concepts.

A set of non-functional properties in Web Service Modeling Ontology (WSMO) (Roman et al. 2005) can be used as a discriminating factor to refine P2P-based Web services so as to provide a more reliable service selection in business workflows. In this chapter, we present a scalable WSMO-based conceptual framework to describe QoS and other features of Web services in a P2P-based environment. We also sketch an automatic concomitant semantic Web service selection process to automatically find appropriate Web services that effectively fulfil the requestor's requirements. Hence, we design an approach to deal reasonably with the correlation between those requirement specifications and to select the most appropriate peers that will foster a better service composition.

The rest of this chapter is structured as follows. Section 2 illustrates our P2P architecture approach. Section 3 presents our QoS model and WSMO integration in our P2P framework and sketches a practical solution for selecting appropriate peers with multiple properties, specified by our service quality conceptual model. Section 4 introduces our implementation of the peer assignment prototype and two algorithms for service selection. Section 5 concludes with a discussion of other related work.

2 P2P MAS to Compose Semantic Web Services

Generally, in a P2P MAS architecture dedicated to sharing resources, the MAS acts as an interface to a set of resources distributed across a network. Each agent within the system typically acts as a gatekeeper to a local repository of resources that it shares with other similar peer agents as they broadcast their requests. In this architecture, all agents cooperate in fulfilling queries and having access to their repository of resources whenever a query received can be assisted by their local resources. Resources shared can be information (files of data, music etc.), e.g. as specified in systems similar to those in (Beydoun et al. 2009; Mine et al. 2004) or services as in this chapter.

In our proposed P2P framework, the MAS consisting of all cooperating agents responds to requests by a user (e.g. a service requester, a software developer, a human Web user) who is also represented by an agent in the P2P network that acts on his/her behalf. This agent aims to fulfil the request, e.g. locates services and responds to queries by other similar agents. The collection of all these agents and agents assisting them in their tasks form a P2P community-based cooperative MAS. For composing services using their semantics, a P2P MAS is shown in Fig. 19.1. An agent (an oblong in Fig. 19.1) representing a user (a hexagon in Fig. 19.1) contains a knowledge base about the services/resources that the user is willing to share with other users. Each service/file/resources (a cylinder in Fig. 19.1) is identified by a

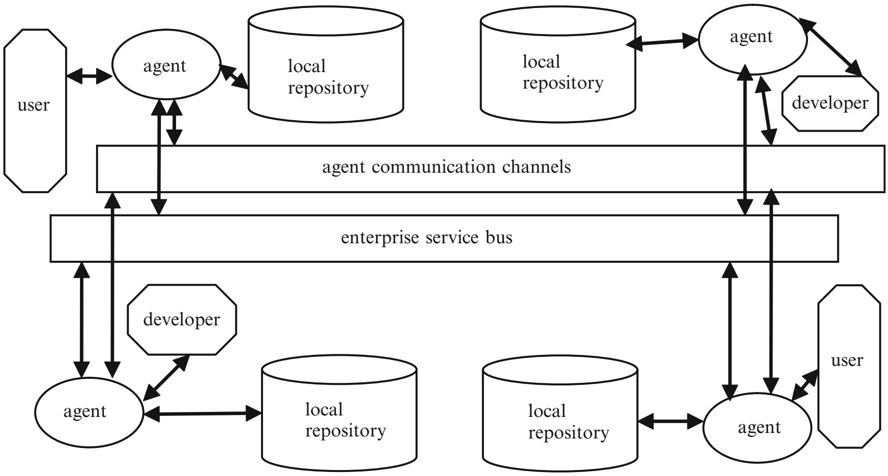


Fig. 19.1 The P2P multi-agent system

unique identifier within the P2P network (e.g. service identifier, HTML, PDF, music or video).

As agents automatically interact on behalf of users seeking services to be composed, communities of interest begin to emerge. These communities may overlap. Providers and users of services may belong to more than one community; for instance, a service to 'open an account' may belong to the community of banking developers as well as that for insurance developers. As more and more services are composed, agents become more efficient and effective by interacting with the agents in the communities most likely to be able to provide them with service components. The P2P system is responsible for locating sites where candidate services are available, based on the previous requests made. The mediation between service requesters and providers is always done by the system. When an agent makes a service request, a candidate agent provider responds either by providing details about services it can supply or refusing the service. When all responses are received, the requesting agent combines and refines the results to compile a list of services that can be composed to fulfil the request. The requester agent can then select which services it wants to compose and initiates the composition.

After a successful composition, a requester's knowledge base is updated to include the received and the composed services. Similarly, for all agents involved in processing a service request, their knowledge bases are also updated with additional information reflecting the domain and attributes of the requester agent. This information is used in future service requests. That is, as agents interact, they develop awareness of the services possessed by their peers and which peers may be interested in the services that they themselves have. Each agent keeps a record of its history of service sharing in order to evaluate the quality of services (QoS) and to use this for future service requests. The collection of this history is in essence a distributed QoS ontology distributed across providers. The QoS ontology will provide assessments of past queries and providers, and also information to make QoS estimates for members. It is used to produce a short list of candidate nodes for future queries, by calculating the similarity between the current query and a past query and its QoS. In a fully evolved P2P system, agents may use this QoS knowledge about other users' interests to request/negotiate for information from their peers when they do not know who has services of interest. New providers are constantly added to the history, expanding the user-agent's contact circle.

The proposed strategy of service sharing can be applied to any domain that can be prescribed by an ontology. The proposed P2P service execution system subsequently allows dynamic composition of Web services in a highly distributed and heterogeneous computing environment (Tran et al. 2006b) that is adapted from Shen et al. (2007) to highlight how ontologies can be used by taking advantage of semantically driven composition of services as is often advocated, e.g. in Sousa et al. (2009). The system will provide, to both service requestors and service providers, the quality of service (QoS) evaluation. The system will identify the capability and performance of the service providers so as to enhance the service composition for service clients over the real distributed service network. Due to the complexity of QoS metrics (Negri et al. 2006; Tsesmetzis et al. 2006), a well-defined QoS service description does not actually exist. With a P2P architecture, the QoS is gauged by a service client through cooperative interactions with other peers that can potentially provide the service. The scope of using ontology-based profiles in this MAS development is possible since most of the current work focuses on the definition of a QoS ontology, vocabulary or measurements and, to a lesser extent, on a uniform evaluation of qualities. We propose to exploit Web Service Modeling Ontology (WSMO) (Roman et al. 2005) as complementary conceptual framework to create the QoS ontology to describe various perspectives on Web services and to facilitate integrating the services. In a specified domain, a problem-solving method (Tran et al. 2006b) unit of analysis will nicely correspond to a service carried by an agent. The agents themselves can dynamically select PSM implementations that best suit the service or the QoS required to match the requested service-level agreement (SLA). This selection will be made using a P2P searching mechanism to locate appropriate services from other peer agents. Cooperative communication between agents about their existing services, their past service requests and their performance will enable service requesters to locate the peer-service provider with the most suitable QoS.

3 Conceptual QoS Model for Agents and Peers Selecting Services

WSMO defines four high-level notions that relate to semantic Web services, namely, ontologies, goals, mediators and Web services. *Ontologies* are viewed as formal and explicit specifications of shared conceptualisations (Roman et al. 2005).

They define a common agreed-upon terminology by providing concepts and relationships among the set of concepts from a real-world domain. *Goals* are depictions of the expectations a service requestor may have when seeking for a service based on functionality, approach and quality of service. *Mediators* coordinate the heterogeneity problem that occurs between descriptions at different levels (Toma et al. 2006): data level (different terminologies), protocol level (different communication behaviour between services) and process level (different business processes). WSMO defines four types of mediators: OO mediators connect and mediate heterogeneous ontologies, GG mediators connect goals, WG mediators link Web services to goals and WW mediators connect Web services resolving mismatches between them. *Web services* are descriptions of services that are requested by service requestors, provided by service providers and agreed between service providers and requesters.

Non-functional properties are usually utilised to describe non-functional aspects such as the creator and the creation date and to provide natural language descriptions. The four WSMO elements have their own non-functional properties. In this chapter, however, our QoS extension is of the same nature as the notion of non-functional properties in 'Web services'. In other words, we mainly introduce descriptors of QoS, such as performance, availability and spatial features of distributed services. The incorporated QoS properties could also be used in parallel with existing non-functional attributes proposed by other WSMO elements. Thus, it is consistent to consider QoS parameters as non-functional properties.

We develop the non-functional properties in WSMO in order to support adaptive P2P-based service composition. Coordinator roles are allocated to agents in our framework at runtime (as described in Sect. 2). These organise the peer/agent selection process and distribute tasks. The resultant decentralised architecture is coordinated and self-managed effectively with services being allocated to peer/agent hosts, who are able to communicate with each other according to a real business process agreement or standard workflow definitions. In the rest of this chapter, we present a more effective representation to enable peers to evaluate candidate composition (in Sect. 3.1) and select the most appropriate peers (in Sect. 3.2) for a requested service in a P2P information system. The P2P-based service selection problem can be generally formulated as the following: Consider P as a set of composing agents, $P = \{p_1, p_2, \ldots, p_N\}$, where each P is an agent that gets involved in the composition of a number of services from the set S covering M atomic services, $S = \{s_1, s_2, \ldots, s_M\}$. Each atomic service ($s_j$) cannot be allocated to multiple Peers, so let $x_{ij} = 1$ if atomic service (s_j) is allocated to Peer (p_i) and $x_{ij} = 0$ otherwise, and a Peer (p_i) can be allocated with a set of atomic services: $A_{p_i} = \{s_1, s_2, \ldots, s_{m_i}\}$. Moreover, let $Q_{p_i,s_j} = <\mathrm{RT}, \mathrm{CT}, \mathrm{AV}, \mathrm{RB}>$ denote the QoS features of Peer (p_i) for atomic service (s_j), and $<\mathrm{RT}, \mathrm{CT}, \mathrm{AV}, \mathrm{RB}>$ represent ResponseTime, ComputationCost, Availability and Reliability. To optimally perform the service composition, the basic objective is to find a set of appropriate Peers that makes response time and computation cost as small as possible, while keeping the availability and reliability as large as possible. Therefore, four sub-objectives can be defined as min($\sum$RT), min($\sum$CT), max($\prod$AV) and max($\prod$RB):

$$O_1 = \min\left(\sum_{p_i \in P} \sum_{k=1}^{m_i} Q_{p_i,s_k}(\mathrm{RT})\right), \; s_k \in A_{p_i} \tag{19.1}$$

$$O_2 = \min\left(\sum_{p_i \in P} \sum_{k=1}^{m_i} Q_{p_i,s_k}(\mathrm{CT})\right), \; s_k \in A_{p_i} \tag{19.2}$$

$$O_3 = \max\left(\prod_{p_i \in P} \prod_{k=1}^{m_i} Q_{p_i,s_k}(\mathrm{AV})\right), \; s_k \in A_{p_i} \tag{19.3}$$

$$O_4 = \max\left(\prod_{p_i \in P} \prod_{k=1}^{m_i} Q_{p_i,s_k}(\mathrm{RB})\right), \; s_k \in A_{p_i} \tag{19.4}$$

However, in order to consider the four objectives as a whole, it can be set as:

$$F = \max\left(\frac{w_3 \cdot O_3 + w_4 \cdot O_4}{w_1 \cdot O_1 + w_2 \cdot O_2}\right) \tag{19.5}$$

where $\{w_1, w_2, w_3, w_4\}$ denote the weights for the four QoS properties: RT, CT, AV and RB. Subject to the following constraints:

$$\sum_{i=1}^{N} x_{ij} = 1, \quad j = 1, 2, \ldots, M, \quad \sum_{j=1}^{M} x_{ij} = m_i, \quad i = 1, 2, \ldots, N, \quad \sum_{i=1}^{N} m_i = M$$

which respectively ensure there is no conflict between Peers to conduct atomic service allocation and guarantee that the number of allocated atomic services of a peer are valid.

4 System Implementation

To enable the peers' coordinating agent to intelligently select peers and plan a whole composition process, we used the ant colony algorithm (ACO) and genetic algorithm (GA) to test how composition can be achieved faster. First, we developed a JXTA-based peer-to-peer system to allocate tasks to a group of agents/peers, who in turn manage the invocation of relevant atomic services. Figure 19.2 is the basic configuration of the grouping and paring of peer's capability to serve a certain Web service with specific QoS agreements.

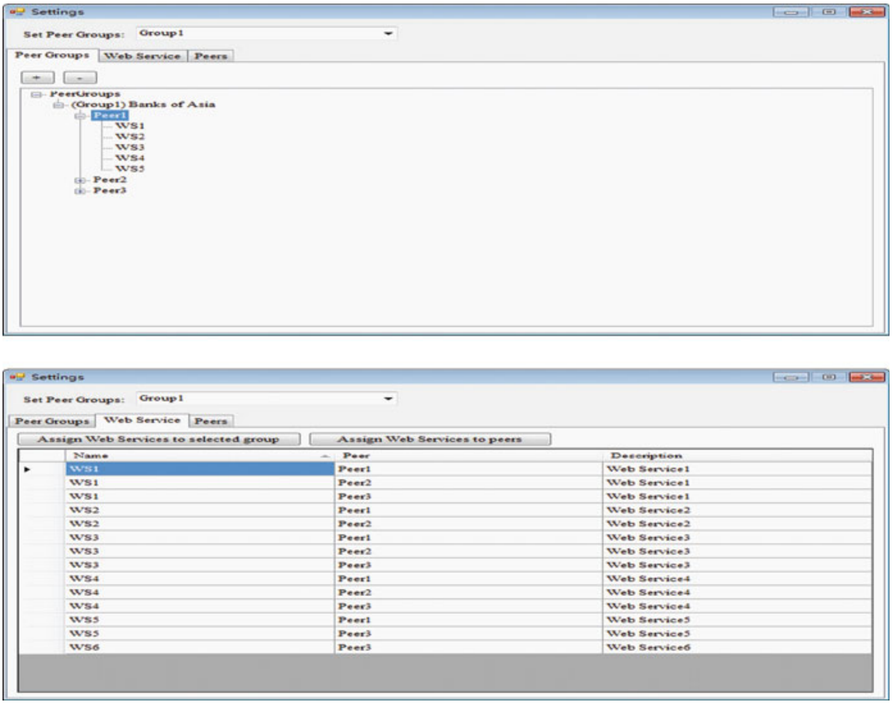


Fig. 19.2 Configuration of peer groups and peer-service pairs

The QoS parameters for ResponseTime, ComputationCost, Availability and Reliability are based on history observations in prototype, but for simulation purposes, they are generated randomly to be applied in the relevant algorithms. When all parameters are set up for peers and services, we apply ACO and GA (shown in Figs. 19.3 and 19.4, respectively).

The ACO and GA algorithms can be implemented in Java on top of the peer-to-peer system. For simulation purposes, we used Matlab to easily test the different parameter settings with meaningful comparison results. Table 19.1 shows how fast the ACO algorithm can achieve a successful composition with regard to different number of activities in a composition (3, 30, 40 and 50) and number of supporting peers. The experiments assume there are 100 available atom Web services for selection and all peers are available to support these services. We assume an optimal service selection is a composition of agent-service pairs that yields the maximum F value, i.e. the converged objective function value as detailed in Eq. (19.5).

The two algorithms were compared and evaluated for average values of running the objective function for 30 times with standard deviation and the value of statistical significance ($p < 0.05$).

As shown in Fig. 19.5, when the scale of the MAS peers is less than 60, the average quality of solution of the two algorithms tends to be similar. However,

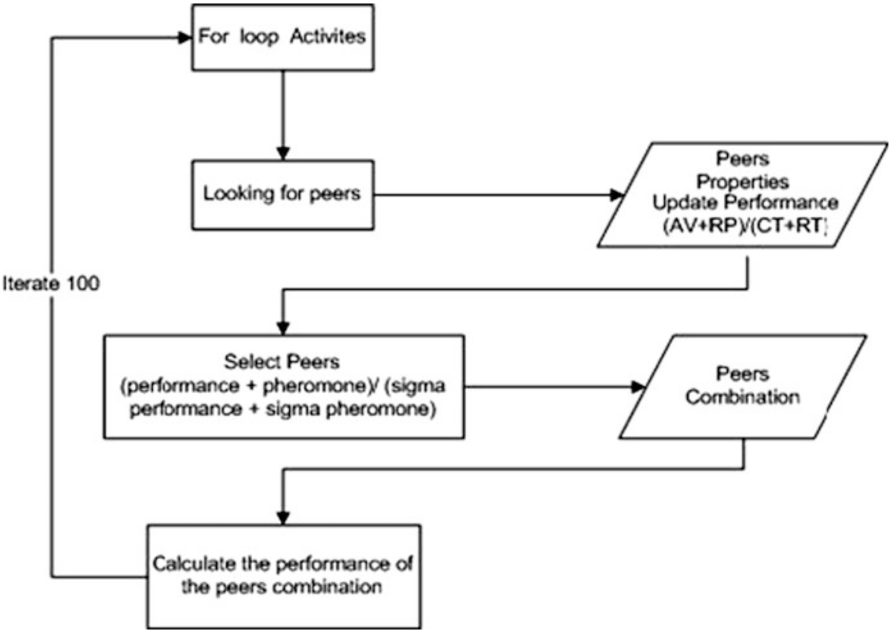


Fig. 19.3 Diagram of ACO algorithm for peer/service selection

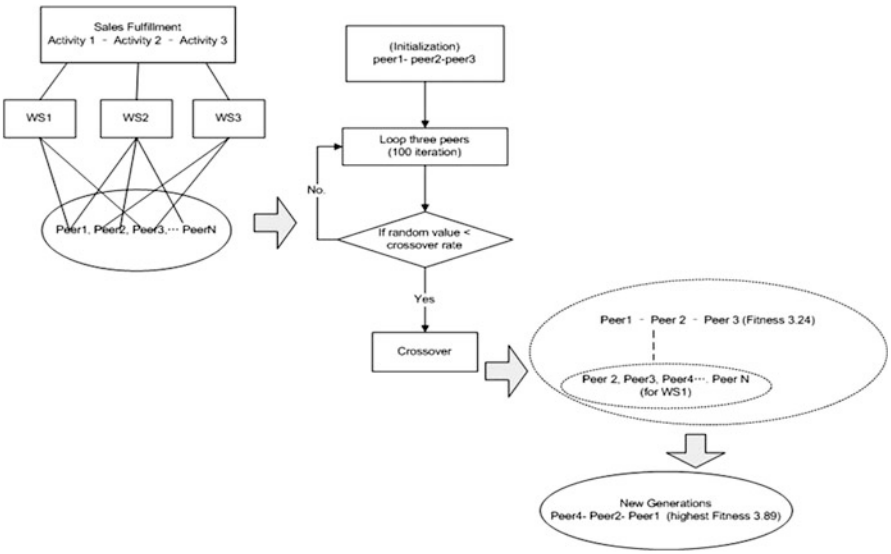


Fig. 19.4 Diagram of GA algorithm for peer/service selection

Table 19.1 Iterations of ACO implementation for 100 Web services

Number of activities	Number of peers	Possibility of composition	Number of iteration
3	10	1,000	14
3	20	8,000	20
3	30	27,000	31
3	40	64,000	65
3	**50**	**125,000**	**97**
3	60	216,000	369
3	70	343,000	491
30	**50**	$\mathbf{50^{30}}$	**324**
40	**50**	$\mathbf{50^{40}}$	**422**
50	**50**	$\mathbf{50^{50}}$	**514**

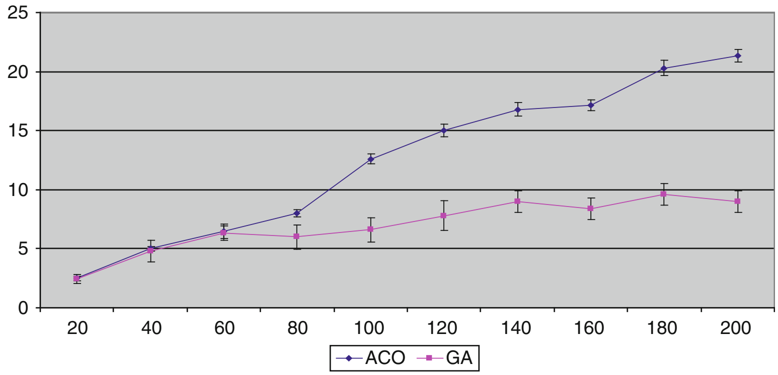


Fig. 19.5 Comparison to the quality of service composition

when the scale increases, the performance of GA decreases while the ACO algorithm remains stable. A more detailed comparison between GA and ACO is beyond the scope of this chapter (Shen et al. 2011).

5 Discussion and Conclusion

This chapter aimed to ease semantic Web services development for business process management systems. Towards this, it focussed on QoS-aware service selection and composition which has been subject to considerable attention recently, e.g. (Yuan and Shen 2007, 2008; Zhou et al. 2004). Functional and non-functional properties are typically seen as the two essential aspects for requirement analysis (Sadrei et al. 2007), and they are typically used to describe the semantic of Web services. Functional properties describe how a Web service meets the functional requirements of an anticipated service while non-functional properties describe the performance of the service. This distinction underlies many Web services identification frameworks which in turn can be implemented in common ontology languages such as OWL-S (Yuan and Shen 2007, 2008). Such representations form the ontological description of services and the design of the selection process. Our previous prototype in Yuan and Shen (2008) was limited to a single specification. It only considered 'ResponseTime' as the selection criterion, which was not sufficient for effective services composition. This chapter extends the description of non-functional properties via a model-driven WSMO specification and also presents an approach for the coordinator to automatically identify the best peers through unifying qualities and properties.

Our work is in line with other works focussing on the development of QoS ontology languages and vocabularies, as well as the identification of various QoS metrics and their measurements with respect to semantic e-services, e.g. Ankolekar, Lee et al. (2003), Zhou et al. (2004) or Ran (2003). We presented a QoS ontology to complement the DAML-S (Ankolekar) ontology in order to provide a better QoS metrics model. Works, Lee et al. (2003) and Ran (2003), emphasised a definition of QoS aspects and metrics, but have not included the extensible aspects in QoS, such as incorporating geo features which we earlier proposed in Yuan and Shen (2008). In Ran (2003), all of the possible quality requirements were introduced and divided into several categories, such as runtime-related, transaction support-related, configuration management and cost-related and security-related QoS. Both Lee et al. (2003) and Ran (2003) present their definitions and possible determinants. Unfortunately, they are too abstract to suit the implementation requirement. They did not tend to present a practical approach for real services selection. In Papaioannou et al. (2006) and Tsesmetzis et al. (2006), the authors focussed on the creation of QoS ontology models, which proposed QoS ontology frameworks aimed at formally describing arbitrary QoS parameters. Additionally, Liu et al. (2004) and Mou et al. (2005) attempted to conduct a proper evaluation framework and proposed QoS-based service selection, despite the authors failing to present how to validate the effectiveness of such a framework or algorithm. In contrast, our proposed P2P-based service selection framework automates the basic problem of service selection using a representation with multiple properties and augments the WSMO description by involving typical QoS perspectives. Our service peer selection model will be further modernised in the future by focussing on concrete and detailed geographic features for location-based services. This will improve our framework for P2P-based workflow under more dynamic circumstances more effectively.

References

Ankolekar A, Burstein M, Hobbs JR, Lassila O, Martin DL, McDermott D, McIlraith SA, Narayanan S, Paolucci M, Payne TR, Sycara K. DAML-S: Web service description for the semantic Web. In: The first international semantic Web conference (ISWC'02). http://www.cs.cmu.edu/~terryp/Pubs/ISWC2002-DAMLS.pdf

Beydoun G, Low G, Mouratidis H, Henderson-Sellers B (2009) A security-aware metamodel for multi-agent systems (MAS). Inf Softw Technol 51(5):832–845

Lee K, Jeon J, Lee W, Jeong S, Park S (2003) QoS for Web services: requirements and possible approaches. W3C Working Group Note 25, 2003. Available at http://www.w3c.or.kr/kr-office/TR/2003/ws-qos/
Liu YT, Ngu AHH, Zeng LZ (2004) QoS computation and policing in dynamic Web service selection. In: Proceedings of international conference on World Wide Web (WWW'04). Alternate track papers & posters, pp 66–73
Mine T, Matsuno D, Kogo A, Amamiya M (2004) Design and implementation of agent community based peer-to-peer information retrieval method. Cooperative information agents (CIA2004), Springer
Mou Y, Cao J, Zhang SS, Zhang JH (2005) Interactive Web service choice-making based on extended QoS model. In: The fifth international conference on computer and information technology (CIT 2005). IEEE Press, Los Alamitos, pp 1130–1134
Negri A, Poggi A, Tomaiuolo M, Turci P (2006) Ontologies and Web services: agents for e-business applications. In: Proceedings of the fifth international joint conference on autonomous agents and multi-agent systems (AAMAS'06), New York. ACM Press, pp 907–914
Papaioannou IV, Tsesmetzis DT, Roussaki IG, Miltiades EA (2006) QoS ontology language for Web-services. In: Proceedings of the 20th international conference on advanced information networking and applications (AINA 2006). IEEE Press, Los Alamitos
Ran S (2003) A model for Web services discovery with QoS. ACM SIGecom Exch 4(1):1–10
Roman D, Keller U, Lausen H et al (2005) Web service modeling ontology. Appl Ontol 1 (1):77–106
Sadrei E, Aurum A, Beydoun G, Paech B (2007) A field study of the requirements engineering practice in Australian software industry. Int J Requir Eng J 12:145–162
Shen J, Yang Y, Yan J (2007) A P2P based service flow system with advanced ontology-based service profiles. Adv Eng Inform 21(2):221–229
Shen J, Beydoun G, Yuan S, Low G (2011) Comparison of bio-inspired algorithms for peer selection in services composition. In: Proceedings of IEEE international conference on services computing, Washington, DC, pp 250–257
Sousa JPP, Carrapatoso E, Fonesca B, Pimentel MGC, Bulcao-Neto RF (2009) Composition of context-aware mobile services using a semantic context model. IARIA Int J Adv Softw 2 (2):1–13
Toma I, Foxvog D, Jaeger MC (2006) Modeling QoS characteristics in WSMO. Workshop on middleware for service oriented computing (MW4SOC'06). ACM, New York, pp 42–47
Tran N, Beydoun G, Low G (2006a) Design of a peer-to-peer information sharing MAS using MOBMAS (ontology-centric agent oriented methodology). In: International conference on information systems development, Budapest, Hungary. Springer, Heidelberg
Tran QNN, Low GC, Beydoun G (2006b) A methodological framework for ontology centric agent oriented software engineering. Int J Comput Syst Sci Eng 21:117–132
Tsesmetzis DT, Roussaki IG, Papaioannou IV, Anagnostou ME (2006) QoS awareness support in Web-service semantics. In: Proceedings of the advanced international conference on telecommunications and international conference on internet and Web applications and services (AICT/ICIW 2006), p 128
Wooldridge M, Jennings NR, Kinny D (2000) The Gaia methodology for agent-oriented analysis and design. Autonomous agents and multi-agent systems. Kluwer Academic Publishers
Yuan S, Shen J (2007) QoS aware service selection in P2P-based business process frameworks. In: Proceedings of the 9th IEEE conference on E-commerce technology and the 4th IEEE conference on enterprise computing, E-commerce and E-services (CEC/EEE'07), Tokyo, Japan, pp 675–682
Yuan S, Shen J (2008) Mining E-Services in P2P-based workflow enactments. Special issue: Web mining applications in E-commerce and E-services, of online information review, Emerald Group Publishing 32(2):163–178
Zhou C, Chia LT, Lee BS (2004) DAML-QoS ontology for Web services. In: Proceedings of international conference on Web services (ICWS'04), pp 472–479

Chapter 20
Software Process Improvement in Small Companies as a Path to Enterprise Architecture

Alena Buchalcevova

1 Introduction

This chapter concentrates on two initiatives that are capable of bringing business benefits to small and medium companies, that is, enterprise architecture and software process improvement. In recent years, there has been an increasing interest in enterprise architecture (EA) that is widely referred to as an essential instrument for ensuring an enterprise's agility, consistency, compliance, efficiency and business and IT alignment (Kunstova 2011). Although enterprise architecture is mostly applied by large companies, this chapter shows that it might be worthwhile for small and medium companies as well.

Software process improvement (SPI) is a way of improving a status of software development which is in these days still not satisfactory enough. International standards like ISO/IEC 12207 (2008), ISO/IEC 15289 (2006), ISO/IEC 15504 (2004) and ISO 9001 (2008) play an important role in SPI initiatives as companies are willing to show compliance with common business rules. However, it was identified that small companies find it difficult to implement international standards as they do not have enough resources, in terms of number of employees, budget and time (Laporte et al. 2006; Anacleto et al. 2004). To solve these difficulties, the ISO/IEC 29110 standard "Lifecycle Profiles for Very Small Entities" is being developed.

In this chapter, the author presents a possibility of an added value which lies in the combining of software process improvement and enterprise architecture initiatives. Second part of this chapter examines the role of enterprise architecture and current status of its adoption worldwide and also in the Czech Republic. In the third part, the most accepted EA framework TOGAF is briefly described, which is

A. Buchalcevova
Department of Information Technologies, Prague University of Economics, Prague, Czech Republic
e-mail: buchalc@vse.cz

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_20,

recommended by the author for usage in small and medium companies. Fourth part presents reasons for developing the ISO/IEC 29110 standard "Lifecycle Profiles for Very Small Entities" and explains its main concepts and structure. The main contribution of this chapter consists in the fifth part where relationships between enterprise architecture and software process improvement are outlined, and an EA framework for SMEs is presented.

2 Current Status of Enterprise Architecture Adoption

A large number of methods for enterprise architecture management have been developed by academia and practitioners so far. These methods are mostly referred to as frameworks. According to ISO/IEC 42010:2007, "An architecture framework establishes a common practise for creating, organizing, interpreting and analysing architectural descriptions used within a particular domain of application or stakeholder community" (ISO/IEC 42010:2007). To the well-known frameworks belong, for example, TOGAF, the Federal Enterprise Architectural Framework (FEAF) (Information Officer) Council 2001), the Gartner EA Framework (2005), the Department of Defense Architecture Framework (DoDAF) (DoDAF Working Group 2007) and the Zachman Framework (Zachman 1987).

The results of the Cutter Consortium survey conducted in 2010 (Rosen 2011) show a current status in the enterprise architecture domain. 148 respondents took part in this survey, 47 % coming from North America, 21 % from Europe, 10 % from Australia/Pacific, 6 % from Asia and 16 % from other locations. Company sizes ranged from having fewer than 10 to more than 1,000 IT employees. This survey indicates a growth of experience with EA as 93 % of respondents stated that they already use EA or are just starting to. The main reasons resulting in this growth represent "the increased complexity of IT, the cloud, Enterprise 2.0, other new technology options, as well as the need for more flexible IT systems and reduction in IT costs"(Rosen 2011). Thiry-four percent of respondents that use EA stated they utilise TOGAF because "it is the only standards-based, general-purpose framework that can provide a ready set of principles, processes, and structure for organizing architecture programs" (Rosen 2011). Further, 29 % of the respondents take advantage of combining various EA methods, 3 % utilise Zachmann, 2 % utilise DoDAF/MoDAF, 17 % use other methods and 16 % apply none. In spite of the convincing win, "the general-purpose nature of TOGAF also provides one of the biggest challenges, as organizations need to tailor large, generic content down to the subset that meets their individual needs and goals" (Rosen 2011).

Seeing a broad adoption of EA worldwide, our team at the Department of Information Technologies of Prague University of Economics started wondering about the situation in the Czech Republic. Therefore, during 2010, we decided to conduct a nationwide survey focused on various characteristics of IS/ICT management. Even though the scope of the survey was quite large, there was a part concentrating on the usage of various architecture views (architectures) in

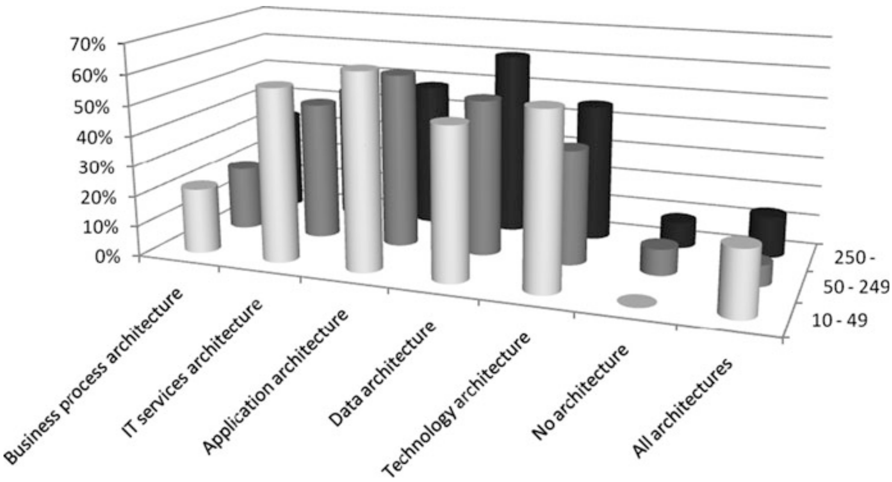


Fig. 20.1 Percentage of using various types of architecture in small, medium and large companies

companies. Respondents composed of companies of various sizes were divided according to the number of employees into three groups – small companies with 10–49 employees (14 respondents), middle-sized companies with 50–249 employees (175 companies) and large companies with more than 250 employees (81 companies). We questioned the companies about the types of architectures being used. Among the possible answers that appeared include business-process architecture, architecture of IT services, application architecture, data architecture, technology infrastructure architecture or none of them. Results in a graphic form are shown in the Fig. 20.1.

Surprisingly, we found out that the percentage of small companies using architectures is the same or even higher than in medium and large companies. Every small company in the sample utilises at least one architecture view. Moreover, 21 % of them use all architectures, which overreaches the number in medium (7 %) and large companies (14 %). Detailed analysis of these results showed that each small company that makes use of all architectures is either a local branch of a foreign company or has its seat abroad. Companies that apply all types of architecture are likely to operate in the EA domain and would be a subject of the next survey focused specifically on enterprise architecture.

3 TOGAF

As stated before, the most popular EA framework nowadays is the Open Group Architecture Framework (TOGAF). TOGAF, created by members of the Open Group consortium, is an enterprise architecture framework that emerged during the

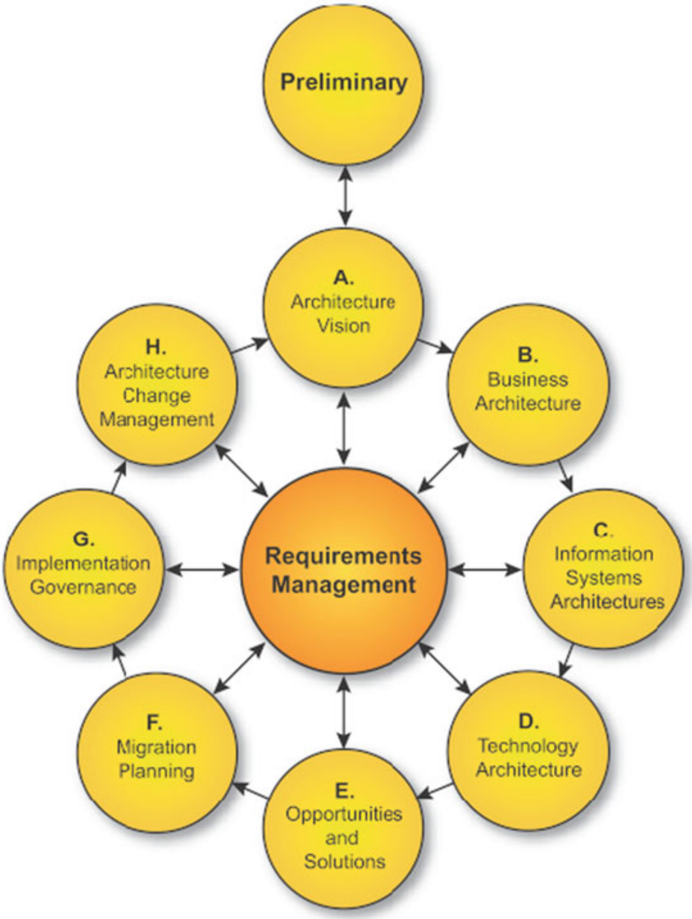


Fig. 20.2 TOGAF ADM (TOGAF – The Open Group Architecture Framework)

last two decades "with the objective of becoming a standard for EA development" (TOGAF).

According to TOGAF, framework is "a structure for content or process that can be used as a tool to structure thinking, ensuring consistency and completeness" (TOGAF). In this sense, TOGAF determines four architecture domains: Business Architecture, Data Architecture, Application Architecture and Technology Architecture. A key component of TOGAF is the Architecture Development Method (ADM), which provides full guidance for implementing and executing an organisation's enterprise architecture. The process consists of multiple, consecutive phases enclosed in a loop (see Fig. 20.2).

The preliminary phase encompasses the preparation and initiation activities for EA. *Phase A*, the initial phase of a cycle, includes defining the scope, identifying the stakeholders, creating the Architecture Vision and obtaining approvals. *Phase B* concentrates on the development of a Business Architecture, whereas *phase C* on the development of Information Systems Architectures for an architecture project, including the development of Data and Application Architectures. *Phase D* is focused on developing of the Technology Architecture for an architecture project. *Phase E* conducts initial implementation planning and the identification of delivery vehicles for the architecture defined in the previous phases. *Phase F* formulates transition architectures. *Phase G* provides an architectural oversight of the implementation. *Phase H* establishes procedures for managing change to the new architecture. *Requirements management* examines the process of managing architecture requirements throughout the ADM TOGAF.

Some authors argue (Vorisek et al. 2011) that additional architecture has to be added into the concept – the architecture of ICT services. Vorisek et al. in (2011) propose to insert the ICT service architecture between phases B and C of ADM. In their SPSPR framework, the ICT service architecture involves all ICT services delivered to users and their links. The application and technology architectures involve only those software and hardware objects which are owned and administrated by the enterprise. They do not involve objects owned by the external service provider.

4 Software Life Cycle Processes for Very Small Entities

Worldwide-conducted surveys (Laporte et al. 2006; Anacleto et al. 2004) indicated that even though very small companies developing software have a significant influence on the economy, most of them do not implement any international standards and models like ISO/IEC12207 (ISO/IEC 12207: 2008) or CMMI. Subsequently, these companies have no or very limited opportunities to be recognised as entities that produce quality software and therefore are often cut off from contracts.

The ISO/IEC 29110 standard “Lifecycle Profiles for Very Small Entities” aims at addressing these issues. The term “very small entity” (VSE) was defined by the ISO/IEC JTC1/SC7 Working Group 24 and consequently adopted for use in the emerging ISO/IEC 29110 software process lifecycle standard (ISO/IEC 29110-1: 2010), as being “an entity (enterprise, organization, department or project) having up to 25 people”. The 29110 standard consists of five parts as shown in Fig. 20.3 (ISO/IEC 29110-1: 2010).

Part 1 Overview (ISO/IEC 29110-1) explains main concepts, terms and structure of the standard. Part 2 Framework and Taxonomy (ISO/IEC 29110-2) presents principles and mechanism of building VSE profiles.[1] Part 3 Assessment Guide

[1] VSE Profile is a subset of the base standard for VSE that is necessary to accomplish a particular function.

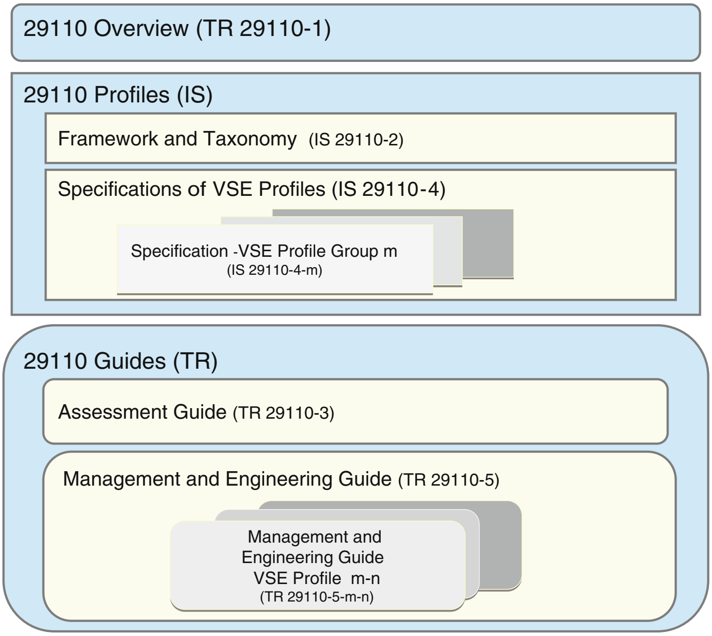


Fig. 20.3 ISO/IEC 29110 set of documents (ISO/IEC 29110-1: 2010)

(ISO/IEC 29110-3) defines the process assessment guidelines and compliance requirements needed to meet the purpose of the defined VSE profiles. This part of the standard is used by certified assessors for VSE assessment. Part 4 Specifications of VSE profiles (ISO/IEC 29110-4) provides the mapping to the source standards and is useful for method developers and assessors. Part 5 Management and Engineering Guide (ISO/IEC 29110-5) is intended for VSEs.

The set of 29110 standards was published in 2010. Part 1, 3 and 5 will be available at no cost from ISO. ISO/IEC 29110 is based on existing standards like ISO/IEC 12207 (2008), ISO/IEC 15289 (2006), ISO/IEC 15504 (2004) and ISO 9001 (2008). The approach used to develop ISO/IEC 29110 consists according to (O'Connor and Laporte 2010) of three steps: (1) selecting ISO/IEC 12207 process subset applicable to VSEs of less than 25 employees, (2) tailoring the subset to fit VSE needs, and (3) developing guidelines for VSEs.

A key concept of the ISO/IEC 29110 standard lies in development of various VSE profiles. As a starting point, the "Generic" profile group was defined, which is applicable to a vast majority of VSEs that do not develop critical software. Within the Generic Profile Group, four profiles were proposed, that is, Entry, Basic, Intermediate and Advanced. By using these profiles, very small companies have the chance to improve their processes in a clear and stepwise manner. At present, the Basic Profile intended for a single project with no special risks or situational factors was developed and published (at the end of 2010). Implementation of the Basic Profile enables VSE to establish good practices in the project management process (project planning, monitoring and control, configuration management) and in the software engineering process (requirements management, analysis, design, construction, verification, validation and testing). As some pilot projects of the Basic Profile implementation in VSEs showed, this profile is still difficult to implement for some companies. That is why the Entry Profile was proposed. The Intermediate Profile is on the contrary intended for VSE, which has more than one project at a time, and therefore is aware of assigning project resources, monitoring projects for achieving business objectives and customer satisfaction and defining, deploying and improving the organisational standard processes to achieve similar results in all projects. In addition, the Advanced Profile is going to support VSEs with business management practices.

To help VSE with implementation of the Basic Profile, series of deployment packages were developed and offered for free (Deployment packages repository). A Deployment Package acts as a detailed methodology that guides company through the process of profile implementation. Typical Deployment Package includes process descriptions, activities, tasks, roles and products, templates, checklists, examples, reference and mapping to standards and models and a list of supporting tools. In order to accelerate the adoption of the ISO/IEC 29110 standard, a set of university courses for undergraduate and graduate students was created (VSE Education Special Interest Group).

5 Combination of Software Process Improvement and Enterprise Architecture Initiatives

Despite a general experience that enterprise architecture is applied mostly by large companies, results of our survey (described in Sect. 2) show that it is also subject of an interest for small and medium enterprises (SMEs). EA is able to bring them business agility, efficiency and competitiveness, which are crucial for them. If an SME accepts the idea that EA could be beneficial, it primarily has to select a suitable EA framework. In this matter, I would recommend choosing the TOGAF framework. This conviction is derived from my experience in building framework for software development methodologies selection, my participation in development of standards for VSEs and analysis of various EA frameworks. TOGAF framework represents from my point of view the best option because of the following reasons:

- TOGAF is the most used framework worldwide.
- TOGAF may be used freely.
- TOGAF is prepared for tailoring.
- TOGAF embraces terminology of ISO/IEC 42010:2007 standard, so it can be combined with ISO/IEC 29110.

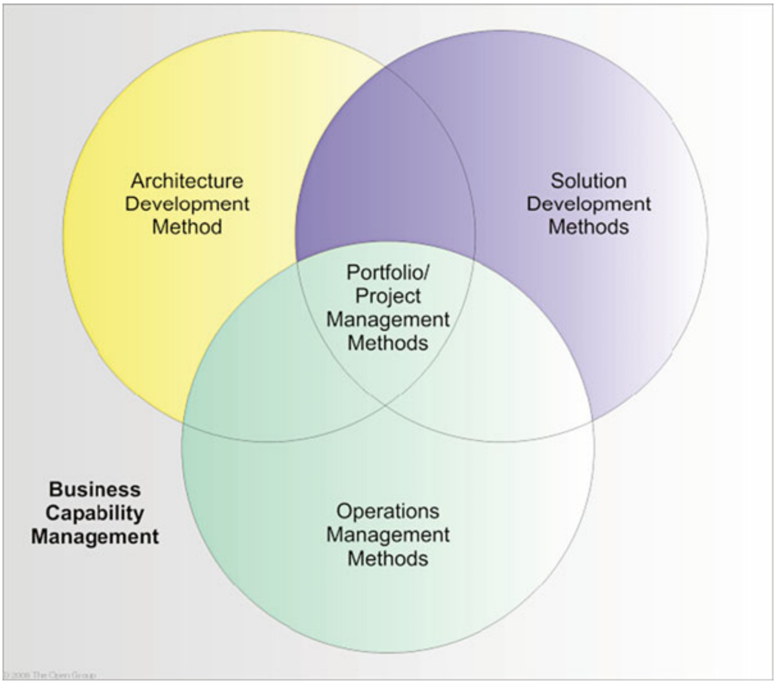


Fig. 20.4 Management frameworks to co-ordinate with TOGAF (TOGAF – The Open Group Architecture Framework)

Despite the fact that using EA framework helps the company significantly, it is a resource-consuming process. According to (Ernst 2008), additional problems with EA implementation are stated, for example, “lack of an actual starting point for an EA management initiative, assumption that EA is introduced as one single piece and thereby lack of an incremental EA development”. In order to use the EA framework more effectively, it should be extended. To do so, it is essential to take into account that EA is not standalone in an organisation, but has to correlate with other existing frameworks or methods. Examples of these relationships, which are mentioned in TOGAF, are shown in Fig. 20.4. Regarding this chapter’s perspective, we focus on relationship between EA framework and Solution Development Methods. We have to control both sides of this relationship which are described in following sections.

5.1 Defining EA Framework for SMEs

EA framework can be improved by adding certain concepts, processes and features from Solution Development Methods which are much more mature. Therefore, we have recently proposed the enterprise architecture for small and medium enterprises (EAFSME) framework which takes an inspiration from the ISO/IEC 29110 standard "Lifecycle Profiles for Very Small Entities".

EAFSME is based on TOGAF, but uses a subset of TOGAF practices that are suitable for small and medium companies. Moreover, a set of profiles is planned to be defined, that is, Entry, Basic, Intermediate and Advanced that will drive a company stepwise to the fulfilment of its goal – the EA implementation.

EAFSME aims to eliminate further limitations of TOGAF which were pointed out by Temnenco in (2007). He states that "TOGAF ADM is not an end-to-end process and requires organization-specific tailoring, as well TOGAF is not as detailed as some other methodologies and does not include any management tools like for example Rational Method Composer for RUP" (Temnenco 2007). To address this issue, we suppose to build up EAFSME profiles in the Eclipse Process Framework Composer that is available under the open-source licence and is also used for capturing VSE profiles. As stated earlier, small companies need detailed guidelines to be able to implement software process standards. This condition applies to EA implementation as well. Therefore, we are planning to develop Deployment Packages to capture detailed activities, roles and work products, as well as templates and checklists.

Extent of the adoption of enterprise architecture depends significantly on enterprise architect competencies. Our team at the Department of Information Technologies of Prague University of Economics carried out a research on the ways of reaction that universities apply in relation to competency requirements of enterprise architects (Gala and Jandos 2010). The results show that university courses provide only a partial knowledge for the enterprise architect's profession. Therefore, a greater effort must be made to prepare undergraduate and graduate courses concentrated on building enterprise architect skills.

A support of architectural modelling constitutes another important condition for effective enterprise architecture adoption. A modelling tool must support both organisation-level and project-level models with full traceability between them; offer a wide range of models, such as the UML, database models, business-process models; and support version control and configuration management of the models (Temnenco 2007).

5.2 *Mapping of ISO/IEC 29110 Standard to Enterprise Architecture*

On the side of Solution Development Methods, it is worth taking into account the relationship to EA as well. It is a great opportunity to map the ISO/IEC 29110 standard for very small entities to EA as I am a member of the working group WG24 which develops this standard. In these days, we are working on the Intermediate and Advanced Profiles where relationship to EA is significant. Within WG24, I am going to propose to describe the relationship to EA, map VSE profiles to EA frameworks and develop a roadmap for VSEs showing how to build EA by implementing software process standards. Following this procedure, that is, first implement software process methods or standards and then enterprise architecture is recommended, and as Temnenco shows, “organizations routinely using project management and software lifecycle methodologies, such as PMBOK, RUP, Scrum are typically more successful at implementing Enterprise Architecture” (Temnenco 2007).

6 Conclusions

In this chapter, enterprise architecture and software process improvement initiatives were presented as an instrument capable of bringing business benefits to small and medium companies. More importantly, utilisation of synergy of both approaches was proposed. On the side of EA, the enterprise architecture for small and medium enterprises (EAFSME) framework was presented. On the side of software process improvement, a need of mapping Intermediate and Advanced Profiles for VSEs to EA was proposed. However, there is now a lot of work to be completed such as development of EAFSME profiles and their implementation in the Eclipse Process Framework Composer and also development of Deployment Packages.

Acknowledgments The work reported in this chapter was supported by the project GACR 403/10/0303 Enterprise Architecture as Management Principle for SMEs.

References

Anacleto A, von Wangenheim CG, Salviano CF, Savi R (2004) Experiences gained from applying ISO/IEC 15504 to small software companies in Brazil. 4th international SPICE conference on process assessment and improvement, Lisbon, Portugal, April

Deployment packages repository. Available from http://profs.logti.etsmtl.ca/claporte/English/VSE/index.html

DoDAF Working Group (2007) DoD architecture framework, vol I: Definitions and guidelines (Version 1.5), 23. The United States Department of Defense, April 2007. http://www.defenselink.mil/cio-nii/docs/DoDAF_Volume_I.pdf

Ernst AM (2008) Enterprise architecture management patterns. In: 15th conference on pattern languages of programs, Nashville, TN, USA, 18–20 Oct 2008. ACM 9781605581514

Gala L, Jandos, J (2010) Enterprise architecture based innovations: competencies perspective. In: IDIMT-2010 information technology – human values, innovation and economy. Trauner, Linz, s. 33–40. ISBN: 978-3-85499-760-3

Gartner Enterprise Architecture Framework: Evolution 2005. http://www.gartner.com/DisplayDocument?doc_cd=130855&ref=g_fromdoc

CIO (Chief Information Officer) Council (2001) A practical guide to federal enterprise architecture. Technical report. Federal CIO Council, Springfield, VA, USA

ISO 9001: Quality management systems – requirements, 2008

ISO/IEC 12207: Systems and software engineering – software life cycle processes, 2008
ISO/IEC 15289: Systems and software engineering – content of systems and software life cycle process information products (Documentation), 2006
ISO/IEC 15504: Information technology – process assessment, 2004
ISO/IEC 29110-1 (2010) Software engineering – lifecycle profiles for very small entities (VSE) – Part 1: VSE profiles overview. International Organization for Standardization (ISO), Geneva
ISO/IEC 29110-2 (2010) Software engineering – lifecycle profiles for very Small entities (VSE) – Part 2: Framework and taxonomy. International Organization for Standardization (ISO), Geneva
ISO/IEC 29110-3 (2010) Software engineering – lifecycle profiles for very small entities (VSE) – Part 3: Assessment guide. International Organization for Standardization (ISO), Geneva
ISO/IEC 29110-4 (2010) Software engineering – lifecycle profiles for very small entities (VSE) – Part 4: Specifications of VSE profiles. International Organization for Standardization (ISO). Geneva
ISO/IEC 29110-5 (2010) Software engineering – lifecycle profiles for very small entities (VSE) – Part 5: Management and engineering guide. International Organization for Standardization (ISO), Geneva
ISO/IEC 42010:2007 Systems and software engineering – recommended practice for architectural description of software-intensive systems. ISO/IEC, 2007
Kunstova R (2011) Assumptions for business and IT alignment achievement. In: ICDIPC 2011, Part I, CCIS 188. Springer, Berlin/Heidelberg, pp 320–332. ISBN: 978-3-642-22388-4
Laporte CY, April A, Renault A (2006) Applying ISO/IEC software engineering standards in small settings: historical perspectives and initial achievements. In: Proceedings of SPICE conference, Luxembourg
O'Connor RV, Laporte CY (2010) Towards the provision of assistance for very small entities in deploying software lifecycle standards. 11th international conference on product focused software development and process improvement (Profes2010), Hosted by LERO, Ireland, 21–23 June 2010
Rosen M (2011) Delivering value with enterprise architecture: a survey of today's best practices and programs, Cutter Consortium
Temnenco Vitalie TOGAF or not TOGAF: Extending enterprise architecture beyond RUP, developer works, 2007. http://www.ibm.com/developerworks/rational/library/jan07/temnenco/index.html
TOGAF – The Open Group Architecture Framework (Version 9 "Enterprise Edition"). The Open Group. http://www.opengroup.org/togaf/
Vorisek J, Jandos J, Feuerlicht J (2011) SPSPR model – framework for ICT services management. J Syst Integr 2(2):3–10. ISSN: 1804-2724
VSE Education Special Interest Group. http://profs.logti.etsmtl.ca/claporte/English/VSE/VSEEducation
Zachman JA (1987) A framework for information systems architecture. IBM Syst J 26(3). IBM Publication G321-5298

Chapter 21
A CMDB Meta Model Based on Services

Javier Saenz, Eugenio Fernandez, and Mercedes De La Camara

1 Introduction

No one doubts now that ICT is a key factor for helping organizations to achieve their business goals, but it is also true that they have the need to properly govern and manage their ICT in order to tightly integrate them with the business as the way to reach the best results.

More and more, ICT government is being talked about, and there are standard such as ISO/IEC 38500 to guide organizations in implementing a good governance of their ICT, but there is also a lack of knowledge of the role they really play: what is considered ICT in the organization? How are they seen by the management staff? Do they add value to the business? Are they strategic? Do they have a clear impact on productivity? All these questions should be answered affirmatively before considering establishing a good governance of ICT in any organization.

It seems obvious that ICT-productivity relationship, although there is no clear evidence, as shown in studies by several authors (Willcock and Lester 2004; Levy and Powell 1998; Brynjolfsson and Hitt 1998 or Dewan and Kraemer 1998), does not automatically increase productivity but is essential to build a working system that will increase it. Sieber (2004) argues that ICT can generate competitive advantages based on the proper use of information, developing new uses from it and improving its flows between the different actors. More specifically, in Henderson et al. (1993), a model (SAM, Strategic Alignment Model) for strategic management of ICT was proposed that was used by other authors (Burn and Szeto 2000; Smaczny 2001; Avison et al. 2004), based on four factors:

– Business strategy
– Organizational infrastructure and processes
– ICT strategy
– ICT infrastructure and processes

and was lately refined by Luftman (2000, 2002, 2003a, b) and Guldentops (2003).

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_21,

However, up to half of the past decade, ICT management was almost exclusively centered on infrastructure, without receiving the slightest information by the organization with regard to their business objectives or benefits to the business and, therefore, without a business vision. This policy, which did not allow to obtain any appraisal of the influence of ICT on the organizations and not even consider whether investment in ICT was adequate or not, resulted in harmful behaviors such as excessive shopping infrastructure in anticipation of possible unplanned demands and gave way to the introduction of rational management models that would allow some way forward in the right direction by using appropriate mechanisms for the management of ICT. In this new context, appeared different frameworks, such as COBIT (ITGI 2004) or ITIL (OGC 2001a, b), based on process management that are guides and good practices on security, financial management, continuity management, incident management, and others. With good management processes, it is also possible to start to measure individually relevant aspects of ICT that can bring light to the governing bodies, such as user satisfaction or response time in resolving incidents or problems. It is at this stage where we can place, in general, the current state of ICT in the organizations so, more or less settled, the policy to follow is that of implementing adequate mechanisms for the management of ICT.

Considering the ICT management as our reference framework, ITIL is the most widely known and used framework for managing IT services and infrastructure (Johnson et al. 2007), and it has becoming the de facto standard in service management (OGC 2006). One of the key processes in ITIL is configuration management, which in V3 is service asset and configuration management (SACM), whose objective is to define and control the components of services and infrastructure and maintain accurate configuration information on the historical planned and current state of the services and infrastructure (OGC 2009). Although it is included in the service transition stage, it covers all the service lifecycle, and it is very important for optimizing the performance of service assets and the costs and reducing the risks caused by poorly managed assets. As indicated by Klosteboer (2008), the configuration management is the center of the universe of ITIL information, as it gathers, manages, and connects critical information for the rest of the processes.

Configuration management is supported by the configuration management system that manages the information about all the configuration items (CI) and their relationships and uses a configuration management database (CMDB) that could also be a federation of CMDBs.

For all kind of organizations, the CMDB is the core which supports the integrity of their ICT and the information necessary for its proper functioning, especially for large companies, with a high level of dependence on ICT for their operation. Small and medium organizations virtually have the same necessities, but actually the things are slightly different: they do not have the money and the knowledge to implement either a configuration management process or a CMDB, and there are no adequate tools in the market for them. Large manufacturers of software tools

(BMC, CA, ASG, IBM, etc.) offer tools for configuration and CMDBs, but always focused on large organizations, although the description of their products affirms their general purpose. A small or medium organization cannot afford the expense of buying a tool like this, nor has the need to indicate their purchase or has qualified personnel to its management. There are versions of open-source CMDBs in the market (OneCMDB; ProcessWorx) that does not entail any cost and could be useful for this type of organizations, but we cannot say that their presence in the market can be taken into account today.

2 Meta Model

There are several authors which have presented studies about configuration management from different points of view. Jonassen (2002) raises several products, projects, organizations, companies, processes and tools, but all of them are mainly based on the technology. Singer (2008), in his proposal for a meta model for CMDB, believes, like most authors, that the CI is the main element on which to build a CMDB and poses an organization based on a model of three layers: services, systems, and components. One of the expressions used in the ICT world in recent years, "business alignment," has been replaced by a more accurate expression that better reflects the idea of collaboration between the business and ICT: "integration with the business." We think that this idea is important for many reasons, the most obvious is that ICT is meaningless in itself but as an element it adds value to the business, providing the necessary services and supporting business processes, while business services could not exist in today's world without ICT. Therefore, we think that an organization of the CMDB by reference to the business service and, from there, built with the configuration elements of information systems that support these services can be more rational while being clearer in its organization. The CMDBs currently on the market pose a solution based on the CI, and although logically integrating all components based on a network of relationships, among them is what determines the CMDBs organization.

The solution we propose, although it came up with the idea of trying to meet the needs of configuration management for those small and medium organizations, could also be used as a reference model for large organizations and is based on a clearly differentiated layer structure. Taking into account the Singer's meta model, we propose another one of five layers with two types of connections: between adjacent layers and between CIs in a layer, and the service at the apex, as explained below. We believe that it is simple, versatile, scalable, and can be optimally adjusted to the needs of the very different companies, as indicated. It applies not only to very few workers' micro-companies where their ICT is practically reduced to a basic infrastructure but also for others who provide services relying on it. Although designed thinking in the business service as a unifying element, his vision is also valid from the infrastructure point of view, that is, you can have an abstract view from the services in which the other components are organized as necessary parts to provide those services, or from the opposite point of view, that is, the infrastructure components which support the other layers. Any choice within this range can be seen in the meta model and adapted as the business needs are growing. From this meta model, and by a configuration application, you can build the CMDB "tailored to fit" the needs of the company itself.

A CMDB should collect all the elements involved in the information system, their relationships, and how they affect each other. These relationships can be simple or can become very complex, with relationships *n*:*m* between two CIs, and they in turn can be physical or logical. A clear example is a hard disk that is used to store a database of information concerning a sales service provided by the company; however, it may also have information for other business services. This structure makes the CMDB relationships, if properly managed and well used the information contained in it, a very powerful tool of knowledge about the information system, while being in a weak position because of the effort required to keep the information updated whenever there is any change in any of its components. If, as noted in the introduction, the service is considered as the backbone for organizing the components of the CMDB, the meta model we propose is shown in Fig. 21.1.

It is an approach based on five layers, each representing a different type of CI, which establishes the relationship between adjacent layers, with the exception of the application infrastructure layer that is considered optional in the meta model. Besides the layers, some elements are shown; although it cannot be considered proper infrastructure, they should be part of the CMDB and be related to different layers. This model allows a view of the CMDB from their layers, that is, they can establish a taxonomy of information stored in it by everyone of them. The adjacent layers are linked in both directions, so that downward relations are obtained by reference to the service, and upward, the view obtained is that of CIs. Each layer is related to the nonadjacent through its superior or inferior. Within each layer, the elements may be related among them, and this will be common in some of them. The following describes each layer:

A. Business services layer

It represents all the business services the company provides to its customers. Each business service can be used by a set of users and can be regulated by one or more SLAs. A business service can be supported by one or more applications, or what we call application infrastructure, and will be used by users and managed by employees of the organization.

B. Application infrastructure layer

The application infrastructures are groups of interrelated applications that support the service. An example of this is the internet banking service provided by banks, which offers a set of "subservices" supported by different applications: management of accounts, transfers, values, and so on. Therefore, this layer is related to the service and the applications in which it is

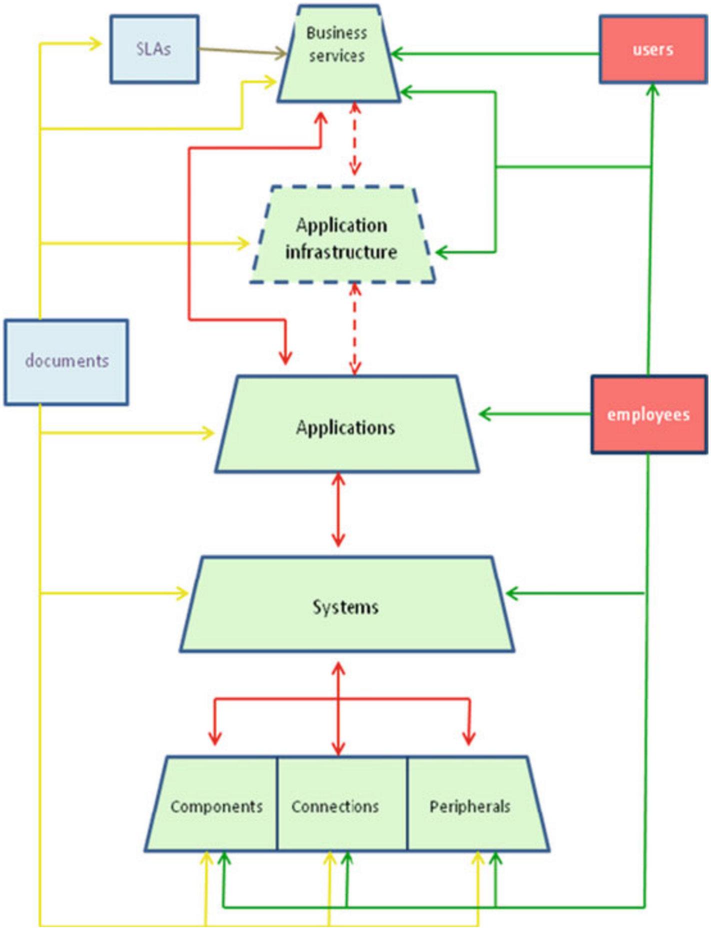


Fig. 21.1 Meta model representation

based. It was considered optional in the model because it will not always be necessary.

C. Application layer

It provides the technological support necessary to ensure the proper functioning of services. Always run-on elements of the systems layer can be grouped, from a logical point of view, with others to form an application infrastructure, as mentioned above. Applications can be diverse: tailor-made, systems, databases, and so on.

D. Systems layer

Systems provide runtime environments for applications and rely on the physical components of the ICT infrastructure.

E. Physical components layer

There are specific hardware elements that make up the systems and networks that connect them. Component is any element that is hosted within a machine (hard drive, graphics card, etc.), peripherals are external components (printer, monitor, card reader, etc.), and connection is any element that binds together two or more systems (cable, local network, router, etc.).

F. Users

They use the business services offered by the company and may be internal (employees) or external (customers).

G. Staff

They are employees of the company and will be those on which the task of keeping the CMDB updated will lie and the beneficiaries of the knowledge stored in it. They have access to all CIs.

H. SLAs

These are legal documents for regulating the service that a company pays to another. They determine the conditions under which the service is provided, the commitments made by both sides, and other information relating to the service.

I. Documents

These are the manuals or documents of different types associated with different CIs or that contribute to clarify some information concerning them.

3 Data Model

The data model that supports the meta model is based on the description of the different CIs that represent the elements of the layer to which they belong, each with its own characteristics, and all relationships between them, both those established between the internal layers and those within a layer.

From this entity-relationship model, and by an application that is explained in the next point, the database tables adapted to the model required by every customer will be defined and configured.

4 Model Generation

The idea is to have an application, which we call "model generator," to define a model from the general meta model described. As explained before, this idea came to fill the gap that exists in the market for configuration management tools for small

Fig. 21.2 Model generation

and medium organizations or for those that require a more flexible model. Given that the needs for information systems of those organizations can be very different, this application for model generation will allow each company to select the layers and configuration items that better describe the needs of the company, from which it will generate the operational database for the model chosen, so that the result will be a "tailor-made CMDB" to serve as a basis for the operational configuration management. In a way, we can say that every company can adapt its CMDB according to its maturity level. The more the maturity, the higher the layer it will reach in the meta model.

Once the database is obtained using the model generator, it will be the basis for the operational management of the CMDB itself, its updates, and queries. To this end, once the meta model is defined and, from it, the complete tables representing different CIs and their relations, the application will generate the specific CMDB according to the customer's needs to show the full set of CI model according to the scheme of layers, their relationships, and characteristics and allow the user to choose which layers better represent their IT structure (and therefore the CIs). Then, for each CI, the features are selected. Once the CMDB is configured, the application will create the real tables that implement it and which will serve as input for operational management. Figure 21.2 shows the pattern explained: the meta model is the entry for the model generator, which generates the particularized CMDB that serves as the basis for the actual management of the CMDB.

5 Conclusion

In this chapter, within the framework of ITIL and the idea of filling the gap left by the major configuration management tools manufacturers in terms of tools to support those small and medium organizations with different necessities from those large companies, on the one hand we propose a meta model of CMDB that is simple, versatile, covering the needs of configuration of ICT and that, considering the business service as the element on which your organization is structured, can present alternate views of that model: the above-mentioned services, a view from the ICT infrastructure itself and from the intermediate layers. Furthermore, we propose the deployment of an application that, from the meta model described, generates the actual CMDB tailored to the needs of the information systems of the company and serves as a basis for the operational configuration management.

It is a simple meta model that is better suited to the business point of view, and therefore the idea of integrating ICT with it, and that allows for service logical views of the information contained in the CMDB, or what is the same, to establish a taxonomy or logical organization of information services at the same time without losing the importance of CIs for a CMDB. We can therefore conclude that it is a simple and versatile, yet complete meta model which may well serve to meet the initial objective of supporting the necessities of those non-large organizations and still be a valid meta model for these ones.

References

Avison D, Jones J, Powell P, Wilson D (2004) Using and validating the strategic alignment model. J Strateg Inf Syst 13(3):223–246. BPMSpace: http://www.bpmspace.org

Brynjolfsson E, Hitt L (1998) Beyond the productivity paradox. Commun ACM 41(8):49–55

Burn JM, Szeto C (2000) A comparison of the views of business and IT management on success factors for strategic alignment. Inf Manage 37:197–216

Dewan S, Kraemer K (1998) International dimensions of the productivity paradox. Commun ACM 41(8):56–62

Duffy J (2002) IT/business alignment: is it an option or is it mandatory? IDC document # 26831

Guldentops E (2003) IT governance: part and parcel of corporate governance. CIO Summit. European Financial Management and Marketing Conference, Brussels

Henderson JC, Venkatraman N, Oldach S (1993) Continuous strategic alignment. Exploiting information technology capabilities for competitive success. Eur Manage J 11(2):139–149, Bus Q 55(3)

Johnson M, Hately A, Miller B, Orr R (2007) Evolving standards for IT service management. ACM, 2007

Jonassen AM (2002) Configuration management principles and practice. Addison Wesley, Boston

Klosteboer L (2008) Implementing ITIL configuration management. IBM Press, Upper Saddle River

Levy M, Powell P (1998) SME flexibility and the role of information systems. Small Bus Econ 11:183–196

Luftman J (2000) Assessing business-IT alignment maturity. Commun AIS 4:1–50

Luftman J (2002) Assessing business-IT alignment maturity. In: Van Grembergen W (ed) Strategies for information technology governance. Idea Group Inc, Hershey, pp 99–128

Luftman J (2003a) Competing in the information age – align in the sand. Oxford press, Oxford

Luftman J (2003b) Measure your business-IT alignment. The longstanding business-IT gap can be bridged with an assessment tool to rate your efforts. Opt Mag (26). (http://www.optimizemag.com/showArticle.jhtml?articleID=17701026)

OGC (2001a) Service support

OGC (2001b) Service delivery

OGC (2006) Best practice for service support. The Stationery Office thirteenth impression

OGC (2009) ITIL V3 service transition

OneCMDB. http://www.onecmdb.org/wiki/index.php?title=Main_Page

ProcessWorx. http://www.zdnetasia.com

Sieber S (2004) TIC: cuanto mejor las usas, más importantes, p. 17. www.computing-es.com 28 de abril 2004

Singer J (2008) Understanding the ITIL CMDB meta model. TDAN, February, 2008

Smaczny T (2001) Is an alignment between business and information technology the appropriate paradigm to manage IT in today's organizations? Manage Decis 39(10)

Willcock L, Lester S (2004) Beyond the TIC productivity paradox. John Wiley series in Information systems the business perspective book, vol 1: The IS view in delivering services to the business

Chapter 22
Tool Support for the Quality Assessment of MDWE Methodologies

F.J. Domínguez-Mayo, M.J. Escalona, M. Mejías, and J. Ponce

1 Introduction

An interesting concern in development teams of Web applications is how to build Web applications automatically, with the least cost and the best quality. In this line, the model-driven engineering (MDE) approach to software development focuses on creating models or abstractions closer to some particular domain concepts rather than computing concepts. It is meant to increase productivity by maximizing compatibility among systems, simplifying the design process and promoting communication between individuals and teams working on the system.

The rise of this paradigm has already influenced current Web developments. The most important research initiative in this area is the Model Driven Architecture (MDA) (OMG: MDA), which is being developed under the umbrella of the Object Management Group (OMG). According to this group, models may be independent from the characteristics of any technological platform. Besides, the life cycle of a software system is completely covered, starting from requirements capture, passing through code generation up to the system maintenance. In this context, Web engineering is a specific domain where model-driven software development can be successfully applied (Escalona and Aragón 2008). The use of MDE in Web engineering is called model-driven Web engineering (MDWE), and as it is noticed in different papers (ISO- International Organization for Standardization, ISO/IEC 9126-1), in the last years, several research groups have proposed different methodologies with processes, models and techniques to build applications such as UWE (UML-Based Web Engineering), WebML (The Web Modelling Language), OOHDM (Oriented Hypermedia Design Method), OOH4RIA (Meliá et al. 2010), Rux-Method (Preciado et al. 2008) or NDT (Navigational Development Techniques). Some of them involve most of the levels of abstraction, and they

F.J. Domínguez-Mayo (✉) • M.J. Escalona • M. Mejías • J. Ponce
Department of Computer Languages and Systems, University of Seville, Seville, Spain
e-mail: fjdominguez@us.es; mjescalona@us.es; risoto@us.es; josepg@us.es

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_22,

even have tools that support the automation of transformations in the development processes. A suitable design is needed as well as updated MDWE methodologies and effective tools. To this end, our work focuses on evaluating and comparing existing proposals, bearing in mind that the framework could be extended to other areas in the future. In order to offer a suitable tool support for evaluating the quality of approaches, this chapter proposes a tool support for QuEF (quality evaluation framework) which is a framework for the quality evaluation of MDWE methodologies.

This chapter is organized into the following sections. Following this introduction, Sect. 2 offers an overview of QuEF. Section 3 describes the tool support description and basic proposed interfaces together with the users' context. Finally, Sect. 4 states a set of conclusions and contributions and proposes possible future work.

2 An Overview of QuEF

QuEF (Domínguez-Mayo et al. 2010a, b, c) is a framework for the assessment of MDWE methodologies. This approach is oriented towards the quality evaluation of MDWE methodologies in a specific environment by means of objective measures. Given the high number of methodologies available and proposed in recent years, it has become necessary to define objective evaluation tools which enable organizations to improve their methodological environment and help Web methodology designers to create new effective and efficient tools, processes and techniques. Due to the progressive evolution of methodologies, it is not only necessary to evaluate quality but also to find out how to improve and optimize this process in order to reduce costs. In this work, an approach or methodology for the development of Web applications is a model-driven proposal. It provides a set of guidelines, techniques, processes and/or tools for structuring specifications which are expressed as models.

There are only considered Web modelling approaches based on MDA in the framework. In addition, a framework is a basic conceptual structure composed of a set of elements used to evaluate, in this case, MDWE methodologies, although it could be extended to other area or domain. It consists of the following elements:

- *Quality Model*. It includes the basis for quality requirements specifications with the purpose of evaluating quality. It specifies each element and its purposes.
- *Analysis Process*. It includes the description templates for the evaluation of methodologies and defines the importance of each element. It depends on the Quality Model description.
- *Evaluation Process*. It includes the definition and specification to carry out the quality evaluation process.
- *Multi-Criteria Group Decision Making*. It includes the necessary elements to better the standardization of the access channel and communication among users

of different MDWE methodologies. It also reaches a consensus in the definition, the weight value importance of each definition and weight value influence relationships among the different elements.

3 The Proposed Tool Support for QuEF

Figure 22.1 shows the general system architecture of QuEF-TS (QuEF Tool Support) for each component. In this figure, the Quality Model component is shown together with each element and the relationship among them. According to the Information Needs description, a Feature contains a set of Sub-Features that have a set of Metrics to measure them. Additionally, as Quality Aspects, Quality Characteristic includes a set of Quality Sub-Characteristics that have to be associated to the Sub-Features.

The Quality Model component in QuEF-TS has a set of elements to be defined. Consequently, we have to determine a set of Features, Sub-Features, Metrics, Quality Characteristics, Quality Sub-Characteristics and the relationships between these Sub-Features and Quality Sub-Characteristics. All these elements and their relationships provide the basis for the quality requirements specification and evaluation in a specific domain (in our case, MDWE). Features, Sub-Features and Metrics help to define the Information Needs. In other words, they conform to the information requirements of the evaluator, clients and other pertinent audiences to

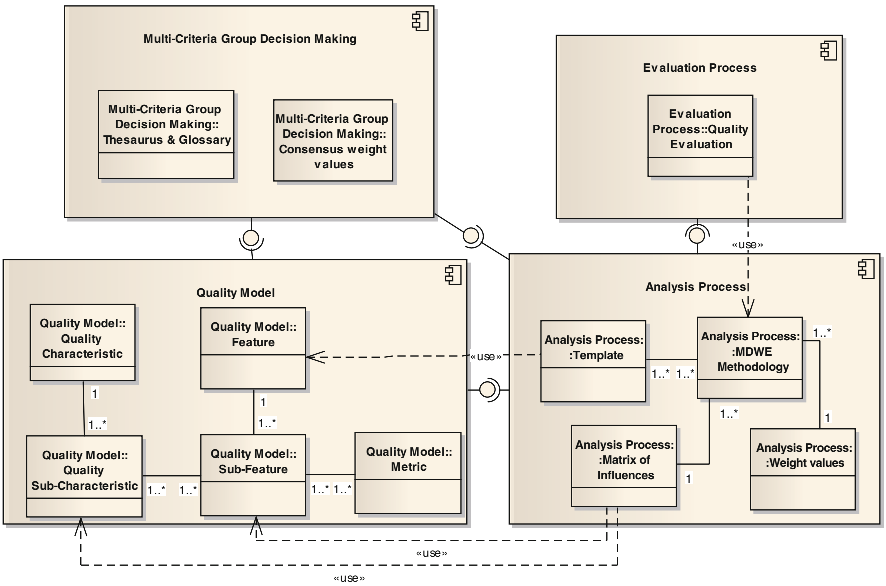


Fig. 22.1 The general system architecture of QuEF tool support

be met by means of the evaluation. Quality Characteristics and Quality Sub-Characteristics define the quality aspects of the required model. Finally, these Information Needs have to be mapped with the quality aspects.

- *Quality Characteristic*. This is a higher-level quality aspect. For instance, Usability, Maintainability, Functionality, Reliability or Portability could be considered Quality Characteristics.
- *Quality Sub-Characteristic*. In a hierarchy of quality aspects, this is a lower-level quality aspect. For instance, Usability is defined through several Quality Sub-Characteristics, such as Learnability, Understandability and Operability.

Each characteristic in ISO/IEC 9126 (ISO- International Organization for Standardization, ISO/IEC 9126-1) and ISO/IEC 25000 (SQuaRE) is described in relation to a software product. However, in our particular case, all Quality Characteristics and Quality Sub-Characteristics are described regarding approach characteristics.

In MDWE, models are progressively refined and transformed into new models or codes. To this end, tools may also be useful to test, verify or validate models. Moreover, each methodology may define its development process and/or techniques. The quality of methodologies subsequently depends on the diverse Features, such as the *MDE*, the *Web Modelling*, the *maturity* of a methodology and the *tool support* applied to discover faults or weaknesses. The main idea is to characterize the whole MDWE process. The elements describing the Information Needs in the Quality Model are:

- *Feature*. This is a higher-level description concept of an approach. It may be, for example, the software development process, models, metamodels, languages, tools or the transformations used.
- *Sub-Feature*. This is a lower-level description concept of an approach. For example, the model-driven engineering is a Feature with several Sub-Features, such as the Language Definition, Transformations and Trace.
- *Metric*. In the Quality Model, metrics should measure the degree to which a Sub-Feature reaches the required model. To simplify, a metric is used for measuring Sub-Features.

The Analysis Process component would include the templates which have to be generated in terms of Quality Model description, specifically the Information Needs description. The methodology can be analysed with these templates considering the Weight value definition of Sub-Features and Quality Sub-Characteristics and the Matrix of Influences (MoI), which indicates the influence in Sub-Feature and Quality Sub-Characteristics. Finally, the Evaluation Process component performs the quality evaluation, and the Multi-Criteria Group Decision Making is responsible for reaching a consensus on the terminology and weight values among users.

Different basic interfaces are proposed and given in this chapter.

As far as user context is concerned, this tool support would have three different user contexts for QuEF-TS. As it is shown in Fig. 22.2, the first one would be the Developer user, who cannot modify anything. It would have a general view of the

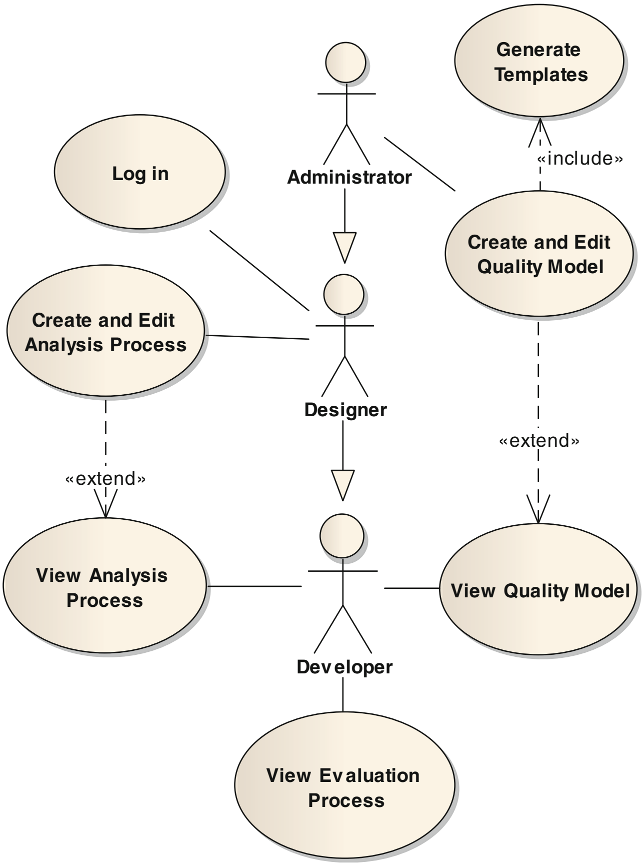


Fig. 22.2 Use cases for the proposed tool support

Quality Model, the Analysis Process and the Evaluation Process. The second one would be the Designer user, who would have more permission and could not only view but also create and edit the Analysis Process components such as the MoI, the Weight values for Quality Sub-Characteristics and Sub-Features and Template values. This user could also log in the tool support. Thus, it would save different MoIs, Weight values and Templates and exchange them with other Designers. It would be useful to evaluate methodologies according to other designers' points of view. Finally, the third user would be the Administrator who would create and edit the Quality Model and generate the templates based on it in order to be used in the Analysis Process.

QuEF-TS can generate templates and MoIs in terms of the Quality Model specified. Thus, an important issue is to define the Quality Model and its knowledge.

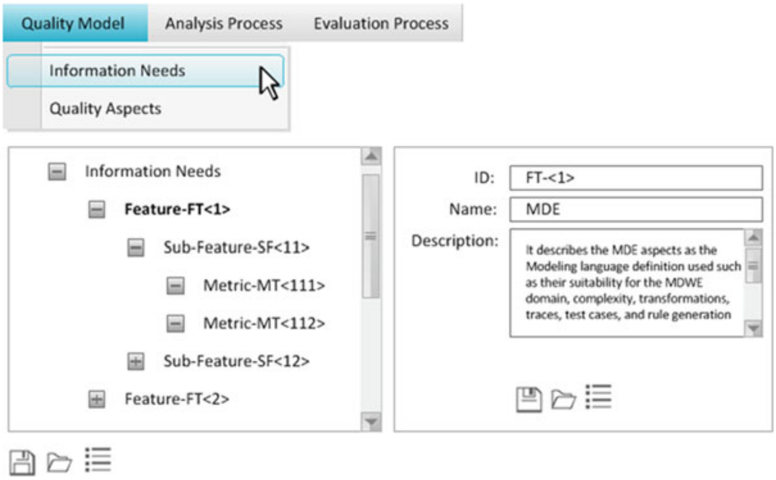


Fig. 22.3 Basic interface for information needs description

3.1 The Quality Model

Figure 22.3 represents the basic interface of Information Needs. This interface shows how the Features, Sub-Features and Metrics are defined. In order to build the different elements, a tree structure is proposed and used by the Administrator with the aim of creating and editing Features, Sub-Features and Metrics in terms of the Information Needs established by the community of designers. The definition of this hierarchy is currently being developed, although we are also working on Multi-Criteria Group Decision Making (MCGDM) (Preciado et al. 2008) to reach a consensual definition of the hierarchy.

A set of Quality Characteristics and Quality Sub-Characteristics for each Quality Characteristic based on current literature, such as ISO/IEC 9126 or other standards which are adapted to MDWE methodologies, are identified, classified and placed in a hierarchy. Likewise, the Quality Characteristics and Quality Sub-Characteristics are defined in the Quality Model, as shown in Fig. 22.4. On its left side, this figure shows a tree structure where the user would create and edit Quality Characteristics and Quality Sub-Characteristics in terms of the Quality Aspects considered. Furthermore, in order to define a Quality Model, it must contain association links between the Sub-Features and the Quality Sub-Characteristics. These links represent the dependencies between Sub-Features and Quality Sub-Characteristics and show how each Sub-Feature affects every Quality Sub-Characteristic. Association links may be based on proven real-world experience or determined by real case study applications in a number of real projects. We are currently working on a MCDGM method which could permit reaching a consensus value according to different experts' views.

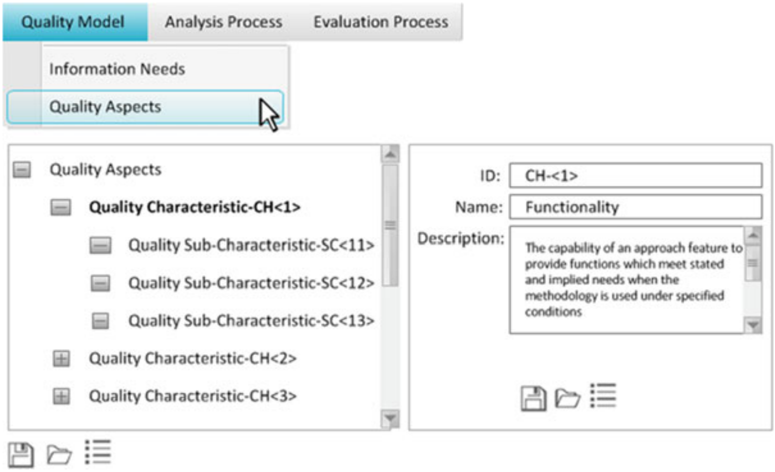


Fig. 22.4 Basic interface for quality aspects description

3.2 The Analysis Process

In this component, the association links between Sub-Features and Quality Sub-Characteristics are defined in the MoI, which indicates, by means of a figure or value, how every Sub-Feature affects Quality Sub-Characteristic.

For instance, Usability is described as a set of Quality Sub-Characteristics. As shown is Fig. 22.5, these Quality Sub-Characteristics could be affected by one of the Sub-Features. This figure also represents the basic interface for the MoI. Each value indicates the influences between Sub-Features and Quality Sub-Characteristics appearing in this figure for every Sub-Features of MDE, such as Levels of Abstraction, Standard Definition, Model-Based Testing, Transformations and Traces, as well as for Quality Sub-Characteristics of Usability such as Learnability, Understandability, Simplicity and Interpretability.

The templates in the *Analysis Process* are based on the Quality Model definition, and they are used to describe an input methodology. Figure 22.6 represents the suggested basic interface. These templates would be used as input in the environment, analysed in the evaluation process and compared with the required quality model of the Quality Model definition. Templates for MDE, Web Modelling, Tool Support and Maturity have already been developed. Finally, the *Evaluation Process* would contrast the information from each input approach template with the information from the Quality Model definition. The main evaluation purpose is to identify trade-offs and sensitivity points of the methodology under study. The aim is to determine which aspect needs to be improved on a MDWE methodology.

Each Sub-Feature has to define a set of quantitative or qualitative metrics used to measure each Sub-Feature. In these lines, it might be interesting to establish standard metrics on MDWE to centralize them. In the literature, numerous

Fig. 22.5 Basic interface of the matrix of influences (*MoI*)

references to metrics can be found, but standardization has not been carried out yet. The types of metric defined are either qualitative or quantitative. Qualitative metric range indicates if the metric is Supported, Partly Supported or Not Supported, as appeared in Fig. 22.6.

On the contrary, Fig. 22.7 offers the Weight values setting for the Information Needs and Quality Aspects. Weight value represents the importance of each Sub-Feature in the set of Sub-Features for a Feature. Similarly, a weight value is also used to define the importance of each Quality Sub-Characteristic in the set of Quality Sub-Characteristics for a Quality Characteristic. MGDCM methods are being currently developed to define weight values for Sub-Features and Quality Sub-Characteristics.

3.3 The Evaluation Process

The Evaluation Process is the component which users can evaluate whether they have reached the required model of quality or not. The proposed basic interface, as shown in this example of Fig. 22.8, considers the Sub-Features of Web Modelling.

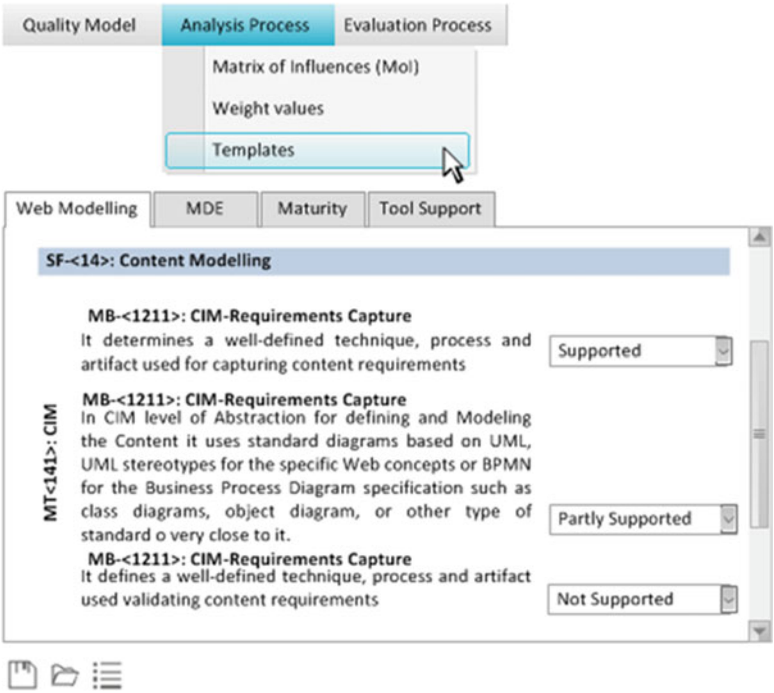


Fig. 22.6 Basic interface of the templates

We can see the Sub-Feature values for the Web Conceptual Levels, Interfaces, Development Process, Content Modelling, Presentation Modelling, Navigation Modelling and Business Modelling. All of them are Web Modelling Sub-Features. In other words, the views of all Sub-Feature values of a specific Feature or a view of all Feature values in general can be shown with this interface.

Quality Characteristics and Quality Sub-Characteristics are shown in the same way that Features and Sub-Features appeared, although in this interface, the user has to indicate which Features has to be considered in order to calculate the Quality Sub-Characteristics and Quality Characteristics.

The reason is that these values depend on the influences that Sub-Features have on Quality Sub-Characteristics. In other words, the value definition of Quality Characteristics and Quality Sub-Characteristics is determined by the relationship (s) between the Information Needs and the Quality Aspects. Figure 22.9 represents the basic interface for Quality Aspects and provides a general view of Quality Characteristics. We can also obtain all Quality Sub-Characteristic values for a Quality Characteristic by selecting the appropriate Data value option. In this example, we can observe all Quality Characteristic values in terms of Web Modelling, MDE and Maturity, which are the Features that have been selected in "In terms of" option.

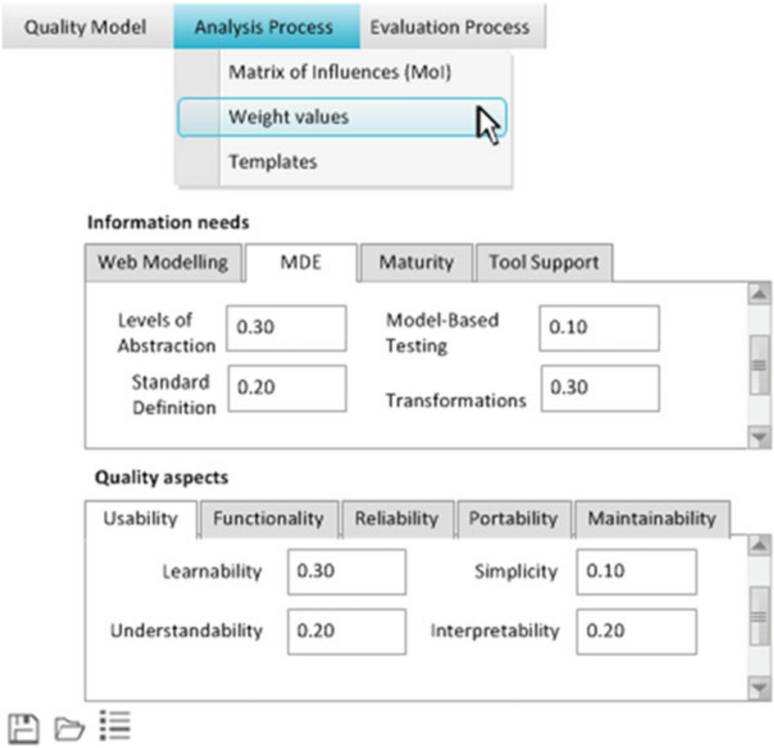


Fig. 22.7 Basic interface of the weight values setting

4 Conclusion

A tool support which implements the QuEF framework is needed for the improvement of current proposals and would be highly useful for the successful development of a new MDWE methodology. In this chapter, QuEF-TS, a proposed tool support to implement QuEF, is presented, and different basic interfaces are described. We consider that using this tool support will enhance the quality of products, processes and techniques of approaches because designers can analyse, control and improve the quality of their approaches easily. This tool can be used for the cost optimization in the improvement of methodologies since the Sub-Features selected to be implemented can be reduced by means of the Matrix of Influences (MoI) because we would take into account the minor number of Sub-Features with the major influence in Quality Sub-Characteristics. Thus, the use of QuEF-TS could improve the efficiency and effectiveness of MDWE methodologies and, consequently, may lead to a more widespread use due to the fact that this evaluation approach helps people to understand the strengths and weaknesses of a methodology. We consider it necessary to carry out a standardization of terminology to improve the access channel for communication on MDWE. In this context, we are

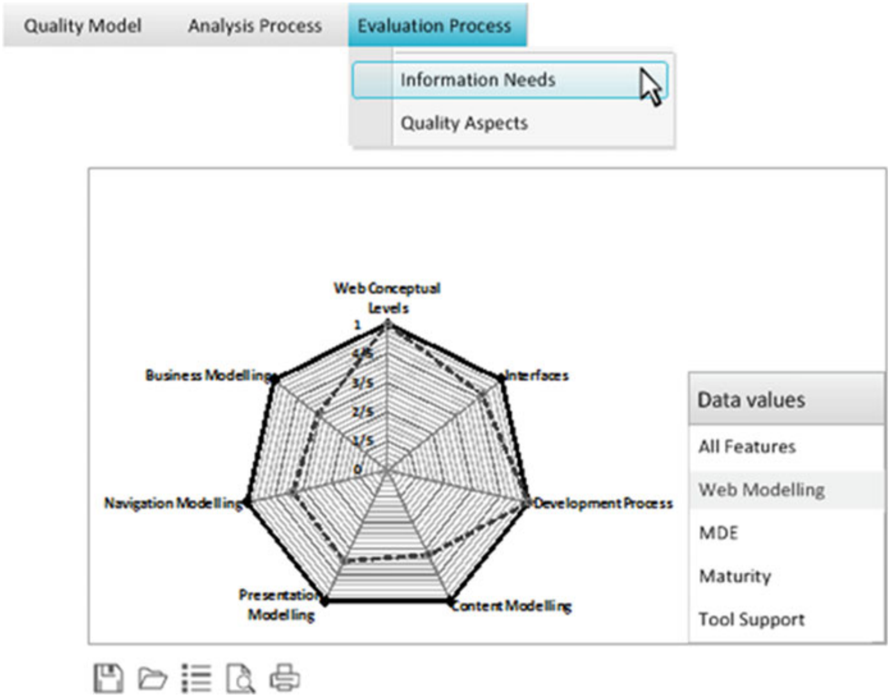


Fig. 22.8 Basic interface of information needs for the evaluation process

currently working in MCGDM methods to reach a consensus on methodology designers groups, as well as on the definitions of several aspects, such as Quality Model (Feature, Sub-Features, Metrics, Quality Characteristics and Quality Sub-Characteristics), Weight values and the influence of each Sub-Feature on each Quality Sub-Characteristics.

Other future work will be the integration of QuEF-TS in well-known social networks like Facebook (Facebook for developers) or Twitter (Twitter for developers), to make easier the MCGDM methods and reach this consensus more easily among different users.

Finally, another important aspect is that this environment could involve different areas or domains since QuEF-TS can generate the templates and all necessary elements based on the Quality Model description in order to analyse, control and evaluate the specified domain.

Acknowledgements This research has been supported by the project QSimTest (TIN2007-67843-C06_03) and by the Tempros project of the Ministerio de Educación y Ciencia, (TIN2010-20057-C03-02), Spain.

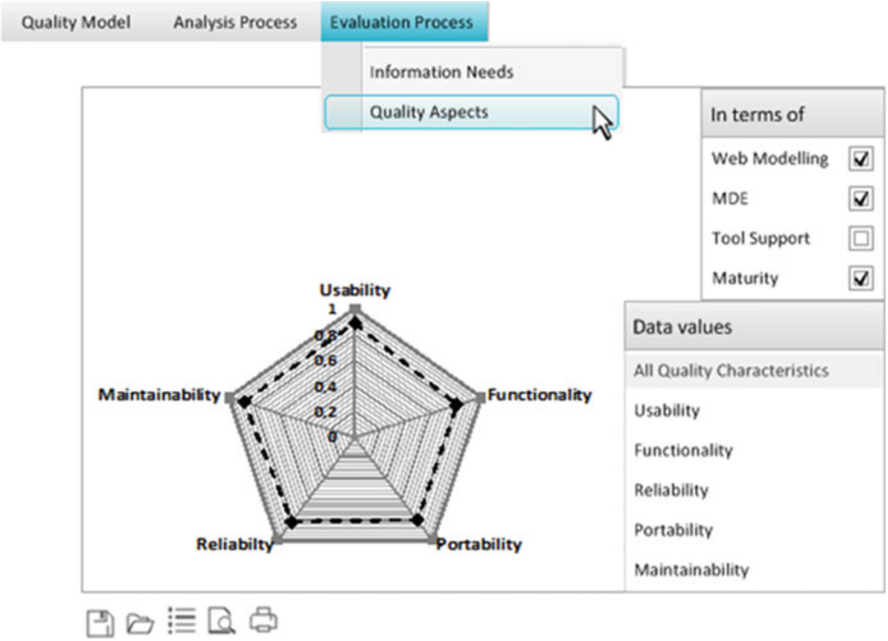


Fig. 22.9 Basic interface of the quality aspects for the evaluation process

References

Domínguez-Mayo FJ, Escalona MJ, Mejías M (2010a) QuEF (quality evaluation framework) for model-driven web methodologies. In: Proceeding ICWE'10, Proceedings of the 10th international conference on current trends in web engineering, pp 571–575. ISBN:3-642-16984-8 978-3-642-16984-7

Domínguez-Mayo FJ, Escalona MJ, Mejías M, Ramos I, Fernández L (2010b) A quality evaluation framework for MDWE methodologies. In: Proceedings of the 18th international conference on software quality management (SQM 2010), vol 1, No. 1, pp 171–184

Domínguez-Mayo FJ, Escalona MJ, Mejías M, Torres AH (2010c) Towards a quality evaluation framework for model-driven web engineering methodologies. In: Sixth international conference on web information systems and technologies (WEBIST 2010), vol 2, pp 191–194

Escalona MJ, Aragón G (2008) NDT. A model-driven approach for web requirements. IEEE Trans Softw Eng 34(3):377–390

Facebook for developers. Retrieved April 2011, from http://developers.facebook.com

ISO- International Organization for Standardization, ISO/IEC 9126-1. http://www.iso.org. Retrieved January 2010, from http://www.iso.org

ISO/IEC 25000:2005. Software engineering – software product quality requirements and evaluation (SQuaRE) – guide to SQuaRE

Meliá S, Pardillo J, Cachero C (2010) Automatic selection of RIA software architectures using quality models. Quality of information and communications technology (QUATIC), 2010. In: Seventh international conference on the quality of information and communications technology (QUATIC), pp 505–510. ISBN: 978-1-4244-8539-0

NDT (Navigational Development Techniques). Retrieved March 2011, from http://www.iwt2.org/en/ndt.php

OMG: MDA. Retrieved January 2010, from http://www.omg.org/mda/
OOHDM (Oriented Hypermedia Design Method). Retrieved March 2011, from http://www-di.inf.puc-rio.br/schwabe/HT96WWW/section1.html
Preciado JC, Linaje M, Morales-Chaparro R, Sánchez-Figueroa F, Zhang G, Kroiβ C, Koch N (2008) Designing rich internet applications combining UWE and RUX-Method. In: ICWE '08 Proceedings of the 2008, 8th international conference on web engineering. ISBN:978-0-7695-3261-5
Saaty TL (1983) Introduction to a modeling of social decision process. Math Comput Simul 25:105–107
Twitter for developers. Retrieved April 2011, from http://dev.twitter.com
UWE (UML-Based Web Engineering). Retrieved March 2011, from http://uwe.pst.ifi.lmu.de
WebML (The Web Modelling Language). Retrieved March 2011, from http://www.Webml.org

Chapter 23
EFQ_TIL: Relationship Model Between ITIL and EFQM to Assure Service Quality in IT Processes

Juan José Sánchez Peña and Eugenio Fernández Vicente

1 Introduction

Throughout history IT has played a very important role. Recently through automation and computerization of its management, IT has become an indispensable tool and a key to organizations, companies and public institutions.

IT service management (ITSM) is a process-based practice intended to align the delivery of information technology (IT) services (Galup et al. 2007) with the needs of the enterprise, emphasizing benefits to customers. ITSM (Bardhan et al. 2010) involves a paradigm shift from managing IT as stacks of individual components to focusing on the delivery of end-to-end services using best practice process models. Information Technology Infrastructure Library (ITIL) is a globally recognized collection of best practices for IT service management (Betz 2006).

Information is one of the main sources of business in the world; business generates large amounts of information. Its right use is important and strategic importance and should not be considered as one tool among many others.

Not so long time ago, the IT infrastructure simply provided support services primarily to databases, telephony, telecommunications and management applications, and sometimes they could be compared with office supplies, something important and essential for the proper functioning of the organization, but nothing else.

However, today this has changed and the IT services usually represent a key to the processes of the organization (Duffy 2002a).

The objectives of good management of IT services must be to:

- Provide adequate quality management
- Increase efficiency
- Align organizational processes and IT infrastructure

J.J.S. Peña (✉) • E.F. Vicente
Alcalá University, Alcalá de Henares, Madrid, Spain
e-mail: juanjo.sanchez@uah.es; eugenio.fernandez@uah.es

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_23,

- Reduce the risks associated with IT services
- Fulfil the objectives outlined in the organization

Today it is necessary to measure, evaluate and more importantly ensure and improve the quality of IT services. In this context, principles and quality management practices can be useful to meet these needs (Peterson 2003).

The organizations are often very dependent on their IT services and expect that these services not only support the organization but also provide new options for achieving its objectives (Van Der Zee and De Jong 1999). The provision of IT services involves the total management of IT infrastructure; it allows IT service providers wash one hands of the technology and can focus more on the relationship with its customers and the services they offer.

If we were talking about company's processes, ITIL aims through effective management of it that the product resulting from the production will have a more consistent quality for consumer satisfaction.

The quality of a service or product is evaluated once it is provided. The quality can be defined as the ability to achieve the desired operational objectives.

Therefore, the purpose of the quality is to provide to the customer a suitable offer with controlled processes while ensuring that this improvement does not result in additional costs. It is possible to improve a large number of problems at a low cost. However, when the perfection is closer, the costs are higher.

2 Adding Quality in IT Service Management Models

Under the premise taken today by organizations that IT can help substantially to achieve the goals set by the business, the challenge is to carry out a proper management of ITs within the organization in order, firstly, to measure and know in which level improvements occur in the objectives of the business through IT and, secondly, to direct them properly towards an alignment IT ← → business objectives that they allow to obtain the best results. In this context, it emerges the concept of government of IT as the framework more general and abstract to be working today, and under it will be placed a proper management of ITs.

However, as a general rule, these aspects are not addressed in models of existing IT governance, which are based on the principle that it requires a satisfactory answer to these questions, understanding that it is no sense to address it in opposite way: if the governing bodies believe that ITs do not add value, they are not strategic or they are not valued in the organization, then, why would be they governed? Some authors, as (Weill and Ross 2004a, b), have done studies on how to govern IT in organizations with large and scattered sets which are available to get a view of reality, although we must be careful not to extrapolate these studies to any organization because the profile of those included in such studies is very particular and restricted to organizations with a high degree of implementation of IT.

An appropriate model requires a holistic recognition of their complex and dynamic nature (Duffy 2002a, b; Patel 2003; Peterson 2003); a set of interrelated factors is necessary to address adequately the problem. In this sense, some authors (Peterson 2003; Van Grembergen et al. 2004; Weill and Woodham 2002) propose that the government of IT can be addressed using a mixture of structures, processes and relational mechanisms (components) interconnected correctly. A lower level would be a definition of an exact set of mechanisms that vary depending on many factors in each organization; there is no single model or standard (Patel 2003; Ribbers et al. 2002).

In the rest of this chapter, we will analyse the main existing approaches to governance and management of IT.

2.1 Action Frameworks

There are various action frameworks, models, methodologies, standards and guidelines of good practices that can be used to address good governance and a good management of IT. The question is to decide which and for what. In the next lines, we will use the term *framework* to refer to all of them in order to unify, independently if some are methodologies, other standards, and so on.

In summary, the following can be listed as advantages of standard applications (Oud 2005):

- It avoids reinventing the wheel again and again.
- It is a low cost of adopt it instead of making the development of a methodology.
- It facilitates the services outsourcing, enabling a common environment between the organization and the organization that is outsourced.
- It provides audit and control.
- It allows the use of universal indicators for evaluations across organizations.

About the first question (which), we must consider a frameset referenced in the literature: COBIT, ITIL, ISO/IEC 38500 and ISO 20000 and others less known: EFQM (which will be the one in which we will base our study), CMM, MOF or BS 15000. The second question (for what) is necessary to be tackled in terms of governance or management, so we must select the appropriate frameworks and develop appropriate IT governance or management. This is not a simple issue, since many of these frameworks can be used to address both domains, for example, COBIT. We are going to describe each of them:

- COBIT: This is a framework internationally accepted as good practice for information control, IT and related risks.
- ITIL: It will be described later (see Sect. 2.1.1).
- ISO 20000: It describes an integrated set of processes that can deliver in an effective way IT services to organizations and their customers.

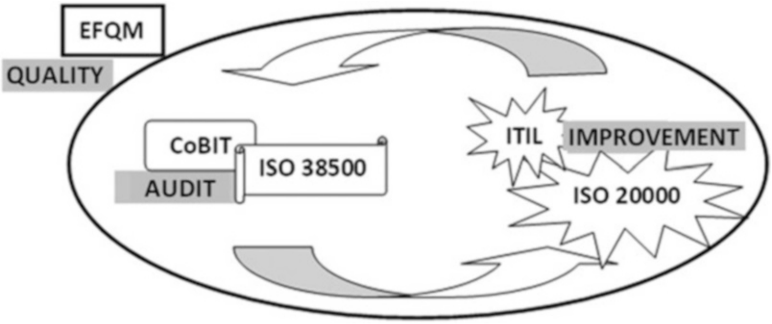


Fig. 23.1 COBIT/ISO 38500, ITIL/ISO 20000 and EFQM

- ISO/IEC 38500: It provides a service framework for the management of organizations using them to evaluate, manage and monitor the use of IT.
- EFQM: It will be described later (see Sect. 2.1.2).

The trend is the integration of the action frameworks which allow their use efficiently, as is the case of the four mentioned above (Fig. 23.1).

It is important that there is no single answer to the selection of frameworks to be used at any time. In principle we can say that there is no approach that covers everything, from the government of IT to the implementation of specific processes such as security, but, rather, a set of approaches that complement each other to cover the whole scenario.

There are other action frameworks that we could enumerate them (Fig. 23.2), although the consideration of the previous four models, they amply cover most of the claims concerning government and IT governance in an organization. The important thing in any case is to do a good selection of these and know with which of them one should start to work.

2.1.1 ITIL Overview

ITIL was created by the UK's Office of Government Commerce (OGC) to organize IT management in the public sector. ITIL is now managed by the Information Technology Service Management Forum (ITSMF). ITIL is in its third edition, called ITIL V3, released in 2007. One of the main goals of ITIL is to transform IT departments into service-oriented organizations (Fig. 23.3).

Service management *is a set of specialized organizational capabilities for providing value to customers in the form of services* (Office of Government Commerce 2007).

ITIL doesn't include mechanisms to evaluate the reference quality levels or measurement of quality improvement, so it could be interesting to use ITIL with one of the models of excellence in management, as proposed by the European Foundation for Quality Management EFQM. In the same way, the Six Sigma

	International Standard	National Standard	Organizations Standard
ICT Management	*ISO 20000*	BS 15000	*COBIT* *ITIL* MOF
Project Management			PMBOOK PRINCE2 APMs
Security Management	ISO 13335 ISO 13569 ISO 17799 ISO 15408	NIST – 800 series BS 7799-2 GAO's FISCAM German BSI	ACSI-33 *COBIT* ISF ENV12924 SEI's OCTAVE SEI's SW-CMM Baseline_Production_Manual
SW Development	ISO 12007 ISO 15504	Ticket	CMMI Bootstrap
Quality Management	ISO 9001	*EFQM* Baldridge_National_Quality_Plan	
ICT Government	*ISO 38500*	AS 8105 COSO	*COBIT*
Risks Management		AS / NZS 4360 COSO	
Continuity Management		PAS-56 AS / NZS 4360 HB 221-2004	
Audit	ISO 19011		*COBIT*

Fig. 23.2 Action frameworks

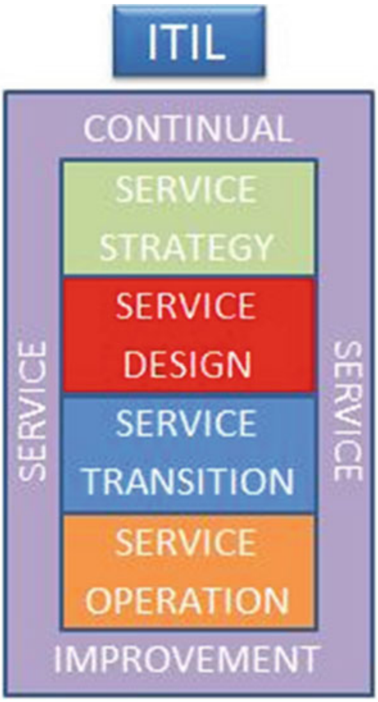


Fig. 23.3 ITIL's service management: service strategy, service design, service transition, service operation and continual service improvement

program and the revised version of ISO 9001:2001 bet to generalize this organizational orientation.

Service providers are more and more focusing on the quality of services; while they adopt an increased focus on business and the customer, they are approaching service delivery and cost optimization. To ensure a complete services management, operational and functional requirements can be expensive and even wrong in an organization.

The quality of a service is the ability it has to meet the needs and expectations of the client, taking into account the effort and cost required to achieve it.

Since the quality of a service is to meet the real needs of customers, it becomes necessary to manage the expectations of them. These should be translated into requirements to develop a system to meet those requirements in a predictable manner, to establish a methodology for action to achieve *zero defects* at work and to have a measurement procedure for monitoring customer satisfaction.

For the measurement of quality, there are different models. These quality models identify key elements of any organization and provide evaluation mechanisms for improvement. Some of these models are:

- EFQM: Described in Sect. 2.1.2.
- ISO 9000: Is a set of rules and they set up a model for the quality assurance in design, development, production, installation and after-sales service. This standard develops and implements a system of quality management in a company.
- ISO 20000: It describes an integrated set of processes that can effectively deliver IT services to organizations and their customers.
- CMM: A model for improving development processes that provides guidance for designing effective processes (time and cost) in different domains (development of products and services, acquisitions and maintenance), within the scope of an organization whose main premise is:

  > The quality of a product is determined largely by the quality of the process used to develop it and maintain it.

2.1.2 EFQM Model Approach

Once the different models for measuring quality have been described in this chapter, we have focused on EFQM because it is one of the most popular models today. EFQM is the model excellence assessment currently most widely used by European organizations.

An important advantage of the EFQM self-assessment is the quantitative character that it has; in other words, it can be compared against other organizations, and it can be compared to previous assessments of the same organization or between different business units within the same organization.

The EFQM Excellence Model is an instrument to do self-assessment and management. It is used to know in what position is one organization, to guide its management in accordance with the principles of quality management.

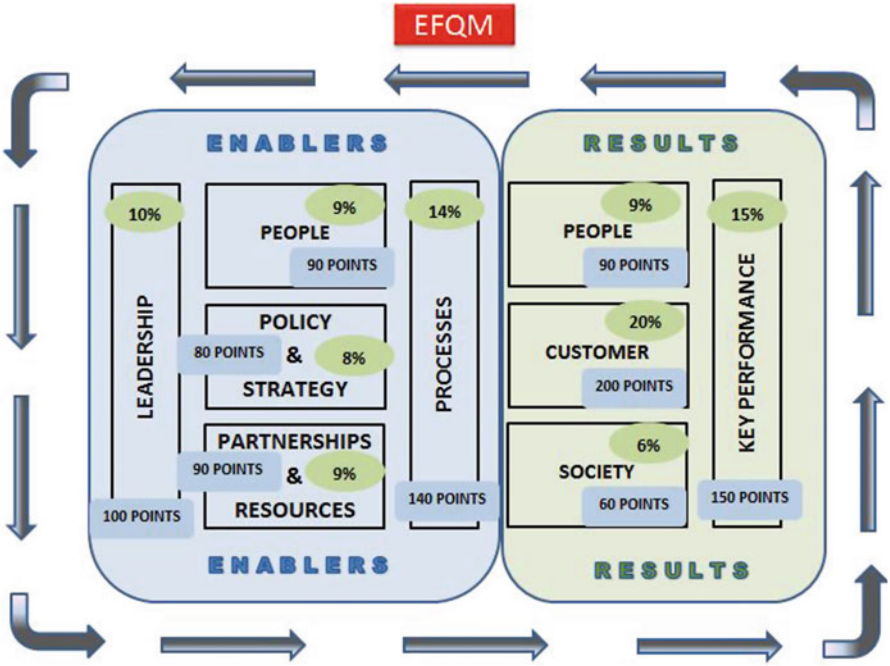


Fig. 23.4 The excellence model framework

The model recognizes that the excellence in everything related to results and performance of an organization can be achieved through different approaches, based on that:

> The excellent results regarding the performance of the organization, customers, people and society in which it operates are achieved through leadership that directs and drives the Policy and Strategy, which will be realized through people of the organization, partnerships and resources and processes.

The EFQM Excellence Model (see Fig. 23.4) consists of nine criteria which are divided in two groups: enablers criteria and the results criteria. The first criteria deals with what the organization does and they refer to causal factors whose effects take the form of that in the second criteria. The criteria referred to as *results* are about what the organization achieves. The *results* are the result of *enablers* and *enablers* are improved using feedback from *results* (Urbaniak 2004). Also this figure shows the weighting applied to the allocation of points to each of the criteria according to the model, since not all criteria have equal weights in the final score.

The criteria are interrelated not only within each group but within the whole model. Each criterion is composed by 32 different subcomponents, and these are deployed too, on different elements to consider.

In the previous figure, the arrows highlight the dynamic nature of the model, showing that innovation and learning enhance the work of enabler agents resulting in improved performance.

The nine *boxes* of the model represented above show the criteria for assessing an organization's progress towards excellence. The term *excellence* is used because the model focuses on what an organization does, or could do, to provide an excellent service or product to its customers, service users or stakeholders (Karkoszka and Roszak 2005; Lancucki 2001).

3 Implementing EFQ_TIL Model

Once ITIL and EFQM have been briefly described, in this section we will see the possible relationships that may exist between ITIL processes and criteria/sub-criteria that EFQM defines.

Both ITIL and EFQM models have as one of its basic principles and requirements study and analysis of processes. Both include and promote the process approach as an important element for obtaining efficient results because it allows and requires its measurement and improvement.

With the EFQM model line-up, the enablers of Leadership, People, Policy and Strategy and Partnerships and Resources and with ITIL, which bets by the best management, process to achieve the best quality results of IT services to customers, people and society in general.

The innovative aspect of ITIL is that it focuses on the processes of the EFQM Excellence Model in quality from the perspective of the life cycle of IT services (Strategy, Design, Transition, Operation and Continual Service Improvement).

In the first analysis, we can see that there are some ITIL processes that could be related to the criteria of EFQM; we list them below.

3.1 Criterion 2: Policy and Strategy

Sub-criterion *2.a. The policy and strategy are based on the needs and expectations of current and future stakeholders*; they include *what* the organization does:

- To collect and analyse information to help define the market and market segment in which the organization operates both now and in the future
- To understand and to anticipate the needs and expectations of customers, employees, partners, shareholders and society in general

The correspondence of this sub-criterion could be in service design, defined by the process service catalogue management, which provides a single source of information for all products provided by a company. It is a subset of services available from the catalogue of services (defined in the service strategy).

3.2 Criterion 4: Partnerships and Resources

In this sub-criterion *4.b economical and financial resources management*, they include *what* the organization does:

- Managing the economic and financial resources to support policy and strategy
- Developing and implementing strategies and economic and financial processes
- Evaluating investments in tangibles and intangibles
- Employing mechanisms and economic and financial parameters to ensure a structure for effective and efficient resources
- Managing the risks of economic and financial resources

This sub-criterion has its correspondence in ITIL in service strategy, within financial management process, which is responsible for evaluating and controlling the costs associated with IT services to provide a quality service to customers with an efficient use of IT resources.

In the sub-criterion *4.e. information and knowledge management*, you can include *what* the organization does:

- To collect, to organize and to manage the information and knowledge to support policy and strategy
- To enable internal and external users an appropriate access to information and relevant knowledge
- To ensure and improve the validity, integrity and security of information.
- To grow, to develop and to protect intellectual property that only the organization has, to maximize its value for the customer
- Try to acquire, increase and use the knowledge effectively
- Build in the organization a climate of innovation and creativity through the use of relevant information resources and knowledge

This sub-criterion has its correspondence in ITIL in service transition, defined by the process knowledge management, which is responsible for collecting, analysing, archiving and sharing knowledge and information within an organization. The primary purpose of this effort is to improve efficiency by reducing the need to rediscover knowledge.

3.3 Criterion 5: Process

Criterion 5 describes *how to design, manage and improve organizational processes to support policy and strategy and to satisfy fully, generating increasing value, its customers and other stakeholders*.

Since ITIL is composed by different processes, these processes must be well documented and optimized (processes and procedures manuals, service catalogue, etc.); therefore, this criterion of EFQM should get one of the highest scores.

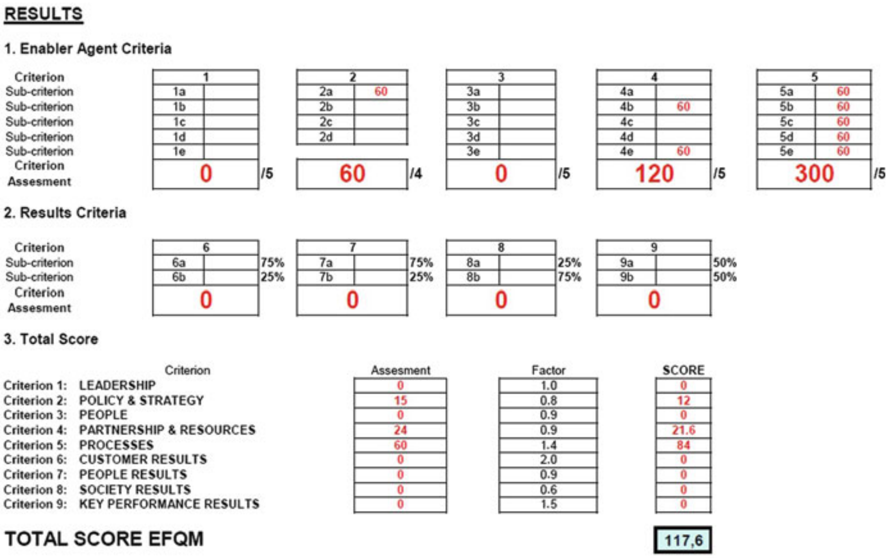
Fig. 23.5 EFQM scores

4 Results

The EFQM Model total score ranges from 0 to 1,000 points. In the following figure (Fig. 23.5), we can see, depending on the score, the various surveys carried out by the EFQM through its local partner in Spain which is the Excellence in Club Management (http://www.clubexcelencia.org). This figure tries to show the different levels of score.

The first external recognition (seal of quality, equivalent to a certificate if we spoke of rules) called *Commitment to Excellence* is awarded to entities applying between 20 and 30 % of the model, while the top prize *European Excellence 500+* is awarded to entities applying for more than 50 % of the model, putting on shows how demanding it is.

In this situation we must remember that entities with scores above 350 points are considered important and organizations with a significant development. In turn, organizations that are starting with the implementation of the model and have some



RESULTS

1. Enabler Agent Criteria

Criterion	1			2			3			4			5		
Sub-criterion	1a			2a	60		3a			4a			5a	60	
Sub-criterion	1b			2b			3b			4b	60		5b	60	
Sub-criterion	1c			2c			3c			4c			5c	60	
Sub-criterion	1d			2d			3d			4d			5d	60	
Sub-criterion	1e						3e			4e	60		5e	60	
Criterion Assesment	0		/5	60		/4	0		/5	120		/5	300		/5

2. Results Criteria

Criterion	6			7			8			9		
Sub-criterion	6a		75%	7a		75%	8a		25%	9a		50%
Sub-criterion	6b		25%	7b		25%	8b		75%	9b		50%
Criterion Assesment	0			0			0			0		

3. Total Score

	Criterion	Assesment	Factor	SCORE
Criterion 1:	LEADERSHIP	0	1.0	0
Criterion 2:	POLICY & STRATEGY	15	0.8	12
Criterion 3:	PEOPLE	0	0.9	0
Criterion 4:	PARTNERSHIP & RESOURCES	24	0.9	21.6
Criterion 5:	PROCESSES	60	1.4	84
Criterion 6:	CUSTOMER RESULTS	0	2.0	0
Criterion 7:	PEOPLE RESULTS	0	0.9	0
Criterion 8:	SOCIETY RESULTS	0	0.6	0
Criterion 9:	KEY PERFORMANCE RESULTS	0	1.5	0

TOTAL SCORE EFQM 117,6

Fig. 23.6 EFQM punctuation calculated in our model

degree of development of its management (some about planning, some HR management, performance of certain measurements, etc.) could be around 100 points; it means using about 10 % of what sets the model.

During our study, we worked out the questionnaire with nine criteria of the EFQM model, which were used to measure the quality in ITIL processes. The scale in this questionnaire is from 1 (the worst) to 10 (the best) defining the degree of fulfilment through the concrete criterion in the process.

For instance, for Criterion 5 (Processes), whose maximum score could be 500 (100 points for each sub-criterion), we found some questions of EFQM related to all Criteria 5x that were covered by some processes of ITIL, so finally we assigned 60 points to each criterion 5a, 5b, 5c, 5d and 5e. We proceeded in the same way for the rest of the criteria.

Once we finished applying EFQM questionnaire to main ITIL processes, all received points in the questionnaire were multiplied by the weights of each criterion given in the EFQM model (e.g. processes must be applied a 1.4 weight). Adding all these final scores of each criterion, we obtained a final score 117.6 (Fig. 23.6):

As can be seen in the previous figure (Fig. 23.6), with a very basic analysis, we could say we have a score of 117.6. So, hopefully, doing a more thorough analysis, we can achieve some level of those proposed by EFQM.

5 Conclusions

The main goal of the researchers has stated as deployment of a comprehensive integrate model to measure the quality of processes in the implementation of ITIL in an organization using the EFQM model, in order to, first, make a proper correlation of ITIL processes and criteria EFQM and, second, explore possible deployment scenarios that maximize the results of evaluations conducted by EFQM. We called this model in our study as EFQ_TIL.

After performing this first analysis, there is some relationship between ITIL and EFQM. Thus, we could say that if an organization implements ITIL, met in a way, some questions that EFQM model raises to obtain EFQM excellence.

As future work, it should specify in greater detail what criteria are covered in full and which are not covered by ITIL, and we should use other methodologies, standards such as ISO 9000 and ISO/IEC 38500.

References

Bardhan IR, Demirkan H, Kannan PK, Kauffman RJ, Sougstad R (2010) An interdisciplinary perspective on IT services management and service science. J Manage Inf Syst 26:13–64. doi:10.2753/MIS0742-1222260402

Betz CT (2006) Architecture and patterns for IT service management, resource planning and governance: making shoes for the Cobbler's children, 1st edn. Morgan Kaufmann, San Francisco, pp 424. ISBN 10: 0123705932

Duffy J (2002a) IT/business alignment: is it an option or is it mandatory? IDC document # 26831

Duffy J (2002b) IT governance and business value part 1: IT governance – an issue of critical importance. IDC Document #27291

Galup S, Quan JJ, Dattero R, Conger S (2007) Information technology service management: an emerging area for academic research and pedagogical development. In: Proceedings of the 2007 ACM SIGMIS CPR conference on computer personnel research: the global information technology workforce, ACM, USA, pp 46–52. doi:10.1145/1235000.1235010

Karkoszka T, Roszak M (2005) Quality and environmental aspects in the technological process management. In: Proceedings of the conference "Projecting & managing of the realisation of the production", Zielona Góra, pp 63–68

Lancucki J (2001) Basis of total quality management. AE, Poznán

Office of Government Commerce (2007) ITIL service strategy. TSO, London

Oud EJ (2005) The value to IT using international standards. Inf Syst Control J 3:35–39

Patel NV (2003) An emerging strategy for e-business IT governance. In: Van Grembergen W (ed) Strategies for information technology governance. Idea Group Publishing, Hershey

Peterson R (2003) Integration strategies and tactics for information technology governance. In: Van Grembergen W (ed) Strategies for information technology governance. Idea Group Publishing, Hershey, pp 37–80

Ribbers PMA, Peterson RR, Parker MM (2002) Designing information technology governance processes: diagnosing contemporary practises and competing theories. In: Proceedings of the 35th Hawaiï international conference on system sciences (HICCS), Maui (CD-ROM)

Urbaniak M (2004) Quality management-theory and practice. Difin, Warsaw

Van Der Zee JTM, De Jong B (1999) Alignment is not enough: integrating business and information technology management with the balanced business scorecard. J Manage Inf Syst 16 (2):137–158

Van Grembergen W, De Haes S, Guldentops E (2004) Structures, processes and relational mechanisms for information technology governance: theories and practices. In: Van Grembergen (ed) Strategies for information technology governance. Idea Group Publishing, Hershey

Weill P, Ross J (2004a) IT governance: how top performers manage it decision rights for superior results. Harvard Business School Press, Boston

Weill P, Ross J (2004b) IT governance on one page. CISR WP No. 349 and Sloan WP No. 4516-04

Weill P, Woodham R (2002) Don't just lead, govern: implementing effective IT governance. CISR WP No. 326 and Sloan WP No. 4237-02

Chapter 24
Accentuated Factors of Handheld Computing

Bo Andersson and Stefan Henningsson

1 Introduction

The recent years of rapid development of mobile technologies create opportunities for new user groups in the mobile workforce to take advantage of information systems (IS) based on mobile technologies – mobile IS. However, to apprehend and harness these opportunities, it is crucial to fully understand the user group and the mobile technology (Zheng and Yuan 2007).

There are evidences in the form of IS failure supporting the opinion that we still have lessons to learn on how to design and develop mobile IS for the mobile workforce. For example, a large corporate group in Northern Europe within the heavy industry and haulage sector implemented a service order system for their 280 service technicians in Sweden, where the end-user platform was a handheld computer. Savings due to shorter lead time from ordered service to sent invoice was one of the main reasons for developing and implementing the system. The desired benefits were achieved when the time from order to invoice was cut from 3 weeks to 3 days. The service technicians, however, deemed the system a failure owing to increased administration on their part from 20 to 90 min/day and lack of support for the service technicians' vital information needs. Post-implementation evaluation showed that the production loss caused by the technicians spending more than one hour less per day actually performing service could have been avoided if the system had been adapted to how the technician performed service order administration in the field (Andersson 2008).

B. Andersson (✉)
Department of Informatics, Lund School of Economics and Management, Lund University, Ole Römers väg 6, SE-22363 Lund, Sweden
e-mail: bo.andersson@ics.lu.se

S. Henningsson
Center for Applied ICT, Copenhagen Business School, Howitzvej 60, 2000 Fredriksberg, Denmark
e-mail: sh.caict@cbs.dk

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_24,

Similar stories in the business press and academic literature indicate that the example above is not an isolated anecdote but a typical example of how mobile IS projects do not harness the potential due to failure in understanding the use situation and the nature of the handheld computing device (Allen and Wilson 2005; Blechar et al. 2005; Luff and Heath 1998; Norman and Allen 2005; Steinert and Teufel 2005). This is the motivation behind this chapter and its focal point in the concept of mobility and what characterizes mobile IS from a designer's perspective.

2 Objectives

This chapter is based on the proposition that mobile IS has distinctive characteristics compared to the more traditional IS in the shape of stationary IS (Fällman 2003; Lyytinen and Yoo 2002). Our purpose is to develop and evaluate a framework for capturing aspects of handheld computing to be of importance during the analysis and design phases of mobile IS development. The intended use of this framework is in the design phase of the construction of a mobile information system for the mobile workforce. The intended user is a person working with the early design and requirement specifications. Consequently, pure technological aspects such as antennas, roaming and handover are not dealt with. The focus is neither differences in the IS content, that is, potential distinction of what kind of IS applications are being used in traditional and mobile IS, nor by which purpose. We are interested in entities that may be managed, or dealt with, in an IS design and development situation.

A specification of the core topic is warranted here. We are principally discussing handheld computing although in the majority of research, the term mobile computing have more or less been equated with handheld computing. In this chapter, the framework is only applicable on handheld devices where the Small Form Factor (Fällman 2003) is applicable. Another important aspect is that the context for this research is the mobile workforce; an effect of this is the treatment of mandatory versus voluntary use. The assumption is that in the workforce setting the use most often is mandatory and that this will influence use and design.

In the first step, this chapter deductively synthesizes a tentative analytical framework for capturing the accentuated factor of mobile IS, based on previous research on aspects on mobility and approaches to analyse mobile IS use and technology in IS design. 'Accentuated' in this chapter refers to factors that are new, factors that have gained importance or factors where the meaning have changed.

In the second step we use interviews with IS designers and developers to evaluate the framework's qualities. Thereafter we evaluate the framework based on criteria of completeness, distinctiveness and simplicity (see Sect. 4).

3 Related Literature

Much research has been done on mobile IS and mobility per se (Dahlberg 2003; Kristoffersen 1998; Perry et al. 2001), but the conceptualization of the term and what implication mobility has for IS design is still limited. Several frameworks have been developed in order to describe or explain aspects of mobility and IS use in a mobile context. Zheng and Yuan's (2007) framework with the entities' mobile workers, mobile context, mobile tasks and mobile technologies is developed to describe differences between stationary and mobile contexts. Kakihara and Sörensen's (2002) discusses mobility and includes temporal, spatial and contextual mobility into mobility as a phenomena. Focusing on design, Tarasewich suggests context to be divided into three categories: activities, environment and participants (Tarasewich 2003). All of these frameworks are important contributions to the field of mobile IS, but they are not specifically developed and focused on the design of mobile IS, and furthermore, they are not evaluated. They are more or less a loosely coupled set of factors assembled by the author. A comprehensive and evaluated analytical framework informing system designers is still missing.

One of the few attempts to illustrate factors related to mobile IS design and development, with an intended audience of designers, is the *additional dimensions of mobility* by B'Far (2005). As a consequence of this, in the construction of the basic framework, we departed from B'Far's (2005) framework of additional dimensions of mobility. Why we only depart from these dimensions, and not solely use them, is because these seven dimensions are not exhaustive; this will be discussed in the following section. B'Far identifies seven different dimensions as a result of mobility: *active transactions*, *limited device capabilities*, *Wireless Connectivity*, *location awareness*, *limited power supply*, *large variety of platforms* and *multimodal and variant user interfaces* (Fig. 24.1).

3.1 Some Issues with B'Far's Framework

However, there are some issues with the consistency in B'Far's (2005) framework; some of the different dimensions are of different magnitude or quantity. *Wireless Connectivity, Multimodal and Variant UIs* and *Location Awareness* are healthy, but *Limited Device Capabilities*, *Limited Power Supply* and *Large Variety of Platforms* are values on some scale and *Active Behaviour* is a desired interaction pattern. A more detailed description and motivation for altering dimensions are presented below. Another issue is the edges (lines) between the dimensions; B'Far (2005) does not explain their contribution to the dimensions. Is it that every dimension affects all other dimensions? If so, do they affect in the same magnitude in every case?

Another crucial aspect is our rejection of the concept 'dimension'; using *dimension* implies equivalent units of measure that are not easily constructed. We argue

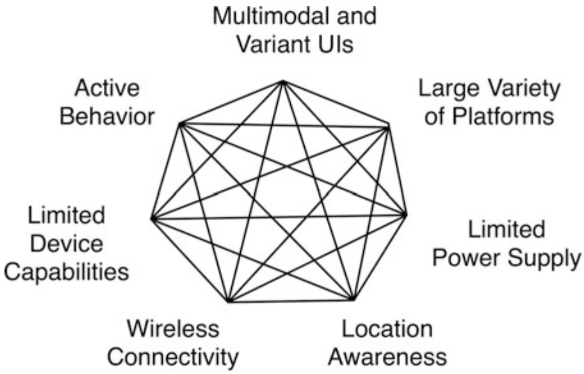


Fig. 24.1 The additional dimensions of mobile computing according to B'Far (Fallman et al. 2004)

that 'factor' is more descriptive, that is, 'a factor is one of the things that affects an event, decision or situation' (Sinclair 1995, p. 595). In this context, the factors in the framework can affect the outcome of a built system depending on how it is managed in the design and development of the system. And *factor* will be used henceforth.

Active Behaviour: Active Behaviour illustrates an interaction pattern a mobile application is supposed to have accordingly to B'Far. (2005). The mobile user is anticipated to benefit on short interaction sequences, short time periods of use and reluctance against long boot sequences, and Active Behaviour is patterned to manage this. The main reason is that the mobile workforce is often occupied with other tasks than working with their computer, a view supported by Marcus and Chen (2002), Pascoe et al. (2000), and Kristoffersen and Ljungberg (1999). However, Active Behaviour is one interaction pattern among others, and the factor is relabelled as *Interaction Patterns*. That is, the mobile user may, or may not, have tasks that will benefit on the interaction pattern of Active Behaviour.

Multimodal and Variant UIs: This factor is complex and regards the Small Form Factor with small screen and limited keyboard but also the increased interaction possibilities as using voice, sound and motion as input and output devices and the variation of different settings as differences between different keyboards or different screen resolution. Keyboard may be missing or is offered with limited set of keys compared to an ordinary keyboard (B'Far 2005; Fallman et al. 2004). This factor is relabelled to *Small Form Factor: Interface;* the main reason to relabeling it is that Small Form Factor is a well-known concept in mobile and handheld computing.

Large Variety of Platforms: The mobile industry is characterized with a large and heterogeneous set of actors and stakeholder. This creates a complex ecosystem with competing technologies and standards that in turn affect designers trying to design systems functional on different platforms. In respect of this set of actors and in the case of mobile applications, the platform variation is large, meaning large variations of operating systems among the handhelds and large variation of hardware configurations (B'Far 2005; Andersson and Hedman 2007). These aspects remains in this factor, however, renamed as *Platform Variation* removing the adjective *large* because it implies a value. There may be cases where the variation is small.

Limited Power Supply and *Limited Device Capabilities*: Both of these factors relate to hardware capabilities. Limited Power Supply is a factor that is highly relevant for mobile devices, since in practice they require to be battery powered and independent of the fixed-power networks. Limited device capabilities regard chiefly the effects of miniaturization on hardware. Effects are reduced processing capability, limitations in storage, etc. (B'Far 2005). However, these two factors are strongly related to each other that they are placed in the same factor and the value (limited) is deleted, renaming it to *Small Form Factor: Hardware*.

Location Awareness: Handheld devices are mobile and therefore able to appear in different places, and they can by different means use information of its physical location. This location awareness can be achieved by GPS, triangulation, accessing nodes or other techniques (B'Far 2005). There has been a considerable amount of work on location-based services mainly of conceptual type or for marketing (Tilson et al. 2004), but lesser work has been done on supporting the mobile workforce with applications using location awareness. This factor remains unchanged in *location awareness*.

Wireless Connectivity: The factor Wireless Connectivity illustrates the unpredictability of quality of service both in transmission rate and connectivity. With wireless networks, disconnection is a factor to manage. Temporary disturbances as sun flares, road tunnels, interference and skip zone affect the transmission (B'Far 2005; Dunlop and Brewster 2002). These factors, possible variation in transmission rate and possible variation in connectivity, remain in the unaltered factor *Wireless Connectivity*.

3.2 Extending B'Far

When performing a literature review, some other factors surface. By adding these factors, we are gaining a more comprehensive framework on accentuated factors.

Field-Use Condition: For the mobile workforce most work is obviously done at the field, rendering a use situation often labelled field use conditions. Field-use conditions could incorporate social settings, supporting technologies, supportive colleagues, etc. However, we argue that field-use condition regards the physical surroundings as quiet or noisy environments, sunlight, darkness, heat or low temperature all influencing the field use in their own ways. The lack of a predefined workplace is also a part of this factor; mobile workers need to adapt to different and diversified workplaces (Perry et al. 2001; Brown and Kenton O'Hara 2003; Marcus and Gasperini 2006). This factor is labelled *field-use conditions*.

Anywhere: Anywhere is almost a trademark of handheld computing and depictures freedom of place. However, there is some ambiguity in the interpretation of anywhere; does anywhere illustrate the mobility of the user? or the mobility of an application? or the mobility of a document (Perry et al. 2001; Makimoto and Manners 1997; Siau and Shen 2003)?

In this chapter, anywhere illustrates the mobility of the user; however, in a work situation, the interpretation of freedom of place can be questioned. We argue that for the mobile workforce and mandatory use, the user most likely is not allowed to choose the place; on the contrary, the place may be specific. A 'just on place' requirement is more applicable, for example, it may be important that a doctor is at a specific place to do something. For clarity of the possible restrictions on anywhere, the alternative label *Place Critical* is put forth and will be used in the forthcoming framework.

Anytime: Anytime is closely related to anywhere and usually describes that the user can access certain information, a service or an application when the user wants a freedom in time (Perry et al. 2001; B'Far 2005; Makimoto and Manners 1997). The same ambiguity as for anywhere surfaces; to contrast the conception of freedom in time, when users actually do need information, it is often relatively a time-critical information such as the repair status on a machine or a some purchasing status just before a client meeting. For the same reasons as for anywhere, 'just in time' is a more accurate term to illustrate the mobile workforce and mandatory use relation to freedom in time. For clarity of the possible restrictions on anytime, the alternative label *Time Critical* is put forth and will be used in the forthcoming framework.

Security Issues: In wireless communication, security issues are present due to the risk of interception. These may be the different types of threats as masking, listening, browsing and distortion (Elliott and Phillips 2004; Nikita et al. 2001). Another security issue is the Small Form Factor and its omnipresence. The handheld device's small size opens up to the factor to be carried along in a greater extent than, for example, a laptop computer. This frequent exposure increases the risk of it being stolen or lost (greater exposure in foreign environments) than, for example, a desktop computer (Elliott and Phillips 2004; Ravi et al. 2002). These two security issues, wireless transmission threats and the increased risk of device being lost, are merged in the factor *security issues*.

Supporting Technologies: Compared to the office worker, the mobile workforce accessibility to supporting technologies is often limited. Important documents may not be easily accessed and displayed through a handheld device. File management, servers, fax machines, written manuals, written ledgers or other support systems may not be available in the same extent as in an office environment (Zheng and Yuan 2007; Brown and Kenton O'Hara 2003). This factor is labelled *Supporting Technologies*.

Support Situation: The fact that a considerable part of the mobile workforce is working by themselves on the field renders a lack of interaction of colleagues within an informal fashion. The coffee room interaction (Orr 1996) is missing and furthermore, offering IT/IS support can be more problematic due to the sheer distance (Andersson 2011). This factor is labelled *Support Situation*.

Information System Dependencies: IS dependencies regards the mobile workforce high reliance on their information system. If an implemented application is the only application the user may access, and this application is crucial for the user to conduct the work, the reliance on this application is high. If the application malfunction, or that the implemented workflow, does not match the actual/real workflow, these problems will have extensive negative impact on perceived usefulness and productivity (Andersson 2008; Andersson and Carlsson 2009). The options to ‘bypass’ problems with post-it notes or other IS application are often lesser. This factor is labelled *Information System Dependencies*.

3.3 A Tentative Framework

Departing from B’Far’s (2005) seven dimensions, and by a literature review extending it, a framework of 13 aspects was built. Some of the labels from B’Far’s framework were altered to reduce logical inconsistencies as being of different magnitude or quantity.

The tentative frameworks based on previous research on factors concerning handheld computing are *Field-Use Conditions*, *Information System Dependency*, *Interaction Patterns*, *Location Awareness*, *Place Critical*, *Platform Variation*, *Security Issues*, *Small Form Factor: Hardware*, *Small Form Factor: Interface*, *Support Situation*, *Supporting Technologies*, *Time Critical* and *Varying Connectivity*.

We deliberately choose not to illustrate the factors in a graphical notion as B’Far just because we do not want to confuse the reader as we were imposing relations between factors.

4 Method

The framework put forth in this chapter is an example of theory for analysing and describing (Gregor 2006). The framework in form of a collection of individual factors can be considered as a theory for describing the components of handheld computing, that is, ‘The theory does not extend beyond analysis and description. No causal relationships among phenomena are specified and no predictions are made’ (Gregor 2006, p. 620). The evaluation criteria for theory for analysing suggested by Gregor are applied. We conclude that the usefulness of this type of theory may be refined to be evaluated by its completeness, distinctiveness and simplicity. Completeness means that important categories or elements should not be omitted from the classification system, that is, the framework should be able to capture all important resources. Distinctiveness means that boundaries between categories and characteristics that define each category are clear. The empirical phenomena encountered should be possible to categorize according to these criteria without too much difficulty. Simplicity refers to that which by making a model or framework too elaborated or comprehensive, it makes it hard to work with and in the end makes it useless for its purpose (Gregor 2006).

Table 24.1 Informant profiles

Job title	Employer	Year of experience of mobile development
Senior developer	Cybercom	8
Software engineer	Mashmobile	6
CEO	Qubulus	11
Development consultant	Stratal	7
Program owner	Cybercom	11
Senior developer	Mobimation	14
Software engineer	Yahm	4
Software engineer	QlikTech	5
Software developer	Databolaget	6
Program manager	Logica	5
CTO	WIP	12
Program manager	Sigma	3
Program manager	Sogeti	12

The empirical data required to assess the frameworks usability was gathered by semi-structured face-to-face interviews. Thirteen informants with experience of design and development of handheld applications were interviewed (see Table 24.1).

The interviews had an average duration of 1 h and 30 min. All interviews revolved around what the informant considered being significant in the design of mobile information systems, the differences between stationary/desktop design and about the importance of the factors derived from literature and the applicability of a framework as such. All interviews were recorded and transcribed. After each interview, transcriptions were coded into groups related to factors and analysed chronologically in order to identify eventual saturation. The QDA software HyperRESEARCH was used during analysis. Saturation here refers to that no additional, unknown comments or suggestions regarding the three evaluation criteria came up during the interview, and saturation was reached after five interviews. Due to saturation, after nine interviews the interview guide was modified, aiming to also find dependencies between factors; however, the interviews still revolved the factors and were possible to analyse in line with the preceding nine interviews.

5 Framework Evaluation

The empirical evaluation of the framework was in large extent in favour of the tentative framework; however, regarding completeness, suggestions on adding one factor were put forth (see following section). Concerning distinctiveness, suggestion on the separation of aspects in one factor was also put forth (see following section). Concerning simplicity, all informants allege they understood the factors (see Table 24.2).

Table 24.2 The informant's evaluation on factors

Informant	Completeness	Distinctiveness	Simplicity
A	Add high-velocity environment	Agree	Agree
B	Agree	Agree	Agree
C	Agree	Agree	Agree
D	Agree	Agree	Agree
E	Agree	Division of SFF interface and multimodal interface	Agree
F	Agree	Agree	Agree
G	Agree	Agree	Agree
H	Add high-velocity environment	Division of SFF interface and multimodal interface	Agree
I	Agree	Division of SFF interface and multimodal interface	Agree
J	Agree	Agree	Agree
K	Agree	Agree	Agree
L	Agree	Agree	Agree
M	Agree	Agree	Agree

High-Velocity Environment was suggested as an additional factor. It illustrates the fast-changing environment with competing vendors, manufacturers and content providers, and it was argued that this factor is more fierce and withstanding than within stationary computing. High-Velocity Environment exists in stationary computing but the importance of this factor is greater in handheld computing. The life cycle of an application is shortened on a handheld device due to shorter expected lifetime of the device, more frequent changes in operating systems versions (with low degree of backward compatibility) and faster changes in carrier's platforms etc. However, the label *High-Velocity Environment* on this factor implies a value and to be consistent it is relabelled to *Industry Dynamics*.

Another factor created from the evaluation was Multimodal Interfaces. This aspect already exists in the framework in Small Form Factor: Interface; however, it was argued that a separation of concerns was necessary. Small Form Factor: Interface should concern the reduction on interface due to the Small Form Factor. Multimodal Interface, however, is not a consequence of Small Form Factor; it is a set of new I/O possibilities as motion control or LDR sensors and therefore should be distinguished by an own factor: *Multimodal Interface*.

To summon the evaluation of the accentuated factor framework, the informants supported the tentative framework although with the extension of two factors. Some comments on the framework applicability were that they regarded the framework as highly useful for senior designers with project management assignments and for persons responsible for procurement of IS/IT solutions and likes, both in design/development and in benchmarking of existing competing systems in a procurement process. As two informants put it:

This framework would be useful for anyone procuring a mobile IS, assisting that person to evaluate if all, for that specific case, important factors are recognised. (Senior developer, Cybercom)

Useful to use this framework and to specify the important factors in design, and also to evaluate an existing system and see if it matches the needed requirements. (Senior developer, Mobimation)

The final frameworks of accentuated factors for mobile information systems are Field-Use Conditions, Industry Dynamics, Information System Dependency, Interaction Patterns, Location Awareness, Multimodal Interfaces, Place Critical, Platform Variation, Security Issues, Small Form Factor: Hardware, Small Form Factor: Interface, Support Situation, Supporting Technologies, Time Critical and Varying Connectivity.

6 Conclusions

With the ambition of improving mobile IS design and development, we have in this chapter put forth a framework for describing accentuated factors in IS design for handheld computing from a designer's perspective.

Theoretically, we add to the existing knowledge base on mobile IS design in two regards. First, we apply a holistic approach on the accentuated factors by synthesizing previous literature into a comprehensive framework; second, we evaluate and expand it based on evaluations by experienced designers.

Practically, the proposed framework was evaluated based on the criteria that it should help mobile IS designers to better apprehend the properties of handheld computing. It can be used as an analytical tool in the design process to ensure that the accentuated factors are appropriately recognized, or as a tool for analysing competing solutions in the procurement process.

In this chapter we commence a theorization process aimed at analysing and describing accentuated factors of mobile IS design, with the final goal of providing mobile IS designers with tools to dealing with the particular problems with mobile IS design.

We foresee that the next steps in a cumulative process towards this ambition would be furthering of the empirical basis for drawing conclusions of the interdependencies between factors, that is, how do different factors affect each other and more in-depth analysis of the importance of the factors – finding critical factors. For example, explanatory theory to design artefacts fitting the work context of mobile IS designers will be needed.

References

Allen D, Wilson TD (2005) Action, interaction and the role of ambiguity in the introduction of mobile information systems in a UK police force. Mob Inf Syst-Bk 158:15–36

Andersson B (2008) About appropriation of mobile applications – the applicability of structural features and spirit. Paper presented at the proceedings of the European conference on information systems (ECIS), Galway

Andersson B (2011) Harnessing handheld computing – managing IS support to the digital ranger with defensive design. In: Proceedings of the sixth international conference on design science research in information systems and technology (DESRIST 2011), Milwaukee

Andersson B, Carlsson SA (2009) Designing for digital nomads: managing the high reliance on single application. Paper presented at the proceedings of the global mobility roundtable, Cairo

Andersson B, Hedman J (2007) Developing m-services: lessons learned from the developers' perspective. Commun Assoc Inf Syst 20(1):605–620

B'Far R (2005) Mobile computing principles: designing and developing mobile applications with UML and XML. Cambridge University Press, New York

Blechar J, Constantiou I, Damsgaard J (2005) The role of marketing in the adoption of new mobile services: is it worth the investment? ICMB 2005: international conference on mobile business, pp 370–376

Brown B, Kenton O'Hara B (2003) Place as a practical concern of mobile workers. Environ Plan A 35(9):1565–1588

Dahlberg P (2003) Local mobility. Diss Göteborg Univ, 2003, Department of Informatics, University of Göteborg

Dunlop M, Brewster S (2002) The challenge of mobile devices for human computer interaction. Personal Ubiquitous Comput 6(4):235–236

Elliott G, Phillips N (2004) Mobile commerce and wireless computing systems. Pearson Education, Harlow

Fällman D (2003) In romance with the materials of mobile interaction: a phenomenological approach to the design of mobile information technology. Umea University, Umeå

Fallman D, Lund A, Wiberg M (2004) Scrollpad: tangible scrolling with mobile devices. p 6

Gregor S (2006) The nature of theory in information systems. MIS Q 3:611–642

Kakihara M, Sørensen C (2002) Mobility: an extended perspective. Paper presented at the 35th annual Hawaii international conference on system sciences (HICSS'02)

Kristoffersen, Ljungberg (1998) Representing modalities in mobile computing – a model of IT use in mobile settings. In: Proceedings of interactive applications of mobile computing, Rostock

Kristoffersen S, Ljungberg F (1999) "Making place" to make IT work: empirical explorations of HCI for mobile CSCW. Paper presented at the proceedings of the international ACM SIGGROUP conference on supporting group work, Phoenix, Arizona, United States

Luff P, Heath C (1998) Mobility in collaboration. Paper presented at the proceedings of the 1998 ACM conference on computer supported cooperative work, Seattle, Washington, DC, USA

Lyytinen K, Yoo Y (2002) Research commentary: the next wave of nomadic computing. Inf Syst Res 13(4):377–388

Makimoto T, Manners D (1997) Digital nomad. Wiley, Chichester

Marcus A, Chen E (2002) Designing the PDA of the future. Interactions 9(1):35–44

Marcus A, Gasperini J (2006) Almost dead on arrival: a case study of non-user-centered design for a police emergency-response system. Interactions 13(5):12–18

Nikita B, Ian G, David W (2001) Intercepting mobile communications: the insecurity of 802.11

Norman A, Allen D (2005) Deployment and use of mobile information systems. Mob Inf Syst II 191:63–78

Orr JE (1996) Talking about machines: an ethnography of a modern job. ILR Press, Ithaca

Pascoe J, Ryan N, Morse D (2000) Using while moving: HCI issues in fieldwork environments. ACM Trans Comput-Hum Interact 7(3):417–437

Perry M, O'hara K, Sellen A, Brown B, Harper R (2001) Dealing with mobility: understanding access anytime, anywhere. ACM Trans Comput-Hum Interact 8(4):323–347. doi:http://doi.acm.org/10.1145/504704.504707

Ravi S, Raghunathan A, Potlapally N (2002) Securing wireless data: system architecture challenges. Paper presented at the proceedings of the 15th international symposium on system synthesis, Kyoto, Japan

Siau K, Shen Z (2003) Mobile communications and mobile services. Int J Mob Commun 1(1):3–14

Sinclair J (1995) Collins COBUILD English dictionary. Harper Collins, London

Steinert M, Teufel S (2005) The European mobile data service dilemma – an empirical analysis on the barriers of implementing mobile data services. Mob Inf Syst II 191:63–78

Tarasewich P (2003) Designing mobile commerce applications. Assoc Comput Mach Commun ACM 46(12):57–60

Tilson D, Lyytinen K, Baxter R (2004) A framework for selecting a location based service (LBS) strategy and service portfolio. pp 5–8

Zheng W, Yuan Y (2007) Identifying the differences between stationary office support and mobile work support: a conceptual framework. Int J Mob Commun 5(1):107–123

Chapter 25
What Users Do: SA&D with the ATSA Method

Robert B.K. Brown and Ian C. Piper

1 Introduction

Incomplete or inefficient elicitation, comprehension and transmission of client requirements are all sources of information system (IS) failure rates. Requirements may be missed, misunderstood or miscommunicated for the lack of a single, consistent, informing theory. Decades of consistently poor results often attributed to requirements or communication issues strongly indicate high-level misunderstanding and/or misrepresentation of user requirements. It seems developed products are, too often, not what stakeholders and users actually want or need.

The ‘elephant in the systems development room’ is broadly acknowledged, but politely obscured, unacceptably high system failure rate. The CHAOS report (Standish 1995) suggests that over 30 % of projects are not completed at all, that over 50 % of projects run into significant cost overruns and that as few as 17 % of projects are completed both on time and in budget. CHAOS 1995 reported that poor requirements and a lack of user involvement were the highest perceived factors. The OASIG survey (1995) produced the infamous ‘70 % failure’ result. Over the years many studies have reported comparable failure rates across the information systems domain.

Such infamous statistics (particularly from the 1990s) have been marginalized as outdated; however, to date no statistics have been published suggesting any significant improvement. Crear (2009), CIO of the Standish Group, reports that the failure rates in 2009 ‘are low point in the last five study periods. 2009s results represent the highest failure rate in over a decade’.

R.B.K. Brown (✉) • I.C. Piper
University of Wollongong, 2522 Wollongong, NSW, Australia
e-mail: bobbrown@uow.edu.au; ian@uow.edu.au

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_25,

1.1 Towards an Improvement Through Use-Centrism

Approaches sensitive to the user viewpoint are valid as stakeholders generate their own notations and terminologies, complicating elicitation (Sommerville et al. 1998). Significant risk of failure exists in marginalizing stakeholder's *softer* objectives, despite their inherent informality. If poor requirements are at least partially attributable to poor communications between phases, analysts, designers and users, then a case exists for a lightweight, readily learnable, methodologically flexible end-to-end approach, under a single theoretical framework (addressing user activity) that concentrates on the identification of requirements. Observing the shift of focus from technology to people under user-centric design, Constantine and Lockwood (1999) said 'It is not users who must be understood, but usage'. Räsänen and Nyce (2006) argued for anthropological analysis to avoid skewing the focus of analysis to individual users over their larger socio-structural processes. A rough taxonomy of evolving centrisms might be given as product centric, process centric, goal centric, then user centric, which itself may be conceived as containing dependency centrism and motive or activity centrism. This final conception, termed *use-centrism*, conveys 'per user, engaged in work'.

Perhaps the most widely studied and respected motive activity-centric approaches to the issue of requirement elicitation were based upon the Russian psychosocial principles of Activity Theory (AT). Since English lacks noun to denote 'generic actions' (the Russian noun *deyatelnost* has no adequate English equivalent (Verenikina and Gould 1998), we adopt the term 'doings', a plural of 'doing'.

It is well beyond the scope of this chapter to properly introduce AT, so the interested reader is directed to earlier publications (Brown et al. 2006). The briefest possible outline is presented below.

AT identifies an activity as the smallest meaningful task carried out by a human subject. Vygotsky (1978) held that all human activity is carried out by a subject, using physical or psychological tools to achieve some object which may result in a physical outcome. Leont'ev (1978) proposed that all collective activities are directed to a single object (or motive). Within that abstract motive are more specific goal-oriented actions. His three layer hierarchic structure of activity, action and operation represents different levels of intellectual 'engagement'. At the base level, near-autonomic operations react to prevailing conditions. Leont'ev's notion describes doings abstracting to higher levels as the subject devotes more cognitive attention upon them. Kuutti (1991) introduced a topmost abstraction: the activity network. Engström (1987) described the structure of each activity as a seven-node matrix. Traditionally, AT is concerned with the cognitive ramifications of the differences between intended object and resultant outcome.

A number of activities may reside near one another and interact, forming a network that describes a larger process (Kuutti 1991). The outcome of one activity may constitute (among other things) a tool in another (Vrazalic 2004). We are specifically interested in outcome-tool transactions. It is necessary to shift focus away from the psychosocial and cognitive aspects to the investigation of the facilitating tool(s) of an activity, as a subset of these could specify some new system.

Ultimately, the system may be specified by describing those outcome-tool transactions and transformations which may be between activities. The designer must identify and describe them. These descriptions specify the requirements for functions to facilitate these transactions and transformations. We hope to isolate those which could usefully pass through some facilitating computer system, and we will use the term ‘instrument’ to refer to data-artefacts that are passed in such transactions.

AT was not readily convertible into a workable systems analysis and design (SA&D) method. Martins and Daltrini (1999) unhappily reporting that AT had not yet delivered any prescriptive methods. Otwell (2005) bemoaned a lack of prescriptive method for applying AT; saying ‘Activity Theory seems to almost *defy* practical application’. Otwell specifically cited Collins’ activity-centered design (ACD) work of 2002 as offering no concrete example of AT in use for design. Brown (2010) surveyed 13 prior attempts to apply AT to some or all of the SA&D process and found that most were abandoned, converted into niche tools or, at best, delivered little more than a set of issues for systems designers to be mindful of in their work.

It must be acknowledged that although a system may not be what a client wants it could still be what they need. Such tensions could result in the ‘failure’ of an otherwise technically excellent product. There are grounds to suspect that SA&D methods to date have not bridged clients’ inadequate grasp of IS and the analyst/designers’ inadequate grasp of their client.

Taking the broadest brush abstraction of the software development life cycle (SDLC), we consider briefly the analysis of what will be needed, the design of a system to achieve that, and its implementation. During any analysis of ‘what to make’, there will be issues of translation between the analyst and the customers’ paradigms. These continue as requirements are passed between the disciplines of Analysis and Design. Assuming that a viable design can be arrived at, there will be issues of transcription between the design and implementation phases conveying the instructions ‘make it like this’.

We propose that a common framework, with a single set of coherent concepts and a consistent terminology, would offer significant benefits.

2 A Single Framework

We now describe the Activity Theoretic Software Architecture (ATSA) method. ATSA commences with early phase requirements elicitation, generates an activity-based business process modelling (BPM), then informs the specification of a system (including the structure of its interfaces), which would facilitate the clients’ ‘doings’ in a feasible manner.

The word 'architecture' was chosen to reflect the level of abstraction the method operates at. Drawing inspiration from HCI literature, the designed system 'product' was envisaged as existing in, and being complimentary to, some information/activity space. An ATSA design considers how the users interact with each other in such a space, somewhat analogous to classical architectural considerations of structural built environments.

Use-centrism under ATSA envisages a system as a tool to facilitate a community of users *doing work*. Each of these *users* is a person (position) engaged in one of their *activity-specific mindsets* (roles). Thus, from the perspective of ATSA, a position is a collection of users. A position's workstation therefore consists of clusters of activity-specific interfaces between which the position simply *moves*, according to their active role.

3 A Stepwise Description of ATSA

Below we present the briefest outline of ATSA, constrained by available space. More complete and illustrated details will be presented in longer papers currently under preparation and are available upon request of the authors.

To specify a system and its interfaces in a manner that is arranged by role, an activity-oriented BPM is of great use. Unfortunately, businesses typically describe themselves by 'duty', 'responsibility', 'rank' and 'position'. They speak in terms of highly abstract 'missions' and quite specific 'goals' but rarely at the level of an AT 'motive' (object). Brown (2010) identified this as the chief difficulty for the previous attempts to deploy AT in SA&D methods. To address this difficulty, ATSA first elicits the data which businesses can deliver and, through an analytical process, then identifies activities.

3.1 Elicitation Phase

The analyst obtains a listing of positions in the stakeholder group to identify all possible users of the system to be designed. Businesses may be expected to have some organization chart, and positions may be read off from there. If no chart exists, the analyst may facilitate one. Positions will closely correlate to group members as they appear in their hierarchic organizational arrangement. A group's position classifications are considerably more likely to be somewhat arbitrary duty-based classifiers, more indicative of responsibilities than of actual 'doings'. It is well worth understanding to whom each position reports or answers, but to design a feasible system, it will be far more important to know what resources they require to perform their work (from whom they obtain them and under what conditions) and what resources become available for others, as a result of their work.

The analyst identifies a list of roles, being positions *engaged in task-specific work and mindsets* and often using *task-specific tools*. Roles may be initially suggested by position descriptions, as retained for personnel management purposes. Positions and roles have a many-to-many relationship. Typically any one position may fill several roles, but there may also be more than one individual (position) capable or authorized to fill a given role.

As the final *use-centric* system will be described to facilitate 'doings', ATSA considers its users to be roles not positions. ATSA designs for a *community* of users, being roles performing activities, and provides each with appropriate activity-specific tools. Positions, therefore, cease to be of immediate concern to the analyst.

The analyst may need to identify roles from scratch through interview and observation, an excellent opportunity to elicit doing-based information rather than refine or correct the typically somewhat arbitrary duty statements businesses use.

The analyst identifies 'candidate instruments' (CIs) from existing files, folders, forms, records, registers, lists and databases. These are data-like *things* (tangible or not) which roles require to perform their work or which are made available for other roles as a result of their work. CIs may exist as formalized document types within the group. It may be difficult to differentiate a CI from a tool at this stage, and as some tools may yet be automated as system aspects/functions in the final design (thus 'instruments'), they should be recorded at this point as well.

Bureaucratic templates or *pro forma* items constitute separate (though related) CIs to each *instance* or record that conforms to such a template. Original template CIs typically persist even after the creation of separate *instance* CIs.

The analyst should strive to avoid the temptation to define parent-child hierarchies between candidate instruments. There should only be resources and products. Container-like CIs, such as folder or file instruments, can be subsumed into some *constraint statement*, imposing, for example, an obligation or procedure which labels, stores or restricts access. Other complex CIs may be re-expressible as doings.

The analyst may identify any number of intangible CIs such as requests, permissions, state flags or conditions. Others may exist in the tacit knowledge of skilled operators. The broad elicitation questions 'what do you need to perform your task?' and 'what do you produce or provide to others?' must be pursued thoroughly to elicit as many conditions, sequences, implicit or tacit data as possible. Even a verbal call between workers may prove to be a crucial timing flag. All such intangible resources (and products) must also be recorded as they may need to be represented in some form as data objects, doings or constraint conditions.

3.2 AT Analysis Phase

Having obtained the *obvious* details of the client business, the Analyst now seeks out task-specific goals. These are discrete, finite, achievable and measurable and more readily obtained than the more abstract activity-level motives (objects). Under the ATSA method, an action goal is a finite, measurable, discrete and achievable objective which encompasses *what* is to be done and *why*. Goal statements must be as clear and unambiguous as possible, as they are a kind of *genetic* information which will carry across to more abstract levels of analysis.

Called goal-driven actions (GDAs) under ATSA, these are each performed by one specific role, requiring some CIs, performing some work (under conditions and constraints and to some purpose) and then producing some CIs for other GDAs to use.

GDAs have CIs entering and/or leaving (at least one of each). These CIs represent the ingredients of the work done and the results of that work. Each GDA encapsulates one role's work towards satisfying a specific goal and records the CIs coming in and going out. Linked by transacting CIs passed between them, GDAs may be envisaged as nodes of a directed graph, with CI transactions as arcs. For convenience, this may be rendered as a GDA adjacency matrix (GDAAM).

Goals, however, only exist at the action layer of AT. Activities are driven by more abstract and somewhat strategic motives (objects) which ATSA envisaged as *sets of consistent goals*. By way of example, we might envisage 'submission of a research paper' as a motive, comprising numerous discrete goals such as 'conduct literature review', 'spell check', and 'format the paper'.

In order to identify activities, we break GDAs into component pieces and conflate them following AT precepts. The analyst decomposes each GDA into a number of single instrument nodes (SINs) equal to the number of CIs it receives and outputs. Each SIN will have a single instrument attached, either entering or leaving. Each SIN inherits its parent GDA's goal and constraints. GDAs effectively disappear at this stage, as SINs will be recombined to form activities. AT requires that each activity has just one subject or doer (role) and a single coherent motive (set of consistent goals).

Detailed piecewise deconstruction of GDAs into SINs should go some way towards revealing duplications, inefficiencies and ambiguities; awareness of which can strongly provoke consideration of possible changes. The analyst must refrain from instituting changes at this stage, but should record all such 'good ideas' for future consideration. It is first necessary to strive for a complete picture of the extant group process. Such apparent change opportunities may indicate weaknesses in the analysis and the need for further iterative consultation, elicitation and clarification. The current group process most probably does *function* in its current form, though of course there is always scope for improvement.

Following the precepts of classic AT, the analyst identifies activities by conflating of SINs according firstly to the commonality of their role and then the logical consistency of goals under a single motive (object).

It is tempting to consider at this stage that fractured GDAs will simply recombine as they were, and indeed that may happen frequently. The analyst, however, must be prepared to redefine roles as necessary, so long as it is the same *individual person* (position) performing as subject of the conflated activity. Initial identification of roles and GDAs may have been flawed or unfocussed and this sorting process facilitates clarification. It may be that the same *person* is doing something but that it may be under one *of their other mindsets* (roles).

The analyst must consider the goal statements of the SINs and order them into sets, each logically collected under a consistent motive statement. Until some future refinement of ATSA incorporates a rigorously formal multimodal logical consistency checking technique; this requires some careful consideration on the part of the analyst, and would benefit from stakeholder consultation.

As conflation by goal consistency may give rise to further redefinition or clarification of roles, the analyst must be prepared to conduct these two sorting processes iteratively, as required.

It is vital that the conflated motive for each identified activity inherits all the constraints and conditions of its component SINs in a consistent manner. These become the rules by which the activity is governed. Any conflated activity must contain full details of its doings, of all required instruments and of all transformations and products.

Conflated activities will each have numerous CIs, both in and out, and as these will *connect* activities to each other, the network can be illustrated as a directed graph. Each node will now be a fully formed activity which could be described according to its Engström matrix elements. Directed arcs represent instrument transactions. The adjacency matrix of this digraph is a special tool called the combined activity table (CAT).

Identification of the CAT (which may include reconfiguration and rearrangement of the network iteratively, under consultation with the stakeholders), together with details of each activity and all CIs, yields a consensually agreed, activity-oriented BPM. It facilitates arguing the case for designing a central enabling computer system. The analyst can imagine some *space* within the activity network where such a facilitating computerized system *could* be. *Some* of the tools (inclusive of CIs) used, consumed or produced by activities *could* be automated and/or subsumed by a facilitating system *around which* some activities may cluster. This system could replace or automate some entire activities or even create new ones – but the ultimate goal would be for it to *reduce* the cognitive burden of repetitive and mundane tasks, freeing the humans to engage in higher-level doings of more value to the stakeholder group as a whole.

Drawing upon insights gained during elicitation and analysis, together with any good ideas that arose, inefficiencies, duplications, multiple handlings and redundancies in the current process should be highlighted and possible solutions discussed.

3.3 Redesign Phase

Here the analyst now takes the part of a designer. Although traditionally the project might be handed off to a designer at this stage, ATSA allows for the entire SA&D to occur under one consistent paradigm and strongly advocates unifying analysis with design or, at the least, permits closer cooperation and mutual understanding between these two domains.

Under consultation with the stakeholder, the designer seeks to reconfigure and rearrange the group's process to enhance efficiency through the agency of a central facilitating computer system. Once the stakeholders are accepting a design, the method will describe it in sufficient detail to specify its requirements for detailed software code and interface construction and implementation.

The result of this third phase is a *redesigned BPM*, now featuring a specification for a facilitating computer system. The system design will be well suited to the users' doings which should positively impact upon usability, user acceptance and thus system success. With consultation between the analyst-designer and stakeholders, the arrangements of individuals in positions and roles may have changed (in effect, some business process reengineering), but this should only result in a closer match of the system and the groups internal dynamic to the group's overall top-level agenda.

The system will be specified in terms of its instruments (those CIs which ultimately become data-like things that reside within the system and are transacted across the system boundary with users) and their transactions (analogous to data in, data out and transforms) as well as the clustering of its interfaces according to users' doings.

If the stakeholder agrees that any inefficiencies, duplications, multiple handlings and redundancies exist in the current process, then a redesign of the process is indicated. This may involve redefining numerous roles and reassigning tasks (instrument transaction flows). It is strongly recommended that the designer resists any restructuring of the arrangement of positions (people) unless the stakeholder specifically requires changes at that level, though clarification of roles (within position) should only yield enhanced performance.

To assist in rationalizing the pre-system BPM, towards identifying a system, the designer should consider the activity network and identify *pipes*, which are those arcs (instrument transactions) between activities which have a common subject (owner) whereby the subject sends something to themselves, and then *joints* which are nodes (activities) through which an instrument passes unchanged.

Locations on the activity network (BPM) where pipes and joints coincide are strong candidates for collapse and or redesign. The designer must pay particular attention to the elicited constraints (purpose, sequence, obligation, etc.) to ensure any modification is not in breach of higher-level agendas. For example, it may appear efficient for a college to send exam answers directly to the exam marking procedure; however, this would be in breach of the *strategic* reason for the examination, which was to assess how well lessons have been retained by students.

In rationalizing the BPM, activities may be split or joined or new ones created. Whenever this occurs, component SINs must be considered. As in previous stages, SINs may be moved, duplicated or deleted as required; their *genetic* identities (including goals, constraints and the sender/recipients of their CIs) must be retained.

The space *between* the activities where some facilitating computer system could reside should start to become clear. Many (hopefully most) activities could be seen as being tangential to this *system space*, each able to exchange some of their candidate instruments through or with it.

As boundaries of the system become apparent, the designer reroutes instrument transactions through it. Consider that true instruments *reside* inside the (generically labelled) system data repository (SDR). Activity transactions that have been rerouted across the system boundary exchange data *with the system* rather than directly with other activities. Some activities may not conduct any transactions across the system boundary, and these peripheral activities cease to be part of the system. Details of peripheral activities should be retained however, to provide useful context and to facilitate any future revision or redesign.

CIs which are retained (however briefly) in the SDR are now deemed to be true instruments which closely correspond to data objects and variables, but ATSA deliberately avoids such labels to retain paradigmatic neutrality. ATSA outputs feasible, useful and acceptable high-level specifications of the system and its interface, such that the stakeholders comprehend the product, users comprehend their tasks and builders may apply whichever paradigms, techniques and languages are deemed best.

Activities should retain constraints *inherited* from their component SINs. SINs rules consist largely of temporal constraints such as 'must do during state S'. Rewrite as many rules as possible into deontic statements such as 'run iff instrument 'state S' = TRUE'. Such statements form a list of prerequisite instruments and their required states or values which oblige the system to maintain and monitor instrument states. Rules oblige instrument consumers to ensure the correct state. The degree to which rules are converted into deontic constraints is therefore an indication of system reliability.

Goal statements within each activity must be expressed as succinctly and accurately as possible. A reduced vocabulary form of structured language should be employed, perhaps a form of generic pseudocode. The object (motive) of each activity will then be a logically consistent, highly structured set of goal statements in functional requirements form.

There is value in ordering the specification by its instruments rather than by its activities. The coder, especially one who follows an object-oriented paradigm, is more likely to grasp the design on a per-data-object basis.

A database of some kind could prove a useful way to present this information, as it would allow the components of the specification to be arranged in various ways. This would retain activity information with each instrument. Each instrument record would offer a meaningful name and some classification of its functionality defined by the transforms it must undergo and the uses to which it is put as a resource.

It is crucial that each instrument's transaction is easily traceable through the activity network. It is important to *wire up* the transactions correctly, clustering instrument-handling interface widgets according to workstation.

Whilst the functions of the system are described through the instrument list and goals statements (indicating what data objects must move in and out of the system, under what transforms and constraints), the specified nature of the interface must also be generated.

Since each SIN will handle a single terminus of an instrument transaction and these will cross the system boundary at the interface, the ATSA method should provide an interface designer with a sufficiently detailed description of who is handling what, and to what end, that (according to the builder's adopted paradigm) a selection of screen widgets (switches) becomes apparent. Once again, the ATSA method seeks to supply high-level, paradigm-neutral specification to support a wide range of options for the builder.

Whilst much has been written, under various paradigms, about the selection of widgets and colour schemes, etc., an important element of interface construction that is not well understood in the literature (Dix et al. 2004) is the clustering of switches for best use. It is recognized that the assemblage of widgets can greatly impact usability, acceptance and thus success of a system.

Under the ATSA method, interface workstations are assembled according to the coincidence of activity subjects (roles) in individual users (positions). Each station constitutes one or more screen sets, one per activity. That is to say, the interfaces are marshalled on a *per activity* basis. There should be no need for the user to be clumsily swapping between modes or usage instances if the interfaces have been clustered according to what they need to do.

Each activity's screenset may contain a number of screens (windows, instances, tabs, etc.). Each of these should correlate to an action. Actions are no longer clearly defined at this stage however, but may be understood to be sets of SINs (subsets of activity) clustered according to a common goal. Some indicator of the likely location of actions may be obtained from the *genetic* lineage of each SIN. It must be noted that action clustering may have changed as the BPM was redesigned to accommodate the facilitating system. It is at the level of screen clustering that ATSA still retains the greatest flexible capacity for the builder's personal intuition. More rigorous techniques for determining action (screens) from the specification may emerge under future research, perhaps drawing upon logical analysis of deontic and temporal constraints.

4 Conclusion

Perhaps not unexpectedly, the construction of an innovative AT-based method during this research proved difficult in the extreme as the precepts of AT are famously difficult to tie down and deploy prescriptively. After numerous false starts, a set of techniques were arrived at which unexpectedly showed promise of applicability beyond HCI. Indications were that this innovative Activity Theoretic Systems Architecture (ATSA) method might serve for the analysis and design of information systems in general, inclusive of basic requirements for their user interfaces. Thus, having come full circle, AT showed indeed promise as an informing coherent basis across which systems analysis and design might be conducted.

The ATSA method was designed in strict adherence to the precepts of AT and with a view to addressing issues of coherence, communication and ease of use by the inexperienced. To ease acceptance and usefulness, ATSA was intended to be *agnostic* of pre-existing methods and *not unfriendly towards* as many implementation technologies as possible. In an ongoing research ATSA shows considerable promise for producing UML-style specifications deployable under object orientation and may yet offer hope for reconciliation between 'procedural' and 'data-centric' methods.

As tested thus far in case studies and undergraduate classes (paper in preparation), ATSA allows analysis of users' requirements and procedural specification, in some detail, in terms which a coder might readily instantiate. AT *can* serve as a coherent theoretical basis for systems analysis and design, inclusive of its interface requirements. Hopefully, some positive steps towards redressing some challenges which continue to face computer systems analysis and design, and that of their user interfaces, will result. The ATSA method is herewith presented for consideration, testing, development, teaching and, hopefully, application.

References

Brown RBK (2010) The ATSA method for specifying both system and user interface requirements. Ph.D. thesis, University of Wollongong, Australia

Brown RBK, Hyland P, Piper IC (2006) Specifying system requirements using the 5S method. ACIS 2006 proceedings, Paper 100. http://aisel.aisnet.org/acis2006/100

Constantine LL, Lockwood LAD (1999) Software for use a practical guide to the models and methods of usage-centred design. Addison-Wesley, Boston

Crear J (2009) CIO Standish Group. http://www1.standishgroup.com/newsroom/chaos_2009.php. Accessed 24 Mar 2010

Engström Y (1987) Learning by expanding: an activity-theoretical approach to developmental research. Orienta-Konsultit Oy, Helsinki

Kuutti K (1991) Activity theory and its applications to information systems research and development. In: Nissen HE, Klein HK, Hirsheims R (eds) Information systems research: contemporary approaches and emergent traditions. Elsevier Science, Amsterdam, pp 529–549

Leont'ev AN (1978) Activity, consciousness, and personality. Prentice Hall, Englewood Cliffs

Martins LEG, Daltrini BM (1999) An approach to software requirements elicitation using precepts from activity theory. Fourth IEEE international conference on automated software engineering, pp15–23

OASIG (1995) in http://www.it-cortex.com/Stat_Failure_Rate.htm#The%20OASIG%20Study%20. Rudin W (1976). Accessed 15 Mar 2007

Otwell A (2005) Activity theory and user-centered design. Blog: heyblog: a space for half-formed thoughts. (July 29, 2005). http://www.heyotwell.com/heyblog/archives/2005/07/activity_theory.html. Accessed 20 Aug 2008

Räsänen M, Nyce JM (2006) A new role for anthropology? – Rewriting 'Context' and 'Analysis' in HCI. Nordic conference on human-computer interaction: changing roles (NORDICHI'06), Oslo, Norway, Proceedings New York, NY, USA. ACM Press, pp 175–184

Sommerville I, Sawyer P, Viller S (1998) Viewpoints for requirements elicitation: a practical approach. In: Proceedings of the 3rd IEEE international conference on requirements engineering (ICRE'98), Colorado Springs, USA, pp 74–81

The Standish Group (1995) The CHAOS report (1994). http://www.standishgroup.com/sample_research/chaos_1994_1.php. Accessed 24 Mar 2010

Verenikina I, Gould E (1998) Cultural-historical psychology and activity theory. In: Hasan H, Gould E, Hyland P (eds) Information systems and activity theory: tools in context, 1st edn. University of Wollongong Press, Wollongong

Vrazalic L (2004) Towards holistic human-computer interaction evaluation research and practice: development and validation of the distributed usability evaluation method. Ph.D. thesis, University of Wollongong, Australia

Vygotsky LS (1978) Mind in society. Harvard University Press, Cambridge, MA

Chapter 26
Can Relational DBMS Scale Up to the Cloud?

George Feuerlicht and Jaroslav Pokorný

1 Introduction

For more than a quarter of a century, relational databases have been the dominant data management paradigm, but there are indications that this situation may be changing as a result of increasing volume and complexity of data and the emergence of new types of database applications. Some researchers have advocated specialized database architectures for applications such as data warehousing, text processing, e-science, business intelligence, and stream data processing, demonstrating significant performance advantages over RDBMS (relational database management system) (Stonebraker and Cetintemel 2005; Stonebraker et al. 2007). Data aggregation used extensively in data warehousing and e-science applications represents a particular challenge as query performance degrades rapidly with increasing data volumes. Arguably, the most significant recent challenge to relational database technology arises from the need to provide scalable solutions for cloud computing applications. Using increasingly more powerful servers to provide vertical scalability as a common practice for enterprise databases does not provide a practical solution for cloud environments because of the escalating costs associated with increasing server size. Equally, while SQL queries can be readily parallelized, the size and complexity of data make traditional cluster database systems that typically have a relatively small number of processors (e.g., 64) inapplicable. Most experts agree that providing elastic database services requires horizontal scaling over a very large number (typically thousands) of inexpensive

G. Feuerlicht (✉)
Faculty of Engineering and Information Technology, University of Technology, Sydney, Australia

University of Economics, Prague, Czech Republic
e-mail: george.feuerlicht@uts.edu.au

J. Pokorný
Charles University of Prague, Prague, Czech Republic
e-mail: pokorny@ksi.mff.cuni.cz

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_26,

commodity servers deployed in parallel to process database operations. *Shared-nothing* database architectures combined with horizontal data partitioning (sharding) have been used successfully for over two decades in parallel database systems to provide scalability, but these systems have not been designed to support Internet-scale databases that manage petabytes of data on daily basis and support hundreds of millions of users. Data loads and dynamic repartitioning can incur additional performance overheads in parallel databases that may not be acceptable in Internet applications. Massive scale of databases maintained by Facebook (Rothschild 2009), Google, eBay, and similar companies need specialized solutions that are not available commercially today. The so-called NoSQL (not only SQL) databases (e.g., Cassandra (Lakshman et al. 2010), Bigtable (Chang et al. 2006)) achieve horizontal scalability by relaxing transactional atomicity and consistency requirements and by denormalizing data records. Such trade-offs are necessary to achieve performance characteristics required by Internet-scale applications, but may not be acceptable in other applications with more stringent demands on data consistency. Database research and implementation challenges presented by new application requirements have been discussed extensively in recent literature (Agrawal et al. 2009; Feuerlicht 2010; Pokorný 2010).

In this chapter, we focus specifically on database challenges in the context of cloud computing. We discuss new approaches including NoSQL databases designed to support high scalability for cloud computing applications and describe their functionality trade-offs. We consider the role of relational technology and discuss research challenges that need to be addressed to enable effective deployment of RDBMS technology on cloud infrastructure. In the following section (Sect. 2), we review evolution of database technology over the last three decades and then describe examples of specialized database systems (Sect. 3). In Sect. 4 we focus on data management in cloud environments and discuss the relative advantages and drawbacks of NoSQL and relational databases. The final section (Sect. 5) contains our conclusions.

2 Evolution of Database Management

Dramatic improvements in computer hardware price/performance driven by Moore's law (Intel 2011) are constantly redefining the relationship between the capabilities of various components of database architecture (i.e., CPU, memory, and disk storage). Data management has evolved over the past four decades from file-based systems that used navigational access to data programmed in high-level languages to highly sophisticated relational database systems that automate and optimize database operations expressed in a non-procedural language. The first generation of DBMSs that emerged in the 1960s was based on the hierarchical and later on the CODASYL (Conference on Data Systems Languages) network model. These types of DBMSs suffered from poor data independence and required low-level programming to access database records. These early DBMSs were superseded by the next generation of database technology based on the relational model proposed by E. F. Codd in 1970 (Codd 1970, 1971). Codd advocated strong separation of logical and physical database and the use of a non-procedural language for data access. His theoretical work provided the foundation for modern database technology, and his initial proposal was followed by extensive research and development activity resulting in highly successful RDBMS technology. Relational databases solved two major interrelated problems of the earlier database approaches. The first achievement was to decouple the database from application programs by providing effective support for data independence. Second and equally important achievement was to free database application developers from the burden of programming navigational access to database records by introducing the concept of a non-procedural query language. IBM's Structured Query Language (SQL) developed at the IBM San Jose Research Laboratory became universally accepted as the database language for relational DBMS systems. As computer hardware became more powerful and query optimization techniques improved, RDBMS systems became the technology of choice for most applications, including those with stringent response time requirements. Relational DBMSs proved to be extraordinarily successful in taking advantage of new computing platforms and architectures. Most commercial RDBMSs incorporate a range of distributed database features including distributed queries, 2PC (two-phase commit) protocol, distributed transactions, and advanced replication facilities. Another important development involved taking advantage of multiprocessor server architectures to provide support for parallel execution of SQL queries. Parallel SQL was implemented for shared-memory, shared-disk, and shared-nothing parallel architectures with significant gains in performance and scalability when compared to single-server architecture.

Next challenge for relational databases was to incorporate support for complex objects and relationships and to provide seamless integration between database and application-level objects. Two competing approaches emerged: the *revolutionary* approach, seeking to develop a completely new, fully *object-oriented* database solution (Atkinson et al. 1989), and the *evolutionary* approach which took the path of adding object features to SQL (Stonebraker et al. 1990). Many experts regarded ODBMS (object database management system) as the next generation of database technology destined to supersede relational databases. However, ODBMS systems have not been able to match RDBMS technology in a number of important respects, including reliability, scalability, and level of standardization. Of equal importance was the failure of ODBMS technology to address the wider requirements of mainstream corporate applications. The evolutionary approach produced a new type of hybrid object-relational database management (ORDBMS) technology, preserving the benefits of relational databases and at the same time incorporating key object-oriented features. The resulting object-relational data model and SQL:1999 standard address the main limitations of relational databases by enabling data type extensibility via user-defined types (UDTs), implementing object identifiers and references, and supporting type inheritance and encapsulation (Eisenberg and Melton 1999). Another challenge to the dominance of relational databases arose with the emergence of XML and the need to manage semi-structured data at the beginning of this century. Several prototypes (e.g., Lore (Widom J, *Data management for XML research directions*)) and commercial products (e.g., Tamino (Schöning 2001)) were developed, and similar to the situation with ODBMS a decade earlier, some experts believed that *native* XML databases (NXD) will supersede relational databases. While NXD databases remain a niche solution, RDBMS technology based on the SQL:2003 standard (Eisenberg et al. 2004) incorporated XML functionality combining relational, object-oriented, and XML features into a single database environment.

3 Beyond Traditional RDBMS

In this section, we discuss the challenges of RDBMS technology in the context of new application requirements and review some of the efforts to address these requirements with specialized RDBMS systems.

3.1 Traditional RDBMS

Relational DBMS has evolved into highly reliable, secure, and scalable technology that can support thousands of users sharing terabytes of data with sub-second response times in transaction-intensive applications. Supporting ACID (atomicity, consistency, isolation, and durability) properties of database transactions constitutes a key requirement for enterprise DBMS. This involves implementing a concurrency mechanism to prevent the loss of data consistency by isolating the execution of individual transactions and ensuring recovery from failures by logging database records on stable storage. Commercial RDBMS systems implement sophisticated security controls to restrict access to data requiring that the level of authorization is checked for every database access. Database transactions are subject to integrity constraints that need to be checked for every SQL statement that makes changes to data (i.e., update, insert, and delete statements). To meet such stringent requirements, enterprise RDBMSs use highly sophisticated techniques to optimize database operations by cashing frequently used data in computer memory and minimizing physical I/O operations. Additional performance gains as well as improved fault tolerance are achieved by deploying cluster DBMS configurations with multiple servers executing transactions over a single shared database. While improving availability and performance, cluster DBMS involves additional complexity and cost. Leading commercial RDBMS implementations (e.g., Oracle[1],

[1] http://www.oracle.com/us/products/database/index.html

DB2[2]) extend support for ACID transactions into distributed database environments (i.e., situations where transactions span multiple databases) using the 2PC protocol (Ceri and Pelagatti 1984). Finally, most commercial RDBMS systems support data replication features that allow multiple copies of tables to be maintained in different network nodes. Both synchronous and asynchronous replications are supported with the RDBMS system automatically ensuring consistency of replicated data.

Commercially available RDBMS systems have been designed to include a range of features that are essential in enterprise computing environments (i.e., ACID transactions, security, constraint checking, etc.) and numerous extensions that support object-oriented features and XML. Another key design objective for RDBMS technology was to provide an application-neutral platform for data management, irrespective of the type of data, size of the database, and application requirements. The idea is that the same RDBMS platform can be *tuned* to perform well in transaction-intensive and query-intensive applications and handle objects of any complexity and size. Database tuning is a highly skilled task and there is typically a limit to the level of optimization that can be achieved. This approach leads to highly complex *feature-heavy* RDBMS solutions that are difficult to configure and manage. Some influential database researchers argue that "one-size-fits-all" traditional DBMS architecture is no longer applicable and that the "database market will fracture into a collection of independent database engines" biased toward specific types of applications (Stonebraker and Cetintemel 2005; Stonebraker et al. 2007). We discuss some examples of specialized RDBMS implementations in the next section.

3.2 Specialized Databases

To address the issues discussed above, some DBMS vendors have produced RDBMS products that are optimized with respect to specific types of database applications. OLAP (online analytical processing) applications were one of the first types of applications that required specialized database support to allow efficient storage, retrieval, and analysis of large volumes of multidimensional data (Colliat 1996). Although this type of data can be stored directly in relational tables, aggregating data in *fact tables* and the use of advanced indexing techniques can dramatically improve performance of OLAP applications. More recently, data warehouse appliances based on MPP (massively parallel processing) shared-nothing architecture have been developed to support BI (business intelligence) applications. Data warehouse appliances support parallel execution of queries over very large volumes of data (typically tens of petabytes) partitioned across hundreds of processing nodes (Melton 2002), but omit many of the features

[2] http://www-01.ibm.com/software/data/db2/

normally associated with RDBMS products making these systems unsuitable for transaction-intensive applications. Another type of specialized DBMS is a column-oriented DBMS (column store) that stores data in column order (i.e., data for each column is stored contiguously and in compressed form), rather than in row order as is the case in a traditional RDBMS. Column-oriented DBMSs support SQL operations including joins, and their main benefit is improved performance of queries that involve *expensive* operations (e.g., group by aggregation) over a large number of rows but using only a small subset of table columns. Examples of column stores include Vertica[3] based on a research prototype C-store DBMS (Stonebraker et al. 2005). Column stores have also been used to store XML (Částková and Pokorný 2010) and RDF data (e.g., SW-store described in Abadi et al. 2009).

4 Cloud Databases

The latest DBMS challenge is to provide data management for cloud computing applications in a scalable and cost-effective manner. Unlike a typical enterprise computing environment that involves a relatively small number of high-performance database servers, cloud infrastructure consists of potentially hundreds of thousands of commodity servers and storage devices. Data-intensive applications need to deal with data distribution and data consistency issues making the task of parallelizing database applications much more challenging (see discussion in Sect. 4.1). The need to support data-intensive cloud applications has generated extensive interest resulting in numerous commercial and open source implementations known under the *umbrella* term NoSQL databases/data stores. Implementations of NoSQL data stores focus on providing elasticity (i.e., up and down scalability), automatic provisioning, and fault tolerance for data-intensive applications and typically rely on data partitioning and replication implemented on top of a distributed programming platform based on the MapReduce (MR) paradigm (Armbrust et al. 2009; Das et al. 2010; Dean and Ghemawat 2008).

4.1 NoSQL Databases

The term NoSQL is poorly defined and includes a broad range of non-relational data stores that do not use the SQL language to access data and do not support relational operations (i.e., joins, etc.). As a result, implementation of queries requires low-level programming, and the systems lack many of the properties normally associated with a DBMS, including read consistency, multi-record

[3] http://www.vertica.com/

updates, database schema, declarative constraints, and data independence. Examples of NoSQL data stores include Google BigTable[4] and its open source version Hypertable[5] (Chang et al. 2006) that store data in composite (denormalized) records identified by the combination of row key, column key, and a timestamp. The underlying MR platform supports parallel execution of program modules over petabytes of data using thousands of machines, managing task scheduling, inter-machine communications, and recovery from machine failures. Other examples of NoSQL databases include Amazon S3[6] (single storage service) that supports management of large multimedia objects (up to five terabytes) providing extensive scalability (Brantner et al. 2008). Numerous other NoSQL databases in various stages of maturity and implementing variants of BigTable and S3 models have been developed: Hbase (Apache), PNUTS (Yahoo) (Cooper et al. 2008), SimpleDB[7], Dynamo (Amazon), Cassandra[8] (Facebook), Redis[9], MongoDB[10], Voldemort[11] (LinkedIn), and CouchDB[12].

4.2 Relaxing ACID Properties of Transactions

As noted in Sect. 3.1, maintaining high transaction throughput and at the same time ensuring full data consistency can be costly and typically involves high-end proprietary server technology and expensive enterprise-level RDBMS systems. While in most transaction-intensive enterprise applications (e.g., banking and financial applications) ACID transactions are mandatory, there are many other types of applications that can tolerate lower levels of isolation and in some cases even temporary loss of consistency. The idea of *trading* consistency for availability has been investigated in the context of distributed databases and replication management more than three decades ago (Ceri and Pelagatti 1984). More recently, interest in balancing out ACID properties and availability in large-scale distributed systems has been revived as a result of the emergence Internet-scale data-intensive applications. Internet-scale data stores (e.g., Google BigTable, Facebook Cassandra) rely on horizontal scalability over thousands of commodity servers and use a weak consistency replication model to improve data availability. Horizontal scalability is achieved by using a distributed shared-nothing architecture where each

[4] http://www.google.com/base/
[5] http://hypertable.org
[6] http://aws.amazon.com/s3/
[7] http://aws.amazon.com/simpledb/
[8] http://cassandra.apache.org
[9] http://code.google.com/p/redis
[10] http://www.mongodb.org
[11] http://project-voldemort.com
[12] http://couchdb.apache.org

processing node has all the required resources (i.e., processors, storage, and memory) to execute the application. Horizontal scalability works well for the application layer (i.e., application servers that execute compute-intensive tasks) as new nodes can be added or removed dynamically (on demand). However, to support scalability at the data layer involves implementation of a replication mechanism to deal with various types of failures (Maia et al. 2010). As the size of the system grows (i.e., the number of nodes increases), so does the probability of failure, in particular when using inexpensive commodity components. Node failures and network partitioning cause data stored at the affected nodes to become unavailable, so that multiple copies of data need to be maintained at different nodes to improve availability. As the replicas are maintained asynchronously, consistency cannot be guaranteed until re-synchronization takes place, i.e., replicas are guaranteed to converge eventually, but during the latency period (i.e., time between synchronization events) may contain *stale* (out-of-date) data values. Availability can be improved by increasing the number of replicas, but this impacts on performance as the changes need to be propagated to all nodes where replicas reside. Brewer argued that the requirements for consistency, availability, and partition tolerance (network partition tolerance ensures that write and read operations are redirected to available replicas when segments of the network become disconnected) cannot be met simultaneously (Brewer 2000; Gilbert and Lynch 2002), and proposed a weak consistency model BASE (Basically Available Soft-state Eventual consistency) (Pritchett 2008) that guarantees replicas to converge eventually, resolving any conflicts that may arise as a result of failure. Others have argued (notably Stonebraker (2010)) that LAN (local area network) partition is only one of eight possible types of failures that can occur (other types of failures include DBMS errors, O/S errors) and that LAN partitions are very rare so that compromising consistency in order to improve availability may not be the best strategy.

4.3 RDBMS and Cloud

NoSQL data stores address the requirements of Internet-scale data-intensive applications, but, as already noted in Sect. 4.1, lack many of the features that are essential for enterprise data management. RDBMS systems, on the other hand, are expensive to scale up and costly to deploy and operate (Das et al. 2010). To address the cost of deployment (i.e., installation, configuration, and database tuning), some RDBMS vendors (e.g., Oracle) make pre-configured virtual instances available on cloud infrastructure in the form of AMIs (Amazon Machine Images) on AWS (Amazon Web Services) EC2 (Elastic Compute Cloud). However, such RDBMS instances have limited scalability (e.g., Oracle Database Standard Edition can only be licensed on Amazon EC2 instances with up to 16 virtual cores), forcing designers to make upfront choices about the size of the database and likely processing load.

Recently, a number of research efforts have been directed toward developing RDBMS systems that can scale on cloud infrastructure. For example, the ElasTraS research project at UCSB aims to provide horizontal scalability using a cluster of commodity machines while being fault-tolerant and self-managing (Das et al. 2010). ElasTraS is designed to support two types of cloud databases: (1) large databases partitioned across a set of nodes and (2) a large number of small, independent databases used in multi-tenant SaaS (Software as a Service) applications. ElasTraS is a prototype RDBMS that uses similar design to Google Bigtable (i.e., key-value store model) but does not aim to scale up to petabyte-size databases with thousands of servers (Das et al. (2010) reports on a configuration with a 30-node cluster over one terabyte of data). ElasTraS treats a table row as a fundamental *unit of access* and implements a schema-level partitioning. Schema-level partitioning (i.e., storing data fragments from related tables in one partition) allows transactions to operate over multiple tables within a single database partition. This approach differs from parallel database partitioning schemes that typically use table-level fragmentation. Another class of research prototypes involves efforts to take advantage of improved cost/performance of DRAM (dynamic random access memory) and SSD (solid state disk) flash memory. It has been argued that disk storage constitutes the main bottleneck when implementing large-scale data-intensive applications on the Web. The performance of disk storage systems has not improved as rapidly as its capacity; over the last 25 years, disk capacity has improved more than 10,000-fold, while the maximum transfer rate only 50-fold and latency only by a factor of two (Ousterhout et al. 2010). The scalability limitations of disk storage directly impact on RDBMS scalability and force Web companies to implement proprietary hybrid solutions that typically combine relational technology with heavy reliance on memory cashing. For example, in 2009 Facebook database architecture consisted of 4,000 MySQL servers combined with 2,000 memcached servers (Fitzpatrick 2004), keeping 25 % of online data in main memory. RDBMS systems use extensive cashing to keep frequently accessed data in memory; however, the demands of Internet-scale applications are forcing an increasing percentage of the total storage system to be maintained in memory. Such considerations motivate the development of prototypes based entirely on DRAM storage (e.g., RAMCloud prototype at Stanford University (Ousterhout et al. 2010)). The principal advantage of DRAM is very low latency (5–10 microseconds); however, there are also drawbacks as DRAM storage is volatile and requires protection against power outages. To ensure transaction durability, data objects must be replicated across several servers (typically 3), significantly increasing the cost and power consumption. SSD flash memory offers an alternative to DRAM with the advantage of lower power consumption and nonvolatile storage, but with significantly increased latency (5–10 times). According to (Ousterhout et al. 2010), RAMCloud can be 100–1,000 times more efficient than disk-based systems and 1–10 times more efficient than systems based on SSD storage.

5 Conclusions

Most of the current discussion about cloud databases centers on the requirements of Internet-scale data stores that address specific needs of a small number of large Web companies (e.g., Google, Facebook). It is clear that the massive scale of such data stores is currently out of reach of commercial RDBMS implementations and that NoSQL data stores constitute a more suitable and less expensive solution for this category of applications. Another special category of data-intensive cloud applications is SaaS applications (e.g., salesforce.com CRM applications[13]) that, in addition to basic cloud features (i.e., elasticity, fault tolerance, and automatic provisioning), typically need enterprise-level functionality, including ACID transactions, security, and other features associated with commercial RDBMS technology. These requirements are currently being addressed by SaaS providers using a hybrid combination of RDBMS and non-relational solutions (e.g., database.com[14]). Final category includes enterprise databases currently implemented on-premise using RDBMS technology. Implementing a general purpose database as a service (DaaS) on cloud infrastructure presents a number of important challenges. Apart from the technical challenges of developing an enterprise-level DBMS capable of taking full advantage of cloud infrastructure, the costs associated with *renting* a DBMS on a pay-per-use basis may be prohibitive over a long period of time. Finally, we note that the present RDBMS limitations in the context of cloud computing are not inherent limitations of the relational model or of the SQL language, but implementation challenges that are likely to be resolved over the next few years as the price/performance of servers, data storage, and network technologies improve in line with Moore's law and as a result of research efforts in this area. As noted in Sect. 2, relational databases have shown remarkable ability to adjust to new requirements and to take advantage of new computing platforms and architectures, disproving predictions of their demise.

Acknowledgement This research has been partially supported by grants GACR no. P202/10/0761 and GACR no. P403/11/0574.

References

Abadi DJ, Boncz PA, Harizopoulos S (2009) Column-oriented database systems. Proc VLDB Endow 2(2):1664–1665

Agrawal R et al (2009) The Claremont report on database research. Commun ACM 52(6):56–65

Armbrust M et al (2009) Above the clouds: a Berkeley view of cloud computing. EECS Department, University of California, Berkeley, Technical report. UCB/EECS-2009-28

Atkinson M et al (1989) The object-oriented database system manifesto, Citeseer

[13] www.salesforce.com

[14] http://www.database.com/

Brantner M et al (2008) Building a database on S3. SIGMOD '08. ACM, Vancouver
Brewer EA (2000) Towards robust distributed systems. ACM PODC conference.
Částková Z, Pokorný J (2010) Schema-less XML in columns. In: 22nd Australasian database conference (ADC 2011), Perth, Australia
Ceri S, Pelagatti G (1984) Distributed databases principles and systems. McGraw-Hill computer science series 1984. McGraw-Hill, New York. 0-07-010829-3
Chang F et al (2006) Bigtable: a distributed storage system for structured data. OSDI '06
Codd EF (1970) A relational model of data for large shared data banks. Commun ACM 13 (6):377–387
Codd EF (1971) Normalized data structure: a brief tutorial. ACM-SIGFIDET workshop on data description, access and control. ACM, San Diego
Colliat G (1996) OLAP, relational, and multidimensional database systems. ACM SIGMOD Rec 25(3):64–69
Cooper BF et al (2008) PNUTS: Yahoo!'s hosted data serving platform. Proc VLDB Endow 1 (2):1277–1288
Das S et al (2010) Elastras: an elastic, scalable, and self managing transactional database for the cloud. UCSB computer science technical report 2010-04. UCSB. http://www.cs.ucsb.edu/~sudipto/tech_reports/2010-04.pdf
Dean J, Ghemawat S (2008) MapReduce: simplified data processing on large clusters. Commun ACM 51(1):107–113
Eisenberg A, Melton J (1999) SQL: 1999, formerly known as SQL3. ACM SIGMOD Rec 28 (1):138
Eisenberg A et al (2004) SQL: 2003 has been published. ACM SIGMOD Rec 33(1):119–126
Feuerlicht G (2010) Database trends and directions: current challenges and opportunities. DATASO 2010. Matfyzpress, Štědronín, Czech Republic
Fitzpatrick B (2004) Distributed caching with memcached. Linux J 2004(124), p 5
Gilbert S, Lynch N (2002) Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. ACM SIGACT News 33(2):51–59
Intel (2011) Moore's Law [cited 4 April 2011]. Available from: http://www.intel.com/about/companyinfo/museum/exhibits/moore.htm
Lakshman A et al (2010) Cassadra [cited 04-04-2011]. Available from: http://www.slideshare.net/jhammerb/data-presentations-cassandra-sigmod/
Maia F et al (2010) Scalable transactions in the cloud: partitioning revisited. On the move to meaningful internet systems, OTM 2010, pp 785–797
Melton J (2002) Advanced SQL 1999: understanding object-relational, and other advanced features. Elsevier Science Inc, New York, p 592
Ousterhout J et al (2010) The case for RAMClouds: scalable high-performance storage entirely in DRAM. ACM SIGOPS Oper Syst Rev 43(4):92–105
Pokorný J (2010) Databases in the 3rd millennium: trends and research directions. J Syst Integr 1 (1–2)
Pritchett D (2008) Base: an acid alternative. Queue 6(3):48–55
Rothschild J (2009) High performance at massive scale – lessons learned at Facebook. CNS 2009 lecture series archives 2009 [cited 15 March 2010; Centre for Networked Systems Lecture]. Available from: http://cns.ucsd.edu/lecturearchive09.shtml
Schöning H (2001) Tamino-a DBMS designed for XML. IEEE Computer Society
Stonebraker M (2010) Errors in database systems. Eventual consistency, and the CAP Theorem. http://cacm.acm.org/blogs/blog-cacm/83396-errors-in-database-systems-eventual-consistency-and-the-cap-theorem/fulltext
Stonebraker M, Cetintemel U (2005) One size fits all: an idea whose time has come and gone. IEEE. 0769522858
Stonebraker M et al (1990) Third-generation database system manifesto. SIGMOD Rec 19 (3):31–44
Stonebraker M et al (2005) C-store: a column-oriented DBMS. VLDB Endowment. 1595931546

Stonebraker M et al (2007) One size fits all? Part 2: Benchmarking results. In: Third biennial conference on innovative data systems research (CIDR), California, USA
Widom J. Data management for XML: research directions. Bulletin of the Technical Committee on: p 44

Chapter 27
A Comparative Analysis of Agile Maturity Models

Mauri Leppänen

1 Introduction

Agile software development has become increasingly common in the past 10 years. Agility is believed to help a project reduce time to market, improve product quality, increase customer value, improve developers' motivation, etc. Organizations apply the approach in different ways. Some of them have adopted only few agile practices and blended them with practices of their plan-driven methods. Some other organizations have fully committed to the practices of a certain agile method (e.g., XP (Beck 1999), Scrum (SEI 2006)). A small minority of organizations have progressed beyond the team and project level and involved product management and software business as well. After gaining some experience, the organizations start questioning how to assess their current state and how to improve their processes and practices. It is not enough anymore just to be agile, but they need to know how to be agile in a productive and effective way, that is to say, know how agility can help them become successful in project work and software production in general.

For traditional software development, several software process assessment and improvement models, called maturity and capability models (e.g., CMM (Paulk et al. 1993), CMMI (Shalloway et al. 2009), and ISO/IEC 15504 (2008)), have been developed. From the viewpoint of the agile approach, these models are based on different assumptions of development process, organization, and environment. Agility is not visible in the structures and contents of these models, although some correspondences between them and the agile values and principles have been recognized (Glazer et al. 2008; Paulk 2001; Pikkarainen and Mäntyniemi 2006; Sutherland et al. 2007; Turner and Jain 2002). Another way to assess the

M. Leppänen (✉)
Department of Computer Science and Information Systems, University of Jyväskylä, P.O. Box 35(Agora) FI-40014 Jyväskylä, Finland
e-mail: mauri@cs.jyu.fi

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_27,

current state of agile development is to check how faithfully an organization follows the practices of a certain agile method (e.g., XP (Beck 1999), Scrum (SEI 2006)) or to apply some of the so-called agility tests (e.g., the Nokia test, http://www.cedur.se/nokia_test2.html). These kinds of assessments are, however, too simple and not very feasible.

In order to embrace agile features in a more comprehensive way, special agile maturity models (e.g., Ambler 2010; Nawrocki et al. 2001; Packlick 2007; Patel and Ramachandran 2009; Pettit 2006; Qumer and Henderson-Sellers 2008) have been suggested. They all are composed of hierarchical maturity levels, but are otherwise quite different in their domains, backgrounds, structures, and contents. Some of them (Nawrocki et al. 2001; Patel and Ramachandran 2009) have adopted features of the CMM/CMMI model, while the others are separate from them.

The purpose of this study is to make a comparative analysis of agile maturity models, answering to the following research question: How do agile maturity models compare to each other? The comparison is based on the following criteria: purpose, domain, conceptual and theoretical backgrounds, approaches and principles, structure, use, and validation. The results from this study can be used to make selections between the agile maturity models for the needs of a certain organization and to further develop existing models.

This chapter is organized into five sections. In Sect. 2 we discuss agile software development and how its success is seen. In Sect. 3 we consider maturity models of software engineering in general and in relation to agility. In Sect. 4 we present the results of the comparative analysis, and this chapter ends with the discussion and conclusion in Sect. 5.

2 Agile Software Development

Before considering the agile maturity models, we first have to know what makes software development agile and how agility can be assessed. Second, since the aim of using a maturity model is to make software development more successful, we have to know how success in agile development is defined and what factors are critical to such success. These are discussed next.

To characterize the notion of agility it is common to refer to 4 values and 12 principles in the Agile Manifesto (Agile 2002). Unfortunately, the manifesto does not give any definition for agility. The second way of outlining agility is to highlight essential practices of agile methods, such as short iterations, small builds, self-organizing teams, customer involvement, proactive attitude toward changes in requirements, and customer value. The agile methods differ, however, in arrays of their practices (Abrahamsson et al. 2002; Qumer and Henderson-Sellers 2006), and there are differing ways of how practices can be applied. The third way of searching for the essence of agility is to rely on conceptual studies. Conboy (2009), for instance, develops a definition and formative taxonomy of agility, based on a literature review of agility across a number of disciplines. He recognizes two main concepts underlying agility, namely, flexibility and leanness, and conducts an incremental concept development to achieve a definition of agility. The definition goes as follows: "the continual readiness of an ISD method to rapidly or inherently create change, proactively or reactively embrace change, and learn from change while contributing to perceived customer value (economy, quality, and simplicity), through its collective components and relationships with its environment" (ibid, p. 340).

There are many suggestions for how agility of a project or organization can be assessed: e.g., the Nokia test (http://www.cedur.se/nokia_test2.html), Shodan adherence survey (Williams et al. 2004), agile evaluation framework (Krebs and Kroll 2008), and comparative agility assessment (Cohn 2010). In many of them agile practices are used as a yardstick. The practices may belong to some specific agile method(s) or be "commercially labeled agile practices" (Conboy 2009). Neither of the means is sufficient because practices vary among the methods and they can be, or should be, applied in various ways in different situations.

Being agile is not an end in itself. It is just a means to more successful performance. But how is success measured and what are the critical success factors for agile software development? Success metrics of software engineering commonly contain process variables and product variables (Procaccino et al. 2006). The most generic variables are quality and scope for product and time and cost for process (Cohn and Ford 2003; Jiang et al. 2004). These variables can be specialized into a multitude of sub-variables. Furthermore, stakeholders, such as developers, management, customers, and users, have different views of process and product success. Agility has changed these traditional conceptions about success. For instance, scope is allowed to change even radically during the process, and the project may still be successful. In general, agile methods focus more on customer satisfaction and business value.

There is a large body of literature about critical success factors (CSF) in information systems development (Poon and Wagner 2001). Some of the suggestions address agile development (Chow and Cao 2008; Misra et al. 2009). Chow and Cao (2008), for instance, derived three CFSs: delivery strategy, agile software development techniques, and team capabilities. In addition, they found three factors that could be critical to certain success dimensions, namely, a good agile project management process, an agile-friendly team environment, and a strong customer involvement. As Becker et al. (2009) and de Bruin et al. (2005) present, the CSFs should be addressed in the maturity models. But how to make them visible in the models is the question.

3 Maturity Models

Software companies continually face pressures to gain competitive advantage, identify ways of cutting costs, improve quality, and reduce time to market (de Bruin et al. 2005). To assist them, a large array of maturity models has been developed. Here we briefly discuss what the maturity models are, how they are used, and how useful they are seen in the context of agile development.

Maturity means "the state of being complete, perfect or ready" (Stålhane and Hanssen 2008). A *maturity model* presents "an evolutionary progress in the demonstration of a specific ability or in the accomplishment of a target from an initial to a desired or normally occurring end stage" (Mettler and Rohner 2009). The maturity models can be classified into *descriptive*, *prescriptive*, and *comparative models* (de Bruin et al. 2005). Based on their domains, the models can be divided into *process*, *object*, and *people maturity models* (Mettler and Rohner 2009). Furthermore, we can find *organization-level*, *team-level*, and *personal maturity models*. The maturity models typically have "a number of levels, a descriptor for each level, a generic description of the characteristics of each level as a whole, a number of dimensions, a number of elements or activities for each dimension, and a description of each element or practice as it might be performed at each level of maturity" (Fraser et al. 2002, p. 246).

CMM (Paulk et al. 1993) and CMMI (Shalloway et al. 2009) are the most well-known maturity models in software development. They are organization-level models intended for descriptive, prescriptive, and comparative purposes. Other organization-level models are ISO 9001:2000 (http://www.iso9000council.org/) and ISO/IEC 15504 (2008). The Team Software Process (TSP) provides a defined operational process framework to help teams organize large-scale software projects. The Personal Software Process (PSP) is an individual-level model addressing the software development capability of individual engineers (Humphrey 2005).

The usefulness of the maturity models is constantly argued for and against. There is a wide array of empirical studies on positive impacts of process maturity on project performance (e.g., Galin and Avrahami 2006; Herbsleb and Goldenson 1996; Humphrey et al. 1991; Jiang et al. 2004). On the other hand, CMM/CMMI is highly criticized (Boehm and Turner 2003; Fayad and Laitinen 1997). In any case, many companies have invested on CMMI or ISO 9001:2000, and they would like to utilize it also in the context of agile development (Pikkarainen and Mäntyniemi 2006). But the question is the following: To which extent the features of the traditional maturity models apply to the agile software development?

The opinions about this issue are highly divided. On one hand, integrating the plan-driven and agile approaches is said to be "like oil and water" (Lycett et al. 2003; Turner and Jain 2002). The plan-driven approach assumes that software development is a repeatable and predictable process. Agile advocates argue that every project situation is different, which makes these assumptions invalid (Highsmith 2002; Schwaber and Sutherland 2010). CMM/CMMI and ISO/IEC 15504 are seen process-oriented models, whereas the agile approach is more business goal-oriented (Patel and Ramachandran 2009). Furthermore, the origin of SPI goals is traditionally on the organizational level, while in the agile project context the power for SPI lies within project teams (SEI 2006).

There are also several comparative studies (Anderson 2005; Baker 2005; Glazer et al. 2008; Fritzsche and Keil 2007; Jakobsen and Johnson 2008; Kähkönen and Abrahamsson 2004; Marcal et al. 2008; McMichael and Lombardi 2007;

Nawrocki et al. 2006; Paulk 2001; Pikkarainen and Mäntyniemi 2006; Reifer 2003; Sutherland et al. 2007; Schwaber 2004) that have found correspondences between the traditional maturity models and the agile approach/methods. Paulk (2001), for instance, states that CMM and XP contain ideas that can be synergistic. Baker (2005) concludes that there is the possibility for the organization to follow the CMMI goals and still use the agile practices in the software development. Companies using CMMI provide support to agile development making it an easier and smoother method to adopt than in non-CMMI organizations (Jakobsen and Johnson 2008).

Furthermore, there are empirical studies (Anderson 2005; Baker 2005; Bos and Vriens 2004; Kähkönen and Abrahamsson 2004; Paulk 2001) reporting on the use of CMM/CMMI for assessment or as a basis for improvement in organizations or projects employing agile practices. There are also some case studies (Baker 2005; Kähkönen and Abrahamsson 2004; Paulk 2001; Schwaber 2004) that describe how the CMM/CMMI and agile practices have been successfully applied together to formulate the so-called combined improvement approach.

To conclude, the application of the agile approach and methods enables the achievement of CMMI lower levels with some exceptions. Thus, it would be possible, at least in principle, to apply the traditional maturity models to assess agile development as well. The biggest problem is, however, that the traditional maturity models do not address agility in an explicit manner. Secondly, the models have been built with the goal of strictly defined processes, which is contradictory to agile development as an empirical development process (Schwaber and Sutherland 2010). Thirdly, the maturity models have been developed for large organizations, and many of the companies applying the agile approach are small or medium sized.

4 Analysis of Agile Maturity Models

In this section, we make a comparative analysis of agile maturity models to find out how they compare to each other. We use the following criteria: (1) *Domain*: What are the domains the models are targeted to? (2) *Purpose*: What are the purposes the models have been developed for? (3) *Conceptual and theoretical bases*: What are the conceptual and theoretical backgrounds upon which the models have been built? (4) *Approaches and principles*: What are the approaches and principles (e.g., top-down, bottom-up) used to construct the models? (5) *Structure*: What are the architectures of the models? (6) *Use and validation*: To what extent the models have been deployed and validated?

After a careful review of relevant articles in journals and proceedings, as well as by using Google, we selected eight artifacts that can be considered agile maturity models (see Table 27.1). Some of them (Ambler 2010; Nawrocki et al. 2001; Patel and Ramachandran 2009; Pettit 2006) are explicitly called maturity models, while the others provide maturity levels (Lui and Chan 2005) or in some other way

Table 27.1 Agile maturity models

Ref.	Maturity models	Levels	Domain components
Ambler (2010)	AMM	Levels: 1–5	–
Lui and Chan (2005)	Road map	Levels: 1–4	XP practices
Nawrocki et al. (2001)	XPMM	Levels: 1–4	Specific practices/key process areas
Packlick (2007)	Agility maturity map	Levels: 1–5	Agile goals
Patel and Ramachandran (2009)	AMM	Levels: 1–5	Key process areas
Pettit (2006)	"Agile maturity model"	–	Dimensions: 1–6
Qumer and Henderson-Sellers (2008)	AIIM	Blocks: 1–3 Levels: 1–6	Agile properties/agile practices
Sidky et al. (2007)	Agile adoption framework	Levels: 1–5	Agile principles/agile practices

enable maturity assessment of agile development (Packlick 2007; Qumer and Henderson-Sellers 2008; Simpson and Weiner 1989). Some of them consider agile development in general, while the others have a more limited scope, such as XP practices (Lui and Chan 2005; Nawrocki et al. 2001). In the following, we compare these models based on the criteria above. The results are summarized in Table 27.1 and Appendix A.

Ambler (2010) suggests the AMM (agile maturity model) "to provide guidance to an [software] organization to help improve effectiveness at agile software development." The model contains five maturity levels with no dimensions, domain components, and argumentation for the levels. Lui and Chan (2005) propose a road map, in the form of four stages, for the adoption of XP practices in a software team. The maturity stages have been established by clustering XP practices based on their interrelations (Beck 1999). Nawrocki et al. (2001) provide a 4-level maturity model for eXtreme Programming, called XPMM, to indicate the risks associated with a project (i.e., the higher the level, the lower the risk). The model deploys some specific practices (e.g., acceptance test, planning game) and key process areas (e.g., customer relationship management, product quality assurance) from CMMI level 2 (SEI 2006).

Packlick (2007) presents the agile maturity map (AMM) to help "accelerate change, deepen the degree of understanding, and increase their success in execution." It has been developed for a certain company, to be used as a mindset for thinking about agile adoption in terms of goals rather than practices. They have derived five agile goals (e.g., acceptance criteria, green-bar tests and builds, engineering excellence) from the values and principles of the Agile Manifesto and used them as the dimensions in the setting of an individual company. Patel and Ramachandran (2009) propose an agile maturity model (AMM) "to improve and enhance the agile software development methodology and boost up the agile principles and objectives such as lower costs, customer satisfaction and software quality" (ibid, p. 6). They define level-specific key process areas (e.g., project planning, on-site customer availability) and for them large sets of questions for the assessment of the current state of an organization or project. The AMM is said to be based on the agile values, practices, and principles. It is not, however, shown how the levels have been derived from those (Patel and Ramachandran 2009). Pettit (2006) presents an "agile maturity model" that "allows an organization to assess its current state, set targets, plot a course for continuous improvement, and build and measure against a business case for adopting Agile practices." He defines six process areas (e.g., requirement engineering, testing, collaboration) along which the current state is to be assessed and the desired state is to be set. Each of them is said to be able to "inhibit or engender IT's responsiveness to changes in the business environment." No explanation is given for the selection of the process areas, and the model does not provide any unified maturity levels.

Qumer and Henderson-Sellers (2008) present a large framework, called the Agile Software Solution Framework (ASSF), to help organizations adopt an agile approach rapidly and effectively. The framework contains the Agile Adoption and Improvement Model (AAIM) which "guides a software organization to adopt and improve agile practices for a specific situation or project" (ibid, p. 1909). In the AAIM model, six maturity levels are grouped into three blocks, called prompt, crux, and apex, and for each level a set of agile properties and practices are defined. Sidky et al. (2007) suggest the agile adoption framework which contains five agile levels providing different degrees of agility for a project or organization. It can be used for "the assessment of a particular level of agility adoption and to advance to the next level" (ibid, p. 215). The principles in the Agile Manifesto have been grouped into five agile principles (e.g., embrace change to deliver customer value, plan and deliver software frequently), and agile practices have been positioned in the cells of the matrix defined by the agile levels (1–5) and the agile principles (A–E).

Next, we consider more closely how maturity is seen to grow when progressing toward higher levels of the models (cf. Appendix A). The AMM (Ambler 2010) assumes that on the lower levels agile practices, mainly concerning individual teams, are applied in a superficial way, and while progressing onto the upper levels, the approach is more deeply understood and the practices are applied within a broader scope (cf. governance framework, organizational ecosystem). On the highest level, measurement and optimization are deployed. In the model by Lui and Chan (2005), the lower levels address testing, simple design, refactoring, and continuous integration, whereas the higher levels contain XP practices such as pair programming, collective ownership, small releases, and on-site customers. In the XPMM (Nawrocki et al. 2001), it is not possible to find any unified principle while going into higher levels of agile maturity.

The agile maturity map (Packlick 2007) starts with awareness of agile values and goals, proceeds with putting them into use, and reaches the optimized way of developing. The highest level means contributing to learning across the organization. In the AMM (Patel and Ramachandran 2009), special focus areas are defined for each level. The level 2, for instance, focuses on project planning, customer and stakeholder's orientation, value, and collaboration. In (Qumer and Henderson-Sellers 2008) the focus shifts from good communication and cooperation and the production of executable artifacts to the valuing of the people and the establishment of a learning environment and finally to a lean production environment. In the agile adoption framework (Simpson and Weiner 1989), the focus changes from being collaborative and evolutionary to effectiveness and adaptiveness and finally to sustaining and fostering agility throughout an organization.

Because in most of the maturity models the agile practices are positioned onto the maturity levels, it is interesting to see how their positions compare to each other. Here we give some examples of them (see more closely Appendix A). Continuous delivery is positioned on level 1 (Lui and Chan 2005), level 2 (Simpson and Weiner 1989), and level 3 (Patel and Ramachandran 2009). Pair programming can be found on level 3 (Lui and Chan 2005; Nawrocki et al. 2001; Patel and Ramachandran 2009) and level 5 (Simpson and Weiner 1989) and refactoring on level 1 (Lui and Chan 2005) and level 3 (Simpson and Weiner 1989). Forty-hour day is mentioned on level 4 (Lui and Chan 2005; Nawrocki et al. 2001). In addition, working hours in (Patel and Ramachandran 2009) (level 4) and the value of people in (Qumer and Henderson-Sellers 2008) (level 4) are related to this. Self-organizing teams are positioned on level 1 (Ambler 2010) and level 3 (Simpson and Weiner 1989) and unit test on level 1 (Lui and Chan 2005) and level 3 (Simpson and Weiner 1989). Communication and cooperation are focused on level 1 (Simpson and Weiner 1989), level 2 (Qumer and Henderson-Sellers 2008), and level 3 (Patel and Ramachandran 2009). As can be seen from these examples, the models are quite different in their views of agile practices on the maturity levels.

Finally, we analyze the conceptual and theoretical bases, use, and validation of the maturity models. The conceptual basis means how the models have been anchored to the key notions of agility (cf. Chap. 2). Ambler (2010), Patel and Ramachandran (2009), Pettit (2006), and Sidky et al. (2007) do not discuss the concepts in any ways. Lui and Chan (2005) and Nawrocki et al. (2001) adopt the narrow view of XP; Packlick (2007) extend this with the Agile Manifesto's view. Qumer and Henderson-Sellers (2008) discuss agility in terms of five key attributes (flexibility, speed, leanness, learning, and responsiveness), including also the Agile Manifesto. None of the maturity models are established or discussed from the viewpoint of successful agile development. None of the models have been grounded on theoretical underpinnings. Qumer and Henderson-Sellers (2008) do mention the use of grounded theory and industrial analysis in devising the AAIM, but no theory underlying the structures and contents of the model is named.

The analyzed models can be categorized into three groups depending on how they have been used and validated. The first group contains the models, which are merely based on conceptual studies (Lui and Chan 2005; Nawrocki et al. 2001) or personal opinions (Ambler 2010; Pettit 2006). The model in the second group (Packlick 2007) has been developed within and for the single organization, and some experiences are reported. The third group (Patel and Ramachandran 2009; Qumer and Henderson-Sellers 2008; Simpson and Weiner 1989) contains the models from which larger experience and comments have been gotten. The

AMM (Patel and Ramachandran 2009) was discussed in three different organizations, and comments are reported. The AAIM (Qumer and Henderson-Sellers 2008) was deployed in two pilot projects with positive experience. Sidky et al. (2007) presented the framework to 28 members of the agile community in 90-min personal visit with the participants. The feedback was positive. Based on the above, we can conclude that none of the models have been properly validated. On the other hand, the validation of a maturity model is quite challenging (Becker et al. 2009).

5 Discussion and Conclusion

The maturity models analyzed above are mainly targeted at the assessment of adoption of agile values, principles, and practices in an organization or project. The models can be divided into four categories, based on their purpose of use: (1) road maps supporting the adoption of agile approach and practices in an organization (Lui and Chan 2005), (2) frameworks guiding the selection of agile practices for an individual project (Qumer and Henderson-Sellers 2008; Simpson and Weiner 1989), (3) frameworks for the assessment of the current state of the art in the adoption (Ambler 2010; Nawrocki et al. 2001; Packlick 2007; Patel and Ramachandran 2009; Pettit 2006), and (4) frameworks for furthering the deployment of agile values and principles in an organization (Patel and Ramachandran 2009).

Generally speaking, we can distinguish the following dimensions along which the maturity levels can be defined. (1) *Easiness of adoption*: Easy-to-adopt agile principles and practices are positioned on the lower levels, and more challenging ones on the higher levels. (2) *Essence of agility*: Core agile principles and practices are located onto lower levels. The core here means the principles and practices which *must* be involved for development being called agile and by which rapid benefits are achieved. (3) *Levels of understanding*: Understanding of what agile values and principles mean becomes deeper on the higher levels. (4) *Degree of process standardization*: Behavior on the lower levels is more ad hoc-like, and it becomes more standardized and optimized on the higher levels (cf. CMMI, ISO). (5) *Focus*: The focus area expands from a single-team-single-customer-single-product setting (Sidky et al. 2007) to cover product management and the running of an entire company with its ecosystem. In this sense, we can distinguish *agile-in-small* and *agile-in-large*. In addition, there is *agile-in-complex* which means the deployment of agile approach in distributed, large, global multicultural software development projects and organizations (Rohunen et al. 2010). Enterprise-level agility relates to strategic agility (Doz and Kosonen 2008), lean principles (Poppendijk and Poppendijk 2003), and “beyond the budget” movement (Bogsnes 2009; Lohan et al. 2010).

The dimensions are visible in different ways in the maturity models. In Ambler (2010) and Packlick (2007) degree of understanding, level of standardization, and focus are obvious. In Patel and Ramachandran (2009) degree of standardization and process improvement is highlighted. In the other models, the dimensions are more or less invisible. Most of the models have been derived from the values and principles of the Agile Manifesto (Agile 2002). Packlick (2007) has defined the so-called agile goals, (Qumer and Henderson-Sellers 2008) certain focus areas, and (Simpson and Weiner 1989) the so-called agile principles. Lui and Chan (2005) uses the XP practices to populate the maturity levels.

Based on the analysis above, we can conclude that the agile maturity models are still at their early phases (cf. the life cycles of the maturity models (Becker et al. 2009; de Bruin et al. 2005; Fraser et al. 2002; Mettler and Rohner 2009)). Much research work, both conceptual and empirical, is needed to make them conceptually sound and applicable in practice. The maturity models should be established on a sound conceptual basis, derived from the key notions of agility and successful agile development, and structured with well-defined dimensions, levels, and domain components. On the higher levels, issues of the organization-level maturity could be partly derived from lightweight process assessment models (e.g., Petterson et al. 2008).

In the research to come, our aim is to deepen our analysis of the existing maturity models, apply some of them to assess the state of agile adoption in real settings, and make suggestions to extend them with agile-in-large and situational issues.

Appendix A. Maturity Levels and Agile Principles and Practices in the Maturity Models

Ref.	Level 1	Level 2	Level 3	Level 4	Level 5	Level 6
Ambler (2010)	*Rhetorical stage*	*Certified stage*	*Plausible stage*	*Repeatable stage*	*Measured stage*	–
	Self-organizing teams, do not respect mgmt	Certification courses	Scaling agile strategies	Produce solutions, not just software, recognize constraints of ecosystems	Information collection for improvement	
Lui and Chan (2005)	*Stage 1*	*Stage 2*	*Stage 3*	*Stage 4*	–	–
	Testing, simple design, refactoring, coding standards	Continuous integration	Pair programming, collective ownership	Metaphor, 40-h week, small release, on-site customer, planning game		
Nawrocki et al. (2001)	*Not compliant*	*Initial*	*Advanced*	*Mature*	–	–
		Acceptance tests, planning game, customer rel. mgmt, product quality assur.	Pair programming	Customer's and developer's satisfaction, on-site customer, 40-h day		
Packlick (2007)	*Awareness*	*Transformation*	*Breakthrough*	*Optimizing*	*Mentoring*	–
	Understanding of the AGILE goals	Developing practices to satisfy the AGILE goals	Agile practices are used consistently to satisfy the AGILE goals	Improvements in the AGILE goals, creative innovations in improving	High performance teams mentoring other teams, organizational learning	
Patel and		Ramachandran, (2009)	*Initial*	*Explored*	*Defined*	*Improved*
	Mature	–				

(continued)

Ref.	Level 1	Level 2	Level 3	Level 4	Level 5	Level 6
The organization operates in its own unique way	Project planning Agile req's engineering Customer and stakeholder orientation	Customer relationship mgmt, frequent deliveries, pair programming, TDT, coding standards	Project mgmt, working hours, self-organizing teams, risk assessment	Continuous improvement, uncertainty mgmt, defect prevention		
Qumer and Henderson-Sellers (2008)	*Agile infancy*	*Agile initial*	*Agile realization*	*Agile value*	*Agile smart*	*Agile progress*
	Introduction to basic agile properties	Good communication and cooperation	Production of executable artifacts	The value of people within and outside organization	The establishment of a learning environment	A lean
production environment						
Sidky et al. (2007)	*Collaborative*	*Evolutionary*	*Effective*	*Adaptive*	*Encompassing*	–
	Communication and collaboration	Early and continuous delivery of software	Efficiency of the process High quality working software	Responding to change through multiple levels of feedback	Establishing a vibrant environment to sustain agility.	

References

Abrahamsson P, Salo O, Ronkainen J, Warsta J (2002) Agile software development methods – review and analysis. VTT Publications 478

Agile Alliance (2002) Agile Manifesto. http://www.agilealliance.org/

Ambler S (2010) The agile maturity model (AMM). Dr Dobb's. http://www.drdobbs.com/architecture-and-design/224201005;jsessionid=P1KHI0JRB4C5ZQE1GHPCKH4ATMY32JVN

Anderson DJ (2005) Stretching agile to fit CMMI level 3 – the story of creating MSF for CMMI® Process Improvement at Microsoft Corporation. In: Proceedings of the agile development conference (ADC'05), pp 193–201

Baker S (2005) Formalizing agility: an agile organization's journey towards CMMI accreditation. In: Proceedings of the agile development conference (ADC'05), pp185–192

Beck K (1999) Extreme programming explained: embrace change. Addison-Wesley, Reading

Becker J, Knackstedt R, Pöppelbuss P (2009) Developing maturity models for IT management – a procedure model and its application. Bus Inf Syst Eng 1(3):213–222

Boehm B, Turner R (2003) Balancing agility and discipline. In: Balancing agility and discipline – a guide for the perplexed. Addison-Wesley, Boston

Bogsnes B (2009) Implementing beyond budgeting: unlocking the performance potential. Wiley, Hoboken

Bos E, Vriens C (2004) An agile CMM. In: Proc of the 4th conference on extreme programming and agile methods – XP/Agile Universe, pp 129–138

Chow T, Cao D-B (2008) A survey study of critical success factors in agile software projects. J Syst Softw 81(6):961–971

Cohn M (2010) Succeeding with agile – software development using Scrum. Addison-Wesley, Upper Saddle River

Cohn M, Ford D (2003) Introducing an agile process to an organization. IEEE Softw 36(6):74–78

Conboy K (2009) Agility from first principles: reconstructing the concept of agility in information systems development. Inf Syst Res 20(3):329–354

de Bruin T, Freeze R, Kulkani U, Rosemann M (2005) Understanding the main phases of developing a maturity assessment model. In: Proceedings of 16th Australasian conference on information systems

Doz Y, Kosonen M (2008) Fast strategy: how strategic agility will help you stay ahead of the game. Wharton School Publishing, Harlow

Fayad M, Laitinen M (1997) Process assessment considered wasteful. Commun ACM 40 (11):125–128

Fraser P, Moultrie J, Gregory M (2002) The use of maturity models/grids as a tool in assessing product development capability. In: Proceedings of the IEEE international engineering management conference, pp 244–249

Fritzsche M, Keil P (2007) Agile methods and CMMI: compatibility or conflict? e-Inf Softw Eng J 1(1):9–26

Galin D, Avrahami M (2006) Are CMM program investments beneficial? Analysing past studies. IEEE Softw 23(6):81–87

Glazer H, Dalton J, Anderson D, Konrad M, Shrum S (2008) CMMI or agile: why not embrace both! Software engineering process management. Software Engineering Institute. CMU/SEI-2008-TN-003, Pittsburg

Herbsleb J, Goldenson D (1996) A system survey of CMM experience and results. In: Proceedings of 18th ICSE'96, pp 323–330

Highsmith J (2002) Agile software development ecosystems. Addison-Wesley, Boston

Humphrey WS (2005) PSP: a self-improvement process for software engineers. Software Engineering Institute, Carnegie Mellon, Pittsburgh

Humphrey W, Snyder T, Willis R (1991) Software process improvement at Hughes Aircraft. IEEE Softw 8(4):11–23

ISO/IEC 15504 (2008) International standard information technology – software process assessment: Part 1 – Part 7. International Organization for Standardization

Jakobsen C, Johnson K (2008) Mature agile with a twist of CMMI, In: Proceedings of agile 2008 conference, pp 212–217

Jiang J, Kleinb G, Hwange H-G, Haungc J, Hungc S-Y (2004) An exploration of the relationship between software development process maturity and project performance. Inf Manage 41:279–288

Kähkönen T, Abrahamsson P (2004) Achieving CMMI level 2 with enhanced extreme programming approach. In: Proceedings of the conference on product focused software process improvement, pp 378–392

Krebs W, Kroll P (2008) Using evaluation frameworks for quick reflections. Agile J (February 9)

Lami G, Falcini F (2009) Is ISO/IEC 15504 applicable to agile methods? In: Proceedings of XP 2009 conference, pp 130–135

Lohan G, Conboy K, Lang M (2010) Beyond budgeting and agile software development: a conceptual framework for the performance management of agile software development teams. In: Proceedings of ICIS 2010 conference

Lui K, Chan K (2005) A road map or implementing extreme programming. In: Proceedings of international software process workshop (SPW 2005), Beijing, pp 474–481

Lycett M, Macredie R, Patel C, Paul R (2003) Migrating agile methods to standardized development practice. IEEE Comput 36(6):79–85

Marcal A, de Freitas B, Furtado Soares F, Furtado M, Maciel T, Belchior A (2008) Blending Scrum practices and CMMI project management process areas. Innov Syst Softw Eng 4:17–29

McMichael B, Lombardi M (2007) ISO 2001 and agile development. In: Proceedings of AGILE 2007 conference, pp 262–265

Mettler T, Rohner P (2009) Situational maturity models as instrumental artifacts for organizational design. In: Proceedings of 4th international conference on design science research in information systems and technology (DESTRIST'09)

Misra SC, Kumar V, Kumar U (2009) Identifying some important success factors in adopting agile software development practices. J Syst Softw 82(11):1869–1890

Nawrocki J, Walter B, Wojciechowski A (2001) Towards the maturity model for extreme programming. In: Proceedings of 27th Euromicro conference, Los Alamitos, pp 233–239

Nawrocki J, Walter B, Wojciechowski A (2006) Comparison of CMM level 2 and eXtreme programming. In: Proceedings of software quality (ECSQ 2002), LNCS 2349, pp 288–297

Packlick J (2007) The agile maturity map: a goal oriented approach to agile improvement, In: Proceedings of AGILE 2007 conference, pp 266–271

Patel C, Ramachandran M (2009) Agile maturity model (AMM): a software process improvement framework of agile software development practices. Int J Softw Eng 2(1):3–28

Paulk M (2001) Extreme programming from a CMM perspective. IEEE Softw 18(6):19–26

Paulk M, Curtis B, Chrissis M, Weber C (1993) The capability maturity model for software, version 1.1 (No. CMU/SEI-93-TR-24). Software Engineering Institute

Petterson F, Ivarsson M, Gorschek T, Öhman P (2008) A practitioner's guide to light weight software process assessment and improvement planning. J Syst Softw 81(6):972–995

Pettit R (2006) An "agile maturity model?" Agile J. www.agilejournal.com

Pikkarainen M, Mäntyniemi A (2006) An approach for using CMMI in agile software development assessments: experiences from three case studies. In: Proceedings of SPICE 2006 conference, Luxenburg

Poon P, Wagner C (2001) Critical success factors revisited: success and failure cases of information systems for senior executives. Decis Support Syst 30(4):393–418

Poppendijk M, Poppendijk T (2003) Lean software development – an agile toolkit. Addison-Wesley, Boston

Procaccino J, June M, Steven J (2006) Defining and contributing to software development success. Commun ACM 49(8):79–83

Qumer A, Henderson-Sellers B (2006) Comparative evaluation of XP and Scrum using the 4D analytical tool (4-DAT). In: Irani Z et al (eds) Proceedings of the European and Mediterranean conference on information systems 2006 (EMCIS2006)

Qumer A, Henderson-Sellers B (2008) A framework to support the evaluation, adoption, and improvement of agile methods in practice. J Syst Softw 81(11):1899–1919

Reifer D (2003) XP and CMM. IEEE Softw 20(3):14–15

Rohunen A, Rodriguez P, Kuvaja P, Krzanik L, Markkula J (2010) Approaches to agile adoption in large settings: comparison of the results from a literature analysis and an industrial inventory. In: Proceedings of PROFES 2010 conference, Springer LNCS 6156, pp 77–91

Salo O, Abrahamsson P (2007) An iterative improvement process for agile software development. Softw Proc Improv Pract 12:81–100

Schwaber K (2004) Agile project management with Scrum. Microsoft Press, Washington, DC

Schwaber K, Sutherland J (2010) Scrum guide. http://www.scrum.org/storage/scrumguides/Scrum%20Guide.pdf

SEI (2006) CMMI® for development, Version 1.2 – Improving processes for better products. Carnegie Mellon. Software Engineering Institute. http://www.sei.cmu.edu/reports/06tr008.pdf

Shalloway A, Beaver G, Trott J (2009) Lean-agile software development: achieving enterprise agility. Addison-Wesley, Upper Saddle River

Sidky A, Arthur J, Bohner S (2007) A disciplined approach to adopting agile practice: the agile adoption framework. Innov Syst Softw Eng 3(3):203–216

Simpson J, Weiner E (1989) The Oxford English dictionary. Oxford University Press, Oxford

Stålhane T, Hanssen G (2008) The application of ISO 2001 to agile software development. In: Proceedings of conference on product focused software process improvement, pp 371–385

Sutherland J, Jakobsen C, Johnson K (2007) CMMI and Scrum – a magic potion for code warriors. In: Proceedings of agile 2007 conference

Turner R, Jain A (2002) Agile meets CMMI: culture clash or common cause? In: Extreme programming and agile methods – XP/Agile Universe 2002, Springer LNCS 2418, pp 60–69

Williams L, Layman L, Krebs W (2004) Extreme programming evaluation framework for object-oriented languages, version 1.4. North Carolina State University Department of Computer Science, TR-2004-18

Chapter 28
Simulation-Based Evaluation of Adaptive Applications

Jānis Grabis and Inese Šūpulniece

1 Introduction

Enterprise applications (EA) such as enterprise resource planning systems are used to execute business processes. Usually, these are packaged applications providing standardized implementations of business processes. These standardized implementations can be customized and modified to some extent as the result of implementation activities. Users of EA either use predefined workflows or rely on user documentation and best practices to execute their business processes (Curran and Ladd 2000). Besides these standard capabilities, in many cases, users also can use other functions provided by EA subject to their access rights. That means that users have possibilities to introduce their own variations in process execution. By considering these variations, users might come up with more efficient ways of executing business processes (Topi et al. 2005). If an EA supports users in identification of more efficient variations of business process execution and enables for continuous execution refinement, it is referred as to adaptive EA.

There are multiple ways EA could be adapted. Adaptation ranges from adaptation of underlying business processes (Reichert et al. 2009; Zeng et al. 2010) to user interface adaptation (Singh and Wesson 2009a). This chapter approaches the adaptation problem from the operations management perspective and aims to discover the most efficient ways to execute business process using an enterprise application. The efficiency is measured by execution speed and quality (i.e., process is executed without errors and successfully).

The objective of this chapter is to develop a simulation model for evaluation of process execution performance-oriented enterprise application adaptation. The model defines characteristics of EA relevant to their adaptation, proposes a simple adaptation algorithm, and allows analyzing impact of adaptation on performance of

J. Grabis (✉) • I. Šūpulniece
Institute of Information Technology, Riga Technical University, Kalku 1, Rīga LV-1658, Latvia
e-mail: grabis@iti.rtu.lv; inese.supulniece@rtu.lv

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_28,

EA. The EA are characterized from purely technical perspective regardless of their business function. The model is used to evaluate two hypotheses: (1) adaptation is able to identify the most efficient process execution variants, and (2) changing process characteristics leads to different process configurations. These results would serve as the basis for further investigations regarding personalization of EA with respect to users' characteristics and identification and evaluation of different adaptation mechanism, and the model will be used for evaluation of productivity improvement solutions.

The rest of this chapter is organized as follows: Sect. 2 discusses general aspects of process execution adaptation. The adaptation model is elaborated in Sect. 3. Experimental results are reported in Sects. 3 and 4. Section 5 concludes.

2 Representation of Adaptive Applications

As stated earlier, the proposed approach adapts the enterprise application from the operational perspective and attempts to identify process execution variants providing better transactional efficiency. The transactional efficiency measures how easy it is to execute business functions with required speed and reliability (Keystone 2007). Process execution speed and quality are also traditional measures used in evaluation of physical processes (Anupindi et al. 2006). Typical functions of using EA include (1) searching, (2) transaction processing (create, edit, delete), (3) inquiring, and (4) reporting. Similar functions also have been identified in research on task modeling (Barclay et al. 2003). Each function has a specific execution time, and there is also a waiting time before the function execution is initiated (Anupindi et al. 1999). Errors, both technical and logical, can be made when performing each function. Because of their impact on process execution quality, ensuring execution correctness is of major concern (Singh and Wesson 2009b). Therefore, the adaptive application can be represented as consisting of generic functions with attributes characterizing execution speed and quality.

A representation of the adaptive application, which is used to execute a business process P, should provide means for defining available functions and their attributes as well as for representing alternative process execution sequences. It is assume that the adaptive application provides as set of functions $(f_1, \ldots, f_n)$ available for process execution. There are no strict workflows though preconditions for each function are defined, and Π_i contains all the immediate successors of f_i. A function execution depends only upon the previous function. Figure 28.1 illustrates functions provided by the enterprise application and their possible execution sequences (a), the best practice process (BPP) (b), and a sample user-adapted process (c).

2.1 Attribute and Performance Measures

Each function is defined by a set of execution attributes (Table 28.1). The attributes are defined assuming that usage of EA can be characterized by time and quality.

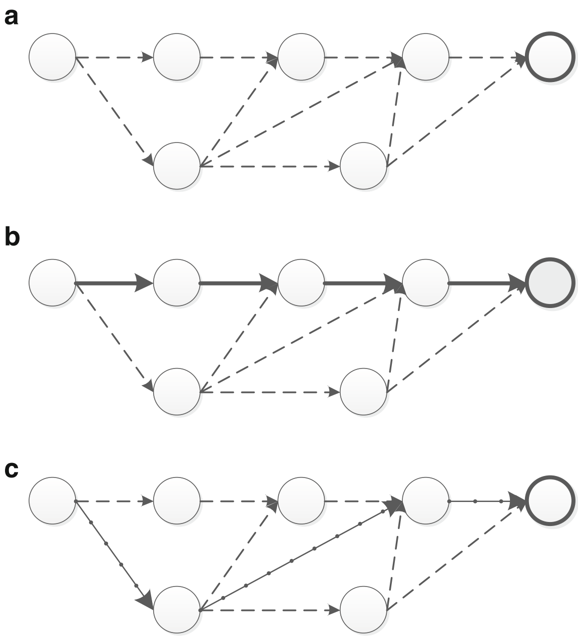


Fig. 28.1 *Circles* denote available functions, *dashed lines* denote available execution sequences, *solid lines* denote the BPP execution sequence, and *solid lines with circles* denote a process execution variant

Table 28.1 Attributes of enterprise application functions

Notation	Description
t_i	Processing time
τ_i	Wait time
e_i	Probability to commit an error during the execution of the ith function
ε_i	Probability to discover and to correct an error during the execution of the ith function
r_i	Probability to return process execution to the previously executed function
ρ_i	Rework coefficient for the ith function

The processing time characterizes time necessary to perform a function in the enterprise application, for instance, filling out a data input form. The wait time refers to synchronization of function execution with other activities of the actual business process, for instance, waiting for a manager to approve a request posted in the enterprise application. Different kinds of errors could be made when performing the function. That is captured by e_i. A number of functions in EA are specifically performed to discover previously committed errors. ε_i describes the error discovery success. The error discovery leads to reduction of possibility to obtain erroneous process outcomes. Many functions in EA are invoked to request lookup information or to verify previous functions. In this case, performing the function results to returning to the previous function, where some modifications are made. r_i characterizes the return probability and ρ_i describes effort needed for modification.

The process execution performance is measured by four performance measures: total processing time, total wait time, total level of erroneousness, and the execution success. The total processing time T and the total waiting time Ψ are calculated as sums of processing time and waiting time, respectively, for all activities included in process execution sequence. The total level of erroneousness characterizes the probability that errors have been committed and have not been corrected during the process execution and it is computed as

$$E = \sum_{i \in P} e_i - \sum_{i \in P} \varepsilon_i, E \in (0, 1).$$

The process execution success is evaluated upon completing the process, and it depends upon E:

$$R = \begin{cases} 1, x \geqslant E \\ 0, x < E \end{cases},$$

where $x \sim U(0, 1)$. $R = 1$ means that process execution has failed, and $R = 0$ means that process execution has succeeded. That implies that a high level of E reduces chances of successfully completing the process.

The total process execution score characterizing overall process execution performance is computed as

$$S = w_1 T + w_2 \Psi + w_3 E + w_4 R,$$

where w_1, w_2, w_3, w_4 are weights characterizing the relative contribution of each performance measure to the overall score. A similar function execution score s_i also can be computed for each function individually using its attributes rather than the total process performance measures.

2.2 Adaptation Algorithm

Processes are executed by users of EA. The users either follow the standard process or attempt to find a more efficient process variant by exploring other function provided by the enterprise application. Behavior and preferences of the users also can be defined using a set of attributes (e.g., Ardissono and Goy 2000). In the context of this chapter, the user is characterized by willingness to deviate from the BPP and by willingness to innovate during the process execution. The former attribute is represented as a probability γ_k that during the process execution the kth user will select other function rather than that prescribed by the BPP. The latter attribute is represented as a probability λ_k that the kth user will try other functions rather than selecting the function prescribed by the most efficient adaptive process.

The process adaptation algorithm models behavior of users as they attempt to perform their process using the EA. The adaptation objective is to minimize the process execution score $S_,$ and a greedy algorithm is used to achieve that. It is assumed that the process is executed multiple times, and each completed execution sequence is added to the set of process variants. In subsequent process executions, the set of process variants is used to lookup the most efficient process execution variants.

The simulated process adaptation algorithm performs the following activities:

1. Start execution of the process from the initial activity for the kth user.
2. Select the next activity to be performed from the set of successors:
   - 2.1. There is $1 - \gamma_k$ chance that the user will follow the BPP and γ_k chance that the user will follow her/his own variant of the process.
   - 2.2. If BPP is followed, select the next function according to the BPP.
   - 2.3. If a variant of the standard process is created, the user either seeks to follow the best variant identified (with probability λ_k) so far or innovates (with probability $1 - \lambda_k$):
     - 2.3.1. In the former case, the user finds the process variant containing the current function and having the lowest S value and selects the next function according to precedence settings of the process variant found.
     - 2.3.2. In the latter case, the user randomly selects one of the successor functions of the current function.
3. Update the overall process performance measures.
4. Determine whether the process should return to the previous activity according to $r_{i:}$
   - 4.1. If $x<r_{i,}$ return to the previous activity, update the overall process performance measures taking into account the rework coefficient, and go to step 2.
   - 4.2. If $x \geqslant r_{i,}$ go to step 2.
5. If the final function has been completed, then compute the process score, add the process instance to the set of process variants, and end the process.

The process execution failure or success is evaluated upon completing the process. In step 2.3, the user has a choice between following the best adapted process variant and attempting to identify her/his own process execution variant. Identification of the optimal process depends upon those users willing to pursue unexplored process execution paths. In some cases, the user might come up with less efficient process execution variant, but otherwise, there would not be a way to find out existence of more efficient process execution variants.

3 Experimental Studies

The experimental studies are used to evaluate the application adaptation process. Two scenarios are analyzed (Fig. 28.2). In the case of scenario S1, there is an option to skip one function F2 (for instance, a request is submitted without completing the optional compliance test). This function allows reducing the possibility to commit an error though it has substantial waiting time.

In the case of scenario S2, there are four possible paths to complete the process. The functions differ by associate waiting time, and error discovery occurs only at functions included in the BPP. The function innovation attribute for all users is set $\lambda_k = 0.5$. The experimental factors considered are the process innovation γ_k and the weight factor for the process execution failure w_4. Other weight factors are set $w_1 = 1, w_2 = 0.1$, and $w_3 = 1$. Changing value of w_4 represented a changing balance between process execution speed and quality. Small values of w_4 should favor variants emphasizing execution speed, while large values of w_4 should favor variants emphasizing quality (i.e., reducing possibility of process execution failure).

In order to test the ability of the adaptation algorithm to find the most efficient process variant, the process execution is repeated 100 times. Each time the process

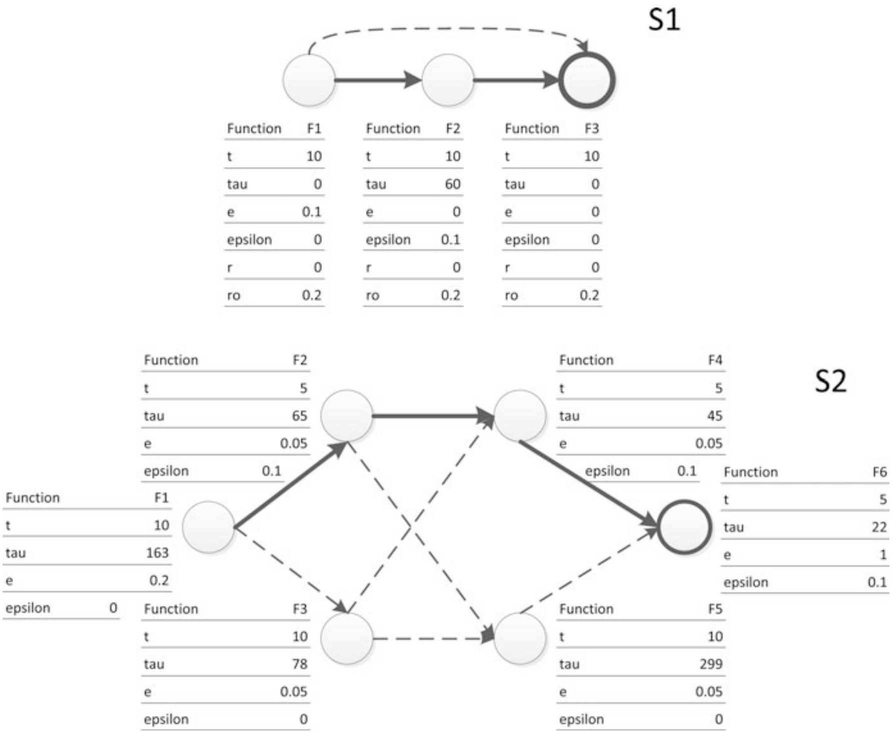


Fig. 28.2 Enterprise application adaptation scenarios

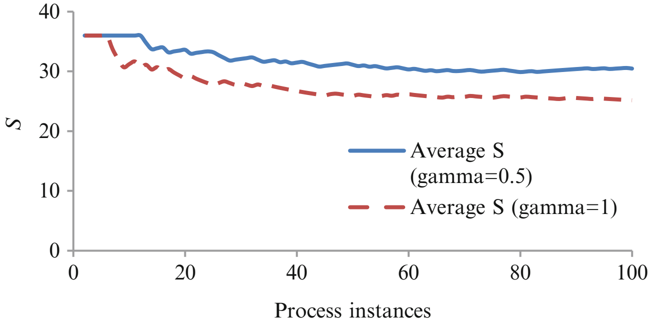


Fig. 28.3 The average process execution score depending upon the number of process instances executed, $w_4 = 10$

Table 28.2 The average score and variant observation count according to the process execution variant for scenario S1

	$\gamma_k = 0.5, w_4 = 10$		$\gamma_k = 1, w_4 = 10$		$\gamma_k = 0.5, w_4 = 25$		$\gamma_k = 1, w_4 = 25$	
Process variant	S	Count	S	Count	S	Count	S	Count
F1F2F3[a]	36.0	64	36.0	30	36.0	60	36.0	31
F1F3	20.7	36	20.5	70	22.6	40	24.4	69
Total	30.5	100	25.2	100	30.6	100	28.0	100

[a]BPP

execution score and the process variant are recorded. Figure 28.3 shows the process execution score averaged over all process instances up to the current process instance. The BPP has the process execution score $S = 36$, and it can be observed that with $\gamma_k = 1$ the adaptation algorithm quickly identifies that there is a more efficient process variant including only activities F1 and F3 (Table 28.2). The convergence is substantially slower for $\gamma_k = 0.5$. Table 28.2 also shows that willingness to explore alternative process execution variants leads to better values of process execution score for scenario S1. Even in the case of $\gamma_k = 0.5$, the most efficient process execution sequence is discovered more frequently than the other variant. However, this result substantially depends upon the relative importance of each process performance measure. If w_4 is increased to 25, there are process instances having worse score (if the chance of process failure has realized) than the BPP, but the process variant still remains more efficient on average.

In the case of scenario S2, the BPP is also the most efficient process and there are no process variants yielding the lower process execution score (Table 28.3). Therefore, the average score is lower for $\gamma_k = 1$ because the users try to explore all possible execution variants. However, the loss of execution efficiency is relatively small because in majority of situations the user is able to find the best variant, which coincides with the BPP.

The results obtained for S1 and S2 confirm that adaptation allows finding the most efficient process execution variant.

Table 28.3 The average score and observation count according to the process execution variant for scenario S2

	$\gamma_k = 0.5, w_4 = 10$		$\gamma_k = 1, w_4 = 10$	
Process variant	S	Count	S	Count
F1F2F4F6[a]	65.5	83	65.5	52
F1F3F4F6	71.9	6	71.9	18
F1F2F5F6	96.0	8	96.0	24
F1F3F5F6	102.4	3	102.4	6
Total	70.6	100	76.2	100

[a]BPP

Table 28.4 Enterprise application functions available for executing the travel management process

Activity	t	τ	e	ε	r	ρ	Activity description
STMP	2	0	0	0	0	0	Start travel management process
CPR	2	0	0.2	0	0	0	Copy previous request
CNR	10	0	0.05	0	0	0	Create new request
ITD	5	0	0.5	0	0	0.2	Input travel data
ITFD	10	0	0.05	0.3	0	0.2	Input travel financing data
LPR	5	0	0	0.1	0	0	Look up previous requests
CSCH	2	0	0	0.1	1	0	Check schedule
CFIN	5	0	0	0.15	1	0	Check financing
ADOC	5	5	0	0.1	0	0.5	Attach documents
SREQ	1	60	0	0	0	0	Submit request
GAPP	1	0	0	0	0	0	Get approval

4 Illustrative Sample

An illustrative example is used to demonstrate possibilities for practical application of adaptive EA. In this example, users use an enterprise application to execute the travel management processes. The main activities of the travel management processes are (1) preparation of the expense request, (2) getting a travel approval, and (3) reporting the travel expenses. Table 28.4 lists functions provided by the enterprise application to execute this business process. One can observe that activity ITD is error prone, and the BPP (Fig. 28.4) offers a number of functions, which can be used to identify errors. For instance, the user can request the CSCH function to check overlapping between proposed travel dates and other events; invoking this function reduces the erroneousness level, and after completing this function, the user always returns to the previous function.

Table 28.5 shows process variants obtained for different levels of γ_k(the adaptation process is executed 200 times). In the case of $\gamma_k = 0$, a user always follows the standard process. In the case of $\gamma_k = 0.5$, adaptation is performed and that allows improving the process execution score. The most efficient process variant is not the most frequently observed because only local adaptation is performed and there is a random activity selection innovation. Still, different variants obtained as the

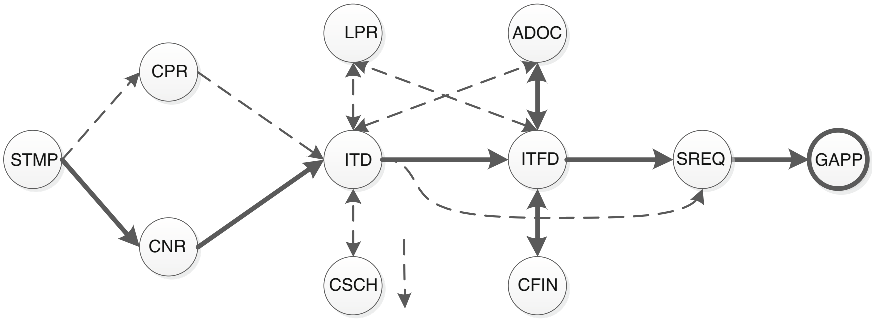


Fig. 28.4 Alternative process execution variants for the travel management application

Table 28.5 Adaptation results according to $\gamma_k, w_4 = 10$

Process variant	Average S	Count
$\gamma_k = 0$	41	200
STMP, CNR, ITD, ITFD, CFIN, ITFD, ADOC, ITFD, SREQ, GAPP*	41	200
$\gamma_k = 0.5$	29	113
STMP, CPR, ITD, ITFD, SREQ, GAPP	20	15
STMP, CPR, ITD, ITFD, CFIN, ITFD, SREQ, GAPP	23	14
STMP, CNR, ITD, ITFD, SREQ, GAPP	26	36
STMP, CNR, ITD, ITFD, CFIN, ITFD, SREQ, GAPP	30	19
STMP, CNR, ITD, ITFD, CFIN, ITFD, ADOC, ITFD, SREQ, GAPP*	38	29
$\gamma_k = 1$	23	142
STMP, CPR, ITD, ITFD, SREQ, GAPP	20	6
STMP, CPR, ITD, SREQ, GAPP	21	88
STMP, CPR, ITD, ITFD, CFIN, ITFD, SREQ, GAPP	24	8
STMP, CPR, ITD, LPR, ITFD, SREQ, GAPP	24	4
STMP, CPR, ITD, LPR, ITFD, CFIN, ITFD, SREQ, GAPP	27	4
STMP, CNR, ITD, SREQ, GAPP	27	28
STMP, CNR, ITD, CSCH, ITD, SREQ, GAPP	34	4

adaptation result give better process execution score than the standard process execution score.

Figure 28.5 shows impact of changing process characteristics on the adaptation efficiency. Increasing values of w_4 can be interpreted as switching from the focus on execution speed to the focus on execution reliability (i.e., avoiding the process failure). As noted above, the standard process emphasizes reliability. However, if the process score is mainly determined by speed, the standard process is not the most efficient one, and better process execution variants with fewer activities can be found. If importance of reliability increases, the best adapted process execution variant still yields better results than the standard process. However, the average score is worse than for the standard process, and adaptation yields many sub-efficient process execution variants.

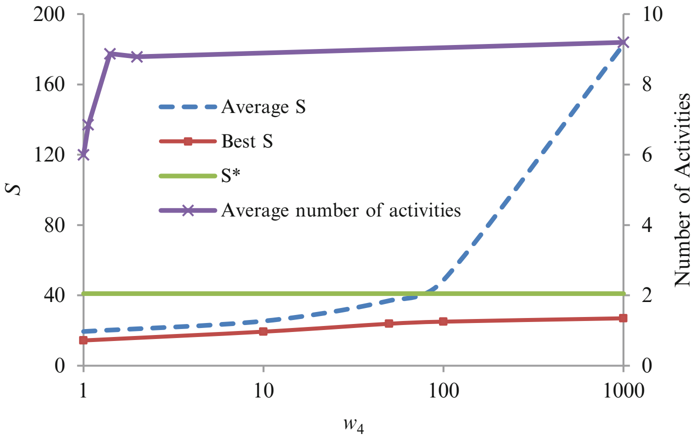


Fig. 28.5 Trade-off between process execution speed and safety

These results show that the choice of the most efficient process execution variant depends upon the process characteristics.

5 Conclusion

The model for simulating adaptive EA has been elaborated. The experimental results confirm the hypothesis about efficiency of adaptive processes and dependence of the preferred process execution variant upon the process characteristics. This model can be further used to evaluate adaptation strategies and alternative solutions for improving process execution productivity.

Dependence of the preferred process execution variant upon the process characteristics is of particular interest because it is rarely accounted for in implementation of EA (main attention is devoted to fine-tuning technical performance parameters rather than process execution parameters). That is especially important because this dependence is temporal, and the most efficient process execution variant changes from time to time. Therefore, the process configuration should be regularly updated according to application monitoring results. The obtained experimental results show that adaptation is most important for high-frequency processes, while it is a risky approach in the case reliability-conscious applications.

The elaborated model has a number of limitations. In the spirit of simulation of complex systems, it is intentionally kept as simple as possible. The model has limited consideration of user preferences because each user or user group might come up with her/his own optimal process execution variant and each user might have different levels of inventiveness. The model does not address an issue of updating the BPP (i.e., under what circumstances the adapted process execution variant would become a new BPP). Another important limitation is that productivity often is not the most important adaptation criterion. Achieving the business goals is more important and leads to more significant gains. However, formal representation and measuring at the level of individual workflows of that is challenging.

The model will be expanded with emphasis on usability aspects and evaluation of different productivity solutions (i.e., what kind of tools needs to be provided to the users to further productivity improvement).

References

Anupindi R, Chopra S, Deshmukh SD, Van Mieghem JA, Zemel E (2006) Managing business process flows principles of operations management. Pearson Prentice Hall, Upper Saddle River

Ardissono L, Goy A (2000) Tailoring the interaction with users in web stores. User Model User-Adap Interact 10:251–303

Barclay PJ, Griffiths T, McKirdy J, Kennedy J, Cooper R, Paton NW, Gray P (2003) Teallach – a flexible user-interface development environment for object database applications. J Vis Lang Comput 14:47–77

Curran TA, Ladd A (2000) SAP R/3 business blueprint: understanding enterprise supply chain. Prentice Hall PTR, Upper Saddle River

Keystone (2007) ERP end-user business productivity: a field study of SAP & Microsoft, Keystone strategy. http://download.microsoft.com/download/Keystone-RPBusinessProductivityStudy.pdf. Accessed 2 May 2010

Reichert M, Rinderle-Ma S, Dadam P (2009) Flexibility in process-aware information systems. In: Jensen K, van der Aalst W (eds) LNCS, vol 5460. Springer, Berlin/New York, pp 115–135

Singh A, Wesson J (2009a) Improving the usability of ERP systems through the application of adaptive user interfaces. ICEIS 2009 – 11th international conference on enterprise information systems, proceedings, SAIC, pp 208–214

Singh A, Wesson J (2009b) Evaluation criteria for assessing the usability of ERP systems. SAICSIT'09, Riverside, Vanderbijlpark, South Africa, 12–14 Oct 2009, pp 87–95

Topi H, Lucas W, Babaian T (2005) Identifying usability issues with an ERP implementation (2005). In: ICEIS 2005 – proceedings of the 7th international conference on enterprise information systems, pp 128–133

Zeng J, Sun H, Liu X, Deng T, Zou J (2010) PRV: an approach to process model refactoring in evolving process-aware information systems, In: Proceedings – 2010 I.E. 7th international conference on services computing, SCC 2010, pp 441–448

Chapter 29
Analyzing Empirical Data in Requirements Engineering Techniques

Li Jiang, Armin Eberlein, and Aneesh Krishna

1 Introduction

The development of large- and medium-sized software systems usually involves complex processes that make use of several development techniques. Since the term "software engineering (SE)" was first coined in 1968 at the first SE conference (Naur et al. 1969), numerous SE techniques have been proposed. However, early experience has shown that there is no silver bullet to deal with software engineering problems (Brooks 1987). Most software development processes employ a combination of techniques (Jiang et al. 2008). Furthermore, Glass (2004) and other researchers (Jiang et al. 2008; Glass 2004; Emam and Birk 2000; Zowghi et al. 2001; Neill and Laplante 2003; Antón 2003) stressed that having a systematic method to help selecting SE techniques is very significant for software industry as the body of SE technique knowledge increases rapidly, and it is far beyond an individual engineer's capability to understand all of them, let alone to be able to process them in a meaningful way and select them in a complex engineering environment. However, it is not trivial to assess the suitability of an SE technique within the context of a software project as many techniques are available and numerous factors influence decision making. We therefore started a research project focusing on the analysis of the suitability of requirements engineering (RE)

L. Jiang (✉)
School of Computer Science, The University of Adelaide, Adelaide, SA, Australia
e-mail: li.jiang@adelaide.edu.au

A. Eberlein
Computer Engineering Department, American University of Sharjah, Sharjah, UAE
e-mail: eberlein@ucalgary.ca

A. Krishna
Curtin University of Technology, Perth, WA 6102, Australia
e-mail: A.Krishna@curtin.edu.au

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_29,

Table 29.1 clustering result with the FCM algorithm

Setting	NOC = 4	NOC = 5	NOC = 6	NOC = 7	NOC = 8	NOC = 9	NOC = 10	NOC = 11	NOC = 12
Value of the cost function	27.12	25.43	21.78	18.81	17.55	17.92	18.12	19.55	19.38

Note: (1) NOC is the abbreviation of number of clusters. (2) Weights for all attributes are set to 1

techniques (a subset of SE techniques)[1] for a software project based on its characteristics (Jiang 2005). This resulted in methodologies[2] and a framework that helps select the most suitable RE techniques for a software project. In our previous research, we developed 31 attributes that help characterize RE techniques and analyze 46 RE techniques in depth (Brooks 1987; Jiang 2005). The attributes are classified into two categories: Category 1 includes 13 essential attributes that are generally applicable to all RE techniques. The attributes in category 2 are supplementary to those in category 1. We developed a matrix containing the information about the suitability of each technique based on published research of RE techniques. We also conducted a survey among eight RE experts from both industry and academia to validate our findings presented in the technique ability matrix (Jiang 2005). We analyzed RE techniques using the Fuzzy C-means (FCM) method (Dunn 1974; Bezdek 1974, 1981). Partial clustering results are given in Table 29.1 which includes the number of clusters and the values of the cost function obtained in the research (see Jiang and Eberlein (2006) for more information about the research result).

Our initial data analysis of RE techniques using FCM provided information about the similarity and differences of RE techniques. However, the detailed analysis of the clustering result revealed that a number of issues still remain unanswered. For instance, we do not know how good the clustering result is after we use FCM, and the fitness of some techniques to a cluster is questionable based on in-depth analysis of the techniques within a cluster. Additionally, finding the right number of clusters is a tedious process, as the clustering process has to be repeated many times. The difficulty to find the optimum number of clusters of techniques prevents us from providing dynamic support for techniques analysis and selection. Thus, one of the challenging questions here is how to find effective methods to analyze the data of RE techniques and using the best method to help cluster and analyze RE techniques. To tackle the problem, we have systematically investigated the existing research on data analysis and mining on software engineering data. Even though there is a plethora of data mining and clustering methods available, our investigation has

[1] As RE techniques are a subset of SE techniques, we infer that the research results derived from RE techniques analysis will be applicable to SE techniques analysis.

[2] We acknowledge the differences between the two terms “method” and “technique” as used in the SE research community and the disparities of the definitions given for these two terms in academia. The term “method” is deliberately used in this chapter to refer to any one or more algorithms and/or methods created for data clustering and data analysis. The purpose of adopting this terminology (in this chapter only) is to differentiate the two terms “method” and “technique” with the latter referring to SE techniques or methods.

shown that the existing research on clustering SE data does not provide comparative or empirical analysis information or heuristics on the performance of the clustering methods, and which clustering methods can help generate the optimum number of cluster automatically. To deal with this problem, we have explored and used several methods to help cluster and analyze RE techniques in this research: clustering based on statistical tests (Gao et al. 2002), genetic algorithm (Zhao et al. 1996), dimension reduction (Jolliffe 1986) (principal component analysis (PCA)) in combination with FCM, and PCA in combination with genetic algorithm method. The objectives of this research is to understand if the existing clustering methods or data analysis methods can be used to offer meaningful help for data clustering and analysis. In order to generate meaningful conclusions, we still use our 47 RE techniques data set (Brooks 1987; Jiang 2005) intentionally as the data set about the techniques are readily available, and we are able to compare and evaluate the clustering results generated by these methods. This chapter reports the findings and experiences obtained in this research.

The rest of the chapter is organized as follows: Sect. 2 discusses related research; Sect. 3 presents a clustering method that is based on statistical tests. Section 4 summarizes our experiences of using a genetic algorithm to cluster RE techniques. The clustering of RE techniques by using the combination of PCA and FCM, and PCA with GAFC is illustrated in Sect. 5. A comparison and evaluation of the clustering results generated by these methods is presented in Sect. 6. Section 7 presents our conclusions and future research.

2 Related Research

The work related to analyzing software engineering data can be traced back to 1950s when information about the "lines of the codes" was analyzed (Jones 2008). However, the latest research in formal classification and analysis of the software engineering data can be accredited to the seminal work done by Khoshgoftaar and Allen (1999) and Mendonca and Sunderhaft (1999). Khoshgoftaar and Allen use classification and regression trees (CART) algorithm to help model various software quality attributes. Mendonca and Sunderhaft surveyed existing approaches that can be used in mining software engineering data. Since then, a lot of research has been done on mining software engineering data using various data mining methods and data analysis methods to analyze SE data. For example, Zhong et al. investigated *k*-means and neural gas clustering algorithms to conduct clustering-based analysis for software quality-estimation problems (Zhong et al. 2004). Jiang et al. (2006) investigated multiple centroid-based unsupervised clustering algorithms for network intrusion detection. Dickinson et al. (2001) used a clustering approach to analyze the execution profile to help find failure data. The most related research is the research done by Baraldi and Blonda in 1999 where five fuzzy clustering algorithms for pattern reorganization are discussed and compared. Even though many clustering methods have been used in analyzing SE data during the last 10 years, most research focuses on designing or using a set of clustering algorithms. The advantage and disadvantage of these algorithms and how to use the clustering algorithms to analyze heterogeneous data sets have not been explicitly researched or discussed. It appears that limited research has been done to systematically examine the merit of a clustering method utilized in a specific application where the number of attributes is large and the data is fuzzy in nature. This research tries to explore merits and issues of clustering methods applied to a specific application.

3 Clustering Based on Statistical Test

As has been discussed in Sect. 1, one of the major problems with FCM is that the optimum number of clusters cannot be determined before the actual clustering begins. It has to be determined by trial and error using repeated clustering. To solve this problem, a statistical test-based clustering (STC) method was proposed by Gao et al. (2002). The idea of the approach is to conduct statistical tests in each cluster formed in the trial so that a unimodal distribution is achieved in all the trial clusters. According to the experiment results of applying the algorithm developed by Gao et al. (2002), M, the number of origins randomly selected, needs to exceed 10 to ensure fast convergence and generality of the algorithm. However, there are only 46 techniques (data points) in the current data set, which means that we get less than 5 ($46/10 = 4.6$) data points in each cluster. Apparently, 5 data points in each cluster are not enough to generate a normal distribution. To apply the algorithm, we need to extend our original data set. According to Lehmann and Casella (1998), it is possible to construct such an extended data set $\boldsymbol{X}$ which is completely equivalent to the original data set X' statistically. Thus, we have tried to inject 184 new data points into the sampled space which increases the total data points to 230. These new data points are called "unlabeled data points" while the original data points are called "labeled data points" which allows us to differentiate them from the inserted ones. The unlabelled data points are inserted in a semi-random manner: the data is generated randomly with the same type and within the same range of the original data set X'. The inserted data can be considered as a kind of perturbed data. Unlike the initial objective of using the perturbed data to protect confidentiality of data (Liu et al. 2006), the objective of the inserted data in this research is to find data patterns or structures within the original data set. According to Burridge (2003), the property of sufficiency of the perturbed data set with respect to the given statistical model must be the same as that of the original data set. Within the given data and the generated perturbed data in the research, it is possible to infer that the extended data set $\boldsymbol{X}$ and the original data set X' have the same sufficient statistic[3] (Hogg and

[3] A sufficient statistic refers to a statistic that has the property of sufficiency with respect to a statistical model and its associated unknown parameter θ that are used in statistical calculation and reasoning (Hogg and Craig 1978), i.e., no other statistic that can be calculated from the same data set provides any additional information as to the value of the parameter θ.

1. Set initial values used in the algorithm, including:
   1) Generating the dataset X including the inserted random data set (unlabelled) based on the original dataset X'
   2) Setting M sampling origins represented with vector C randomly in the data space $X = \{x_i, .., x_n\}$.
   3) where $C = \sum_{i=1}^{n} x_i$, n is the number of elements in X.
   4) Setting cn=2 where cn is the initial number of clusters.
   5) Calculating the distance D_i between each sample x_i and vector C, and the distance U_i between x_i and x_{i+1} where x_i and x_{i+1} are perpendicular.
   6) k is the number of neighbours of x_i that are going to search; while p represent the number of clusters
   7) Selecting $a = 0.05$ and it is easy to find $T(a)$=1.64485 from the statistic table.
   8) Setting s=0;
2. For each cluster, compute the following normalized statistic T_K where K is the number of statistical test, and p is the dimension of the data set x_i:

$$T_K = \left(\frac{1}{M}\sum_{i=1}^{M}\frac{D_i^p(k)}{D_i^p(k)+\frac{1}{2}U_i^p(k)} - \frac{1}{2}\right)\times\sqrt{12M}$$

3. If $T_K \geq T(a)$, Then s=s+1
4. Repeat Step 2 and 3 N times (N>=100),

   calculate the size of test $\bar{s} = s/N$

   If $\bar{s} \geq a$,
   Then data set X is declared as multimodal distribution and is separable.
   Else X is declared as unimodal distribution.
5. For all i,

   If data sets $X_i \subset \mathbf{X}$ is unimodal,
   Then stop, and cn is the number of clusters that is looking for
   Else cn=cn+1, and go to 2

Fig. 29.1 Algorithm for computing the number of clusters

Craig 1978). This is because the inserted data have the same types, range, and distribution as the original. Thus, we can conclude that the result of the statistical analysis of the original data set is the same as the statistical analysis of the extended data set obtained as described above. By using the perturbed data method, we are able to use the modified statistical test-based clustering method and obtain the proper clustering numbers of the original data set of RE techniques.

The modified algorithm is shown in Fig. 29.1. The algorithm was implemented using C++ and the Matlab™ software package. By using the algorithm and extended data set approach, we found that each cluster reached its single peak distribution when NL (number of clusters) = 8. The obtained result NL is similar with the values we got in Jiang and Eberlein (2006). Our initial analysis with the STC algorithm is promising as the algorithm has a major advantage over FCM in that it provides us with a way to find the number of clusters automatically. Even though the number of techniques correctly clustered is slightly bigger than using pure FCM approaches as demonstrated in Table 29.5 (the implication of this will be discussed in Sect. 6), we found that some techniques have not been clustered in the right clusters according to the human researchers' judgment. Thus, we want to explore if genetic algorithm can provide better solution. The exploration results are given in next section.

4 Using Genetic Algorithm in Fuzzy Clustering

Based on the mechanism of natural selection and genetics, genetic algorithms (GAs) have been designed and widely used in many optimization problems (Gen and Cheng 1997; Chambers 2001; Cordon 2001). In this research, we want to explore if GAs can be used to help generate better clustering results or provide more information about RE techniques data. After comparing different GAs used in clustering methods, the GA for the Fuzzy Clustering (GAFC) algorithm proposed by Zhao et al. in (1996) was used to help cluster the RE techniques in this research as the computational complexity of this algorithm is not very high and it can reach better convergence than other algorithms. According to Lee and Takagi, the number of generations and mutation probability can be set within the range of 10–160 and 0.0001–1.0, respectively (Lee et al. 1993). In our experiments, we set the number of generations to 16, and the mutation probability to 0.10. C++ and several Matlab™ packages were used during the implementation of the GAFC algorithm. This algorithm proved to be very expensive when all 31 attributes (dimensions) are used during clustering. Running on a PC with a 1.8GHZ CPU, the algorithm took almost 4 days to get clustering results when the number of clusters is set to 3, and 7 days when the number of clustering is set to 4. To improve the efficiency of the algorithm, we decided to reduce the number of attributes used in the clustering. Instead of using all 31 attributes, 13 attributes in category 1 were selected for the clustering. With the reduced number of attributes, the number of clusters and the values of the cost function calculated using GAFC algorithm and FCM algorithm are shown in Table 29.2. As can be seen from the table, the GAFC algorithm converged quickly in the calculation of the cost function before the number of cluster setting exceeds 6 compared to FCM. However, it turns out that the performance of GAFC algorithm is worse than FCM when the number of clusters is greater than 6. Moreover, it can hardly reach the reasonable decision with GAFC about the exactly number of clusters that should be used in analyzing RE techniques as the values of cost function of the algorithm decrease continuously even after the number of cluster passes 9. Our initial investigation showed that two major reasons for this phenomenon are likely (1) the limitation of the algorithm itself and (2) the

Table 29.2 GA clustering result compared with FCM clustering results

Number of clusters	Cost value by using GA-fuzzy clustering algorithm	Cost value by using fuzzy C-means clustering algorithm
2	12.76	18.89
3	9.32	16.23
4	7.86	12.96
5	6.72	10.65
6	5.91	6.18
7	5.01	5.11
8	4.05	3.60
9	3.29	3.64

characteristics of the date points that include 13 attributes, i.e., the algorithm might not be suitable for data with more than 13 dimensions.

Our further investigation on six attributes, randomly selected from category 1 data set, shows that the value of the cost function with GAFC algorithm also converges fast and reaches the minimum value when the number of clusters reaches 12. The generated number of clusters reaches 12 which are different from the number of clustering (number 8 or 9) generated with the FCM algorithm in our early research. The major reason of the difference is that the essential information of the RE techniques data is lost since the number of attributes (each attribute can be considered as one dimension of a technique) used in the cluster is randomly removed. Thus, we conclude that the performance of GAFCs is the best when it applies to the data with fewer dimensions. As the result of this observation, the immediate question is how to reduce the dimensions of the data points while keeping the essential information of the data points so that the full potential of GAs can be achieved. To tackle this issue, we utilize dimension reduction methods which will be discussed in the next section.

5 Clustering Based on Dimension Reduction

As has been discussed above, one of the major challenges of conducting effective data analysis is that the data points contain too many attributes (dimensions). Many data analysis and clustering algorithms cannot deal with multidimensional data effectively. Thus, one of the solutions to tackle this problem is to reduce the dimensions of the data points while keep the essential information of the original data. Dimension reduction methods have been used widely in computer vision and pattern recognition research and have proved effective in analyzing data that contains many dimensions (Carreira-Perpinan 1997; Jolliffe 1986; Jones 1983; Hastie and Stuetzle 1989). The major objective of dimension reduction is to search for a property-preserved low-dimensional representation of the higher dimensional data, i.e., to map the high dimensional space to a lower dimensional space in such a way that the required properties are preserved. Application of dimensional reduction techniques to SE data makes sense as SE data usually have high dimensionality (Goel and Shin 1997).

There are many dimension reduction methods, such as principal component analysis (Jolliffe 1986), projection pursuit (Jones 1983), and principal curves (Hastie and Stuetzle 1989). In our research, we use the principal component analysis (PCA) method as it is the most widely used method in practice and suitable to the size and type of data that we want to process. Moreover, PCA has already been implemented in the Matlab™ software package *princomp*. The fundamental idea of PCA is to project the data with high dimensions along the dimensions with maximal variances so that the reconstruction error of low-dimension data points can be minimized and properties of the data points can be maximally preserved. In this research, we also used a modified algorithm of PCA in which the weight $W_j \in [0, 1]$,

Table 29.3 An example of generated data set after dimension reduction

Techniques	D_1	D_2	D_3	D_4	D_5	D_6
Brainstorming	−1.892500	0.079340	−0.206770	0.030650	−0.419870	−0.069110
Designer as apprentice	−1.630000	0.103430	−0.414710	0.304470	−0.663370	−0.106870
Document mining	−1.168700	0.379490	−0.075140	−0.112600	−0.558350	0.117600
Ethnography	−1.914800	−0.322170	−0.503340	0.045320	−0.617720	0.004040
Focus group	−1.672900	−0.824850	−0.532970	0.094020	0.177100	−0.117170
Interview	−1.915200	−0.281250	−0.304210	0.018030	−0.334170	0.214430
Contextual inquiry	−1.658000	−0.170510	−0.246680	0.195570	−0.345710	0.303790
Laddering	−0.779230	−0.704200	0.290550	−0.309960	−0.197310	0.635260

Table 29.4 Clustering result

Number of clusters	Value of cost function (a = 1.5)
NL = 2	0.049800
NL = 3	−0.004067
NL = 4	−0.011247
NL = 5	−0.009762
NL = 6	−0.001504
NL = 7	−0.000941
NL = 8	−0.013643
NL = 9	−0.013643
NL = 10	−0.009591

Note: a is fuzzification coefficient

$j = 1, m$, is applied to each attribute based on their importance with respect to the selection of RE technique as judged by requirements engineers. We have used *princomp* in Matlab™ in our experiment in this research. By using the algorithms, we have reduced the 31-dimensional data points (see Table 29.3) to six-dimensional data points. Some examples of the six-dimensional data points with *princomp* are presented in Table 29.3 (see Jiang et al. (2011) for all the detailed results of all data set).

The data generated in the dimension reduction operation was clustered using the FCM algorithm. The results of the clustering are shown in Table 29.4. In the table, a is fuzzification coefficient; normally, a is set to 1.5–2. We set a to 1.5, and we compare the results before and after using dimension reduction techniques. As can be seen, there are two choices for selecting the number of clusters: 8 and 9. The obtained number of clusters is essentially the same with the number of the clusters that we used in our previous research where only FCM was used. The major gain of using PCA is that it helps to reduce the complexity of using FCM in the later stage of clustering. The major problem with the PCA and FCM is that it is still hard to tell the exact number of clusters that shall be chosen, i.e., this number has to be determined by humans as can be seen from Table 29.4; both NL = 8 or NL = 9

Table 29.5 The number of correctly and incorrectly clustered techniques

No.	Clustering methods	Number of correctly clustered techniques	Number of incorrectly clustered techniques
1	FCM	36	8
2	Statistical tests	37	8
3	Genetic algorithm	35	9
4	PCA with FCM	40	5
5	PCA with genetic algorithm	39	5

are valid options. The data generated with PCA was also clustered by using the GAFC algorithm; the number of generated clusters is NL = 7. The performance of using this combination approach (PCA with genetic algorithm) is shown in Table 29.5 as well. The discussion of the implication of the data set is presented in the next section.

6 The Comparison and Evaluation of the Clustering Results

We have analyzed and evaluated the clustering results generated from using clustering methods reported in the previous sections. None of the used clustering methods gives the identical clustering results with our human researchers. The reasons for this require further research. In Table 29.5, the correctly clustered techniques refer to the techniques that are clustered into the same cluster consistently with human researchers. The incorrectly clustered techniques refer to the techniques that are clustered into different clusters by human researchers. The numbers of correctly and incorrectly clustered techniques are summed for each method and presented in Table 29.5. Overall, our observation is:

1. There is a subtle difference of the results between using each single method as can be seen from Table 29.5 (methods 1, 2, 3). It is hard to produce any conclusion on the differences based on current research which is subject to further research. The only advantage of the statistical test method is that the method needs less human intervention than the other two methods.
2. Combined approaches are slightly better than using a single approach. As can be seen from Table 29.5, the performance of clustering using the combination of PCA with FCM or PCA with GAFC is slightly better than using the rest of each single method. If this observation can be generalized, it is part of our future research.

To address the concerns of likely errors, we have used the MRQAP method developed by Dekker et al. (2007) to test if there is randomness or correlation between the results generated in our research. MRQAP is developed based on Krackardt's work on quadratic assignment procedure (QAP) (Krackardt 1987) which assesses the association between data sets through a statistical analysis and

Table 29.6 The differences between the numbers of correctly clustered techniques generated by two methods

	M1	M2	M3	M4	M5
M1	0	1	1	4	3
M2	1	0	2	3	2
M3	1	2	0	5	4
M4	4	3	5	0	1
M5	3	2	4	1	0

Table 29.7 The differences between the number of inconsistent techniques generated by two methods

	M1	M2	M3	M4	M5
M1	0	0	1	3	3
M2	0	0	1	3	3
M3	1	1	0	4	4
M4	3	3	4	0	0
M5	3	3	4	0	0

hypothesis test. To use MRQAP, two matrixes were created and are shown in Tables 29.6 and 29.7. These two tables contain the differences of the numbers of correctly and incorrectly clustered techniques generated by two methods listed in Table 29.5. To illustrate this, let us use $\{a_{i,j}\}$, $\{b_{i,j}\}$, and$\{c_{i,j}\}$ to represent the three matrixes in Tables 29.5, 29.6 and 29.7, respectively. M1 to M5 in Tables 29.6 and 29.7 represents the methods 1–5 in the second column of Table 29.5. The value of the cells in Tables 29.6 and 29.7 is generated using the values of two columns in Table 29.5, i.e., $\{b_{i,j} = |a_{i,1} - a_{j,1}|\}$, $\{c_{i,j} = |a_{i,2} - a_{j,2}|\}$. With common setting of the parameters, number of permutation (simulation) = 500, and random seed = 0, the MRQAP algorithm was run with the two matrices. The correlation coefficient is 0.8929 with a p value 0.021 which is below 0.05 threshold. This gives us good reason to reject the null hypothesis that there is no correlation between the number of correctly clustered techniques and the number of incorrectly clustered techniques. The implication of this is that the numbers obtained in Table 29.5 are free from randomness.

Additionally, the problems related to the different clustering results between using the algorithm and human researchers also indicate that some intervention by human researchers is necessary at this stage for RE techniques clustering.

7 Conclusion and Future Work

Analysis of empirical data is a difficult yet important task in SE (Shin and Goel 2000). This chapter reported our experiments with three data analysis methods and algorithms: clustering based on statistical tests method, genetic algorithm method, and the clustering by using dimension reduction method. We presented the issues and demonstrated possible ways to help deal with the determination of the cluster number and reduction of dimensions for effective clustering. The research has shown that the best solution to analyzing empirical data is to combine different data analysis methods. This combined approach might be a time-consuming and daunting process; however, it is the only way to help discover meaningful information and any underlying structures in data. At this stage, it is safe to say that the STC algorithm is promising as the algorithm can provide a way to find the exact number of clusters automatically. Additionally, a combination of PCA and FCM, and PCA and GAFC can provide better data analysis results based on the research results obtained in this research. Finally, it is still difficult to say whether the combination of PCA, GA, and FCM will always lead to better results for clustering analysis which is the question that is subject to our future research. To design and develop a tool which can facilitate using different data analysis methods to analyze SE data is another topic for our future research.

References

Antón AI (2003) Successful software projects need requirements planning. IEEE Softw 20 (3):44–46

Baraldi A, Blonda P (1999) A survey of fuzzy clustering algorithms for pattern recognition – Part I. IEEE Trans Syst Man Cybern Part B Cybern 29(6):778–785

Bezdek JC (1974) Cluster validity with fuzzy sets. J Cybern 3(3):58–71

Bezdek JC (1981) Pattern recognition with fuzzy objective function algorithms. Plenum Press, New York

Brooks F (1987) No silver bullet-essence and accident in software engineering. IEEE Comput 20 (4):10–19

Burridge J (2003) Information preserving statistical obfuscation. Statist Comput 13(4):321–327

Carreira-Perpinan MA (1997) A review of dimension reduction techniques. Technical report CS-96-09, Department of Computer Science, University of Sheffield

Chambers LD (2001) The practical handbook of genetic algorithms applications. Chapman & Hall/CRC, Boca Raton

Cordon O (2001) Ten years of genetic fuzzy systems: current framework and new trends. In: Proceedings joint 9th IFSA world congress and 20th NAFIPS international conference (Cat. No. 01TH8569), p 1241. 0-7803-7078-3, 978-0-7803-7078-4

Dekker D, Krackhardt D et al (2007) Sensitivity of MRQAP tests to collinearity and autocorrelation conditions. Psychometrika 72(4):563

Dickinson W, Leon D, Podgurski A (2001) Finding failures by cluster analysis of execution profiles. In: Proceedings of the international conference on software engineering (ICSE), Toronto, ON, Canada, pp 339–348

Dunn J (1974) A fuzzy relative of the ISODATA process and its use in detecting compact, well separated cluster. J Cybern 3(3):32–57

Emam KE, Birk A (2000) Validating the ISO/IEC 15504 measure of software requirements analysis process capability. IEEE Trans Softw Eng 26(6):119–149

Gao XB, Ji HB, Li J (2002) An advanced cluster analysis method based on statistical test. IEEE ICSP, pp 1100–1103

Gen M, Cheng R (1997) Genetic algorithms and engineering design. Wiley, New York

Glass RL (2004) Matching methodology to problem domain. Commun ACM 47(5):19–21

Goel AL, Shin M (1997) Software engineering data analysis techniques (tutorial). In: Proceedings of the 19th international conference on software engineering, Boston, Massachusetts, United States, pp 667–668

Hastie TJ, Stuetzle W (1989) Principal curves. J Am Stat Assoc 84:502–516

Hogg RV, Craig AT (1978) Introduction to mathematical statistics. Macmillan, New York

Jiang L (2005) A framework for requirements engineering process development. Ph.D. thesis, University of Calgary, Canada

Jiang L, Eberlein A (2006) Clustering requirements engineering techniques, In: The 10th IASTED international conference on software engineering and applications, Dallas, TX, USA, 13–15 November

Jiang SY, Song XY, Wang H et al (2006) A clustering-based method for unsupervised intrusion detections. Pattern Recog Lett 27(7):802–810

Jiang L, Eberlein A, Far BH, Mousavi M (2008) A methodology for the selection of requirements engineering techniques. J Softw Syst Model 7(3):303–328

Jiang L, Eberlein A, Krishna A (2011) Analyzing empirical data in software engineering techniques. Technical report (1), Jan 2011. School of Computer Science, The University of Adelaide, Australia. http://cs.adelaide.edu.au/~ljiang/research/publicationsList/TechicalReport_REtechniquesClustering.pdf

Jolliffe IT (1986) Principal component analysis, Springer series in statistics. Springer, Berlin

Jones MC (1983) The projection pursuit algorithm for exploratory data analysis. Ph.D. thesis, University of Bath

Jones C (2008) Applied software measurement: global analysis of productivity and quality, 3rd edn. McGraw-Hill, New York

Khoshgoftaar TM, Allen EB (1999) Modeling software quality with classification trees. In: Pham H (ed) Recent advances in reliability and quality engineering. World Scientific, Singapore

Krackardt D (1987) QAP partialling as a test of spuriousness* 1. Soc Netw 9(2):171

Lee MA, Takagi H (1993) Dynamic control of genetic algorithms using fuzzy logic techniques. In: Proceedings of international conference on genetic algorithms, Urbana-Champaign, IL, July 1993, pp 76–83

Lehmann EL, Casella G (1998) Theory of point estimation. Springer, New York

Liu K, Kargupta H, Ryan J (2006) Random projection-based multiplicative data perturbation for privacy preserving distributed data mining. IEEE Trans Knowl Data Eng 18(1):92–106

Mendonca M, Sunderhaft NL (1999) Mining software engineering, data: a survey. A DACS state-of-the-art report, Data & Analysis Center for Software, Rome, NY

Naur P, Randell B et al (1969) Software engineering: report on a conference sponsored by the NATO SCIENCE COMMITTEE, Garmisch, Germany, 7–11 Oct 1968, Scientific Affairs Division, NATO

Neill CJ, Laplante PA (2003) Requirements engineering: the state of the practice. IEEE Softw 20 (6):40–45

Shin M, Goel AL (2000) Empirical data modeling in software engineering using radial basis functions. IEEE Trans Softw Eng (0098-5589) 26(6):567

Zhao L, Tsujimura Y, Gen M (1996) Genetic algorithm for fuzzy clustering. In: Proceedings of IEEE international conference on evolutionary computation, p 716. 0-7803-2902-3, 978-0-7803-2902-7

Zhong S, Khoshgoftaar TM, Seliya N (2004) Analyzing software measurement data with clustering techniques. IEEE Intell Syst 19(2):20–27

Zowghi D, Damian D, Offen R (2001) Field studies of requirements engineering in a multi-site software development organization. In: Proceedings of the Australian workshop on requirements engineering, University of New South Wales

Chapter 30
Conceptual Modelling: A Philosophy of Fiction Account

Balbir S. Barn and Gulzaar K. Barn

1 Introduction

Business analysts, software engineers and indeed anybody concerned in any way with the software development process refer to models in documents and conversations with colleagues. Often the nature of the conversation will be to make assertions about the model which they deem capable of being true or false. For example, it is not uncommon for an analyst to refer to a symbol on a diagram and state, 'this is external to the system' or even refer to another symbol and say, 'this is the system'. Such statements do not appear to cause the analyst any pause for thought, and more often than not, this shared linguistic practice does not present problems and will likely be the basis of a successful conversation.

In this way, models or, as we will refer to them, conceptual models have become the lingua franca of choice amongst IT practitioners and that is reflected in the centrality of models in modern information systems (IS)practice.

For the pursposes of this paper, we treat models as representations of a (software) system at an abstract level that has been developed using some form of modelling language (Booch et al. 2005). Models may take the form of "diagrams" of differing types representing different facets of a target system. In IS practice, the Unified Modelling Language (UML) is a prevailing standard and the current version (UML 2.0) has 13 diagram types and is used extensively to produce IS models.

B.S. Barn (✉)
Middlesex University, Hendon, London NW4 4BT, UK
e-mail: b.barn@mdx.ac.uk

G.K. Barn
University of Manchester, Oxford Road, Manchester M13 9PL, UK
e-mail: gulzaar.barn@gmail.com

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_30,

Amongst the many reasons for the ubiquity of the production and use of models is the capstone nature of modelling in many software engineering approaches where it is used as an approach to user requirements definition and as a basis for developing information systems to meet those requirements (Moody 2005). More generally, conceptual modelling may be used to support acquisition, standardisation and integration of information systems. Such uses may happen at multiple different levels:

- Application level: models are used to define requirements for specific applications.
- Enterprise level: models are used to define requirements for an entire organisation and provide the basis for data management across the enterprise.
- Industry level/sector level: a variant of conceptual models – reference models are used to define information and behaviour requirements for a sector as the basis of standardisation and development of generic software solutions.

The centrality of conceptual modelling to systems design has led many notations and methods with varying levels of use and efficacy. But one abiding concern remains: How good is a model? The quality of conceptual models has the potential to make a significant impact on other IT artefacts. For example, inadequacy of conceptual models may lead to problems in the implemented system and thus may determine the acceptability and usability of software systems (Lauesen and Vinter 2001). Model quality can be described along a set of (partial) dimensions such as: correctness (is the model constructed using the appropriate and agreed syntax?); completeness (is all necessary information available?); consistency (are there any contradictions?) and comprehensibility (is it understood by its intended users?) (Mohagheghi et al. 2009). One commonly cited reason for systems failure is miscommunication between business and IT personnel (Reich and Benbasat 2000). It is likely that a shortcoming of a model will be manifested along one of these quality dimensions. Modelling or, more specifically, conceptual modelling provides a mechanism by which a shared understanding between business domain specialists and IT specialists positively enhances the alignment of business and IT goals leading to improved quality of IT Solutions (Reich and Benbasat 2000). Conceptual models can address different aspects of information systems requirements. The most commonly used application scope of conceptual models is in some form of data (information) modelling. However, it is possible to have conceptual models for process description and functional description such as use cases. Moody identified range of models and their scope with a view to assessing their quality. Such a range requires different languages and structural rules for capturing the model (Moody 2005). This question of quality is all pervasive but is now under greater scrutiny perhaps because conceptual modelling as a production process is maturing. Here, it is useful to draw an analogy with mathematics as described by Godfrey-Smith (2009). Historians of mathematics have noted that mathematical concepts begin life as tools for talking about empirical, or at least accepted, things and then gradually become objects of study in their own right. Similarly a model-building tradition in science goes through an equivalent cycle. Scientists build models to explain things and in so doing engage in approximation and idealisation to fit the explanation. Examples of scientific models include the simple pendulum, models to explain the movements of planets around the sun and of course the Rutherford-Bohr models of the atom. Whilst engineers may dismiss models in science because they consider that science is concerned with explaining phenomena from the natural world, as Loomes and Jones point out:

> In modern accounts of science, however, there is considerable debate over the extent to which scientists are actually dealing with the natural world, rather than the synthetic knowledge structures built by other scientists, thus many aspects of the Philosophy of Science deal with the development of complex information and knowledge systems, rather than "nature" in some primitive sense" ... disciplines such as modern biology focus attention on the ways individual organisms and species develop and evolve through time, and many of the debates surrounding these issues mirror those relating to Software Engineering (Loomes and Jones 1998).

The model in science becomes systematically fictionalised and gives rise to a tradition where it becomes a topic of study in its own right and the quality of a model and how it explains concepts (Bokulich) is one of the most important areas of discourse (Moody 2005). We will return to the 'fictionalised' insertion a little later. First, however, let us focus on quality and its related attributes of validity and accuracy.

Frigg in his account on Scientific Models and Fiction (Frigg 2010) provides a useful list of issues that any model system (the things that make up a model of a target system) should address. These issues are of relevance to models and IS practice and so in this chapter we focus on two of these issues: identity and truth in model systems.

Identity conditions: Model systems are constructed by different authors of the same target system in different ways. This is similar to constructing different models of the same problem domain using different notations and diagramming styles. Frigg's concern is: When are the model systems specified by different descriptions identical?

Truth in model systems: There are right and wrong assertions in any discourse about a model system. On what basis are such assertions qualified as true and false? This is particularly of concern when assertions refer implicitly (or otherwise) to issues for which there is no description in the model.

These issues we argue, need further exploration, and it is the purpose of this chapter to discuss a information systems philosophical problem that has been generated by accounts emerging from the philosophy of fiction. Our central thesis is that conceptual models in information systems practice are not physical models despite their renderings as diagrams. In such models, the truth in a model is difficult to pin down, yet the implicit truthfulness of a model determines its quality and therefore its usefulness. In this chapter, we argue that models share important characteristics in common with literary fiction; therefore, philosophical accounts of theories of fiction can provide an alternative view to how we construct and interpret models. The basis of this argument is predicated on how several philosophers of science have discussed models in science (such as the Rutherford atom model) from a truth in fiction perspective. We introduce a similar account using analogy with scientific models for discussing IS conceptual models. Whilst we set out to provide an exploratory account using the analogy of fiction, the emphasis is on difficulties and issues that this raises for the IS discipline.

2 Models and Fiction

Why fiction The reader may ask? To answer this question, we refer to two areas of discourse. Firstly, IS practice is already utilising techniques drawn from fiction. Secondly, the philosophy of science literature has considered the issue of fiction and models at some length. Earlier, we noted examples of models from science and their role in explaining concepts. Here, we will discuss both aspects in order to explain why we consider the role of fiction to be relevant to models in IS development.

The instruments of fiction, narrative, user stories, actors, roles and so on have been recognised as important to the software design process (Laurel 1993). Narrative-based design methods using techniques such as use cases (Jacobson et al. 1992), scenario-based design (Carroll 1995) and user stories from eXtreme Programming (XP) (Beck 2000) have proven to be effective. Motivation for their use is regarded as originating from user-centred design practice. User-centred design (UCD) or the variant, user-centred systems design (Norman 1986), emerged in the 1980s as an important development recognising the move from batch computing to interactive computing applications where there was a need to involve users in the design process. Whilst their use has become ubiquitous, such fiction instruments are not without shortcomings. In regaling a user story to be later manifested as a requirement in a conceptual model, the author of the story will utilise implicate knowledge and stories need to be seen as contingent on context, for example, the history, differing social perspectives of the storyteller (Stubblefield 2002).

In science, the first step to tackling a problem is to come up with a suitable model that presents a prepared description of the problem. Such a prepared description allows us to remove unnecessary detail, abstract away other detail and allow a focus on what we consider to be the key features of the target system under consideration. We accept in so doing that a full or 'un-prepared' description would make reference to other facts that we have deliberately chosen to exclude. Thus, as Cartwright puts it, 'our prepared descriptions lie' (Cartwright 1983). Such first descriptions are 'as fictions, creatures of the imagination... [a] common property of a community of scientists. They can be investigated collaboratively.' (Godfrey-Smith 2009). Pictorially, they can be represented as in Giere's 1988 analysis of science as shown in Fig. 30.1 (Giere 2004). Here, a language appropriate to the community of practice is used to specify a model system that resembles the target system. The model system may be analysed and debated about and compared with the target system.

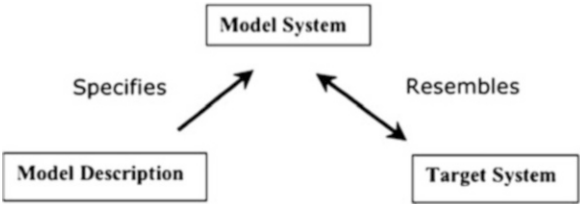


Fig. 30.1 Giere's analysis of models in science

Such a model implicitly seems to belong to the 'semantic' view of models where models are structures that are set-theoretic. A model (S = [U,R]) is a structure S which comprises of a non-empty set U of objects and a non-empty set R of relations on U. These objects and relationships function as an interpreting structure. This approach has formed the basis of most work on conceptual modelling in IS practice as models treated in this way allow us to represent 'any' theory in a formal way including how to model the specification and features of a pizza. The weakness in this approach is the indirection. The model system may not capture the physical nature of the target system without some sense of the use of imagination so 'Taken literally, descriptions that ground structural claims (almost always) fail to be descriptions of the intended target system. Instead, they describe a hypothetical system distinct from the target system'. (Frigg 2010, p. 5).

The parallels drawn between scientific models and fiction have been observed by Cartwright who notes that 'a model is a work of fiction' because some of the properties denoted to objects in the model will be real whilst others will be there for convenience (Cartwright 1983, p. 102).

Having set out the premise that models have a prepared element to them and, in that abstraction, require imagination to complete their ability to explain, we propose that, in the manner that philosophers of science suggest that fiction has similar features to the models used in science, the conceptual models that underpin much of IS practice can similarly be accounted for. Our arguments draw liberally upon key accounts from Smith, Frigg and Toon to present our account to address the issues of identity and truth as described earlier.

As in conceptual modelling, we must ask the question: What is it to be true in a work of fiction? There are several points to consider in answering this question. Firstly, a fictional world is replete with descriptions of places, characters and so on. As competent readers of fiction, we accept and are fully aware that these descriptions do not denote real persons or objects. The situation in the models used in science are similar; frictionless planes, spherical planets, and biological populations living in isolation from the environment are features that do not have counterparts in the real world. Similarly, when we discuss a model of a potential new system in terms of its components and user behaviour, we are describing an imaginary world and all the stakeholders reviewing the model accept that.

Second, within a given fiction, it is reasonable that we infer statements, or facts, that are not explicit in the work's narrative content. A fundamental part of responding to fiction involves the reader 'filling in', or taking for granted an enormous amount of detail that is not explicitly given (Lamarque and Olsen 1996).

It seems therefore that there is a certain class of statements about fiction that are true, despite the fact that there are no passages in the text in which any such statements are expressed clearly (Phillips 1999). For example, it is a fictional truth that the characters in Little Women have functioning digestive systems, respiratory systems and blood running through their veins, though this is never entailed or implied by anything present in the text, yet it would hardly be unreasonable to say that this was the case. The same is true for model systems such as conceptual models. These models are abstractions that represent a partial truth about the target system. We know that a class model entry such as 'student' in a university registration system has other information such as the colour of the eyes of the real-world student that we have chosen not to include. As Frigg points out (Frigg 2010, p. 9):

> The situation with model systems is the same. The description of a model system only specifies a handful of essential properties, but it is understood that the system has properties other than the ones mentioned in the description. Model systems are interesting exactly because more is the true of them than what the initial description specifies; no one would spend time studying model systems if all there was to know about them was the explicit content of the initial description.

Third, any work of fiction has content that goes beyond that what is explicitly stated, and as readers, we have access to that information by various rules of inference. Some of this information is inflationary and superfluous. For example, we know by our knowledge of the human body that humans do not have three nostrils so we can correctly assert that the character Sherlock Holmes does not have three nostrils. Under this analysis, the background against which we reason beyond consists of facts about the actual world. According to this principle, truth in a given fiction depends on matters of contingent fact. If contingent facts are not widely known, then such inference may lead to problems. Inferred truths in fiction can be enhanced by considerations of beliefs shared by a community of interest. If a community shares a set of beliefs, then a fiction can assert a truth by relying on that shared belief. An example of this can be found in the situation where IS development became dominated by user-centred design principles such as ethnography, understanding users and so on, yet designers of some of the devices and interfaces noted by high levels of usability such as those from Apple Inc. do not claim to have used such widely accepted practices (Hong 2011).

An information model describing the types that exist in the system does so in the absence of a user dialogue that might have created an entry in a database. Similarly, we infer how the information stored in an attribute of a type in a model might be used. In a simple estate agent management system, an attribute such as the nearest transport link on a type describing a house for sale will be used by any prospective user of the system as a selection criteria.

Finally, reading serious literature leads the reader to engage in comparisons between situation in the fiction and real circumstances and in doing so, to learn about the world. The novels of John Le Carre, for example, provides direct parallels in the cold war of the 1960s with the fictional world portrayed in his 'Smiley' novels. Again, this has parallels in conceptual modelling. When we develop a conceptual model or even a design model further down the life cycle, we already have one eye on how the system is likely to be deployed – for example, as a web-based application – and therefore, we have knowledge on the likely interaction requirements.

We now drawn upon two recent accounts from philosophy (Frigg 2010, Toon 2010) that have put forward Kendall Walton's 'pretence' theory (Walton 1990) as an approach for understanding how models in science work. In the pretence theory, objects in the pretence world act as 'props' that prompt us to imagine something. An object becomes a prop if there is a rule or 'principle of generation' that prescribes a function that determines what is to be imagined in the presence of an object. The make-believe games of children have many such examples. Rules may be ad hoc or may be publicly announced and agreed upon. For example, the game of pirates usually played in the sports hall deploys the sports hall floor as a prop that states 'that is the sea; when you touch it, you drown' and various sports furniture functions as parts of the pirate ship.

We can refer to this theory and put forward a proposition that interprets conceptual models and modelling as reification of pretence theory. Models should be regarded as *representations* in Walton's sense. A class in an information model such as 'student' is an invitation to imagine the full range of facts that underpins the notion of a student. Furthermore, this rule is publicly announced and so is agreed by a community (e.g. via the UML standard).

Earlier, we raised the issue of truth as being critical to the success of a model. So first we need to discuss in more detail of what truth in fiction means. Frigg describes three types of statements that collectively provide a treatment of truth in fiction. He refers to these as intra-fictional, metafictional and trans-fictional statements. Intra-fictional statements are made within the fiction, and we are not meant to believe them. When Leonard the main protagonist in the novel 'The Innocent' by Ian McEwan describes his 'three lonely years in Birmingham University', we are invited to imagine that. If a few years later, somebody asserts that 'Leonard went to Birmingham University', they are making a metafictional statement because she is talking about the fiction. If she then also says that Leonard is just like a friend of hers who also found Birmingham to be a lonely place, then she is making a trans-fictional statement, as the experience of Leonard is being compared to real persons in another world.

Intra-fictional statements are interesting because we are meant to imagine them. Even a false statement can be qualified as a truth in fiction. In this manner, it is appropriate to talk about a statement by interpreting the statement as 'true in a fictional world', such that we can write a statement: $F_w(P)$ which we read as it is fictional that in the work *w* that *P*. Metafictional statements can be treated in a similar way, so the statement 'Leonard went to Birmingham University' is really saying that in work *w*, Leonard went to Birmingham University which is equivalent to $F_w(P)$. In both cases, the statements can be refined further by adding clauses that state that when a prop in work *w*, *w*-prop, is identified, then the work *w*-rule tells us how to imagine the statement as a truth.

In this way, props via the rules of generation enable the creation of fictional truths. This analysis carries over to conceptual models by replacing *P* by a claim about the model and *w* by a description of the conceptual model. The rules of generation are the principles that have been announced by way of the rules for constructing conceptual models by UML, for example. For example, in a conceptual model, we may represent the truth: A student is only registered for a maximum of three study modules. In this model, the props are the diagram elements such as a rectangle for a class, a line between rectangles representing the registration relationship and an adornment on the line to represent the constraint of three modules limitation. These props engender agreed rules of interpretation to enable us to assert the truth of the student and the maximum number of registrations.

Trans-fictional statements require comparison with a non-existent object. In conceptual modelling though, we are really comparing features of model systems with features of the target system. So assertion of truths comes down to a property comparison. So when we compare a class student in a conceptual model, the properties of that class, such as name, address, and qualifications, are compared to the realworld student person. How that comparison bears will determine the truthness of the conceptual model with respect to the target system. This issue about truth remains a research area in IS practice: "... the concept of truth has not yet been widely considered in the IS research literature... this aspect becomes highly important in analysing, for instance: ...(b) conceptual modelling and modelling in general..." (Beck 2000).

Through this account, we have attempted to address the issue of identity and truth of conceptual models by referring to parallels drawn in fiction.

3 A Diversion to Theory Building

The quality or goodness of a model required for a piece of software can also be viewed from an alternative position – that of theory building exemplified by Peter Naur's seminal 1985 paper 'Programming as Theory Building'. The fundamental premise asserts that programming should be regarded as an activity by which programmers achieve a certain insight or theory of some aspect of the domain that they are addressing. The knowledge, insight or theory that the programmer has come into possession of is a theory in the sense of Ryle [1949], that is, a person who has a theory knows how to do certain things and can support the actual doing with explanations, justifications and responses to queries. That insight or theory is primarily one of building up a certain kind of knowledge that is intrinsic to the programmer, whilst any auxiliary documentation remains a secondary product. How does this relate to models? A key feature of modelling is the existence of an isomorphic relationship between the parts of the model and the parts of the thing modelled at some level of abstraction. Smith [1991] whilst noting these different types of models emphasises the nature and importance of 'representation':

> To build a model is to conceive of the world in a certain delimited way... Computers have a special dependence on these models: you write an explicit description of the model down inside the computer....

Smith suggests this feature distinguishes computers from other machines because they run by manipulating representations. 'Thus there is no computation without representation' (Smith 1991, p. 360). If we pursue this analysis further from Naur, we can state that the program is a theory; from general computation principles, we can state that the program is a model. This leads to the notion that there is an equivalence between program theory model. We might moderate this further by noting that a program is a representation of a slice of 'the' theory. In general though, this blurring between programs, theories and models is confusing and possibly inaccurate. Whilst models may exhibit an isomorphic relationship with their subject matter, this relationship may not reveal the theoretical connections that allow the theory to be defended in the form of Ryle's definition of a theory. In an essay that predates Naur's paper but still based on a prevailing view of the time that programs are theories, James Moor notes:

> Even if there is some theory behind a model, it cannot be obtained by simply examining the computer program. The program will be a collection of instructions which are not true or false, but the theory will be a collection of statements which are true or false. Thus, the program must be interpreted in order to generate a theory. Abstracting a theory from the program is not a simple matter for different groupings of the program can generate different theories. Therefore, to the extent that a program, understood as a model, embodies one theory it may well embody many theories. (Moor 1978, p. 221)

From this analysis arises two key concerns. Firstly, programs and models may have multiple theories, and a program or model may not refer to the same theory. Secondly, these theories must be statable independent of the program or the model. The first point of multiple theories is consistent with the Popperian understanding that questions the notion of 'the' theory as a theory is intrinsic to the human. Much like tacit knowledge, there may be more than one way of 'knowing' how to ride a bicycle, and as we cannot explicate that knowledge, then these multiple theories may exist.

What does this mean for the general thread of our argument around the role of fiction in model building and understanding? We propose that the partial theories built as a result of constructing a model are supplemented by the fiction account, that is, multiple theories can happily coexist but add different perspectives. The role of the fiction viewpoint is to provide an explanation for why multiple theories exist for a program and to provide a mechanism for how one theory can be explained by another theory. The use of the constructs of pretence theory, for example, will allow trans-fictional statements to map from one theory (model) to another theory (model). Consider an example. Imagine an organisation that wants to create a system that enables its employees to make selections regarding updating their computers. An employee may choose a desktop computer (Mac OS or Windows) or a laptop (Mac or Windows). Whilst an information model will capture part of the properties required for selection and therefore be partial model of the domain, certain properties will act as props as in pretence theory that will trigger other information or knowledge about model. The property MacOS X 10.6.7, for example, will act as a prop that may encourage the end user to consider facts that this operating system is less open to malware. The manufacturer name Apple may similarly trigger a generative rule that makes the end user think that the computer is overpriced. This additional information resides in the fictional world but nonetheless helps give a richer model or theory about the domain of discourse.

4 Concluding Implications for IS and Conceptual Modelling

At this stage, the reader may be wondering why. As we laid out in the introduction, we may raise more questions than answers. We suggest that thinking about how we read and interpret fiction may give us new possibilities on how we read and interpret conceptual models. We have attempted to lay out the relationship between fiction and modelling by exploring some of the key concepts and arguments from the general area of philosophy and fiction. We have also included a small but relevant diversion to theory-building views. From this, we may consider specific issues and implications for the IS practice of conceptual modelling. One key issue is that implicit truths are embedded in a model, and in particular, how can we bring to the fore more readily the contingent facts of the domain? In fiction, we rely on the contingent facts and therefore infer other truths. How can this become second nature to models? Or is it already second nature? Consider the example of social software. When props in a model trigger generative truths about how a social software platform should behave, those rules are based on an experiential knowledge of social software platforms. We expect our proposed system to support those functions that we would normally expect to be there. Thus, a prop can function as a trigger for 'importing' additional requirements. Research questions on how to identify, annotate and set up rules for props could then address new tools for making more explicit the contingent and inferred truths. A second issue is how emerging conceptual modelling methods and languagee such as those used for enterprise architecture have focussed on developing artefacts and models for explicit knowledge (Lankhorst 2009, p. 75). Accepting the theory-building view forces us to reject firstly that such an exhaustive methodological approach can lead us to a universal theory of enterprise architecture for the domain under question. Secondly, the focus on explicit knowledge does not allow us to extract from the plethora of method the essence of 'why'. Instead, applying some of the principles and concepts from theories of truth in fiction may afford new opportunities for research.

References

Beck K (2000) Extreme programming explained: embrace change. Addison-Wesley Professional, Reading

Bokulich A How scientific models can explain. Synthese :1–13

Booch G, Rumbaugh J, Jacobson I (2005) Unified modeling language user guide. The Addison-Wesley object technology series. Addison-Wesley Professional, Upper Saddle River
Carroll JM (1995) Scenario-based design: envisioning work and technology in system development. Wiley, New York
Cartwright N (1983) How the laws of physics lie. Clarendon Press, Oxford
Frigg R (2010) Models and fiction. Synthese 172(2):251–268
Giere RN (2004) How models are used to represent reality. Philos Sci 71:742–752
Godfrey-Smith, P (2009) Models and fictions in science. Philos Stud 143(1):101–116
Hong J (2011) Matters of design, part II. Commun ACM 54(4):10–11
Jacobson I, Christerson M, Jonsson P, Overgaard G (1992) Object-oriented software engineering: a use case driven approach. Addison-Wesley, Reading
Lamarque P, Olsen SH (1996) Truth, fiction, and literature: a philosophical perspective. Oxford University Press, Oxford
Lankhorst M (2009) Enterprise architecture at work: modelling, communication and analysis. Springer, New York
Lauesen S, Vinter O (2001) Preventing requirement defects: an experiment in process improvement. Requir Eng 6(1):37–50
Laurel B (1993) Computers as theatre. 1991. Addison-Wesley, Reading
Loomes MJ, Jones S (1998) Requirements engineering: a perspective through theory-building. In: ICRE. IEEE Computer Society, Los Alamitos, p 0100
Mohagheghi P, Dehlen V, Neple T (2009) Definitions and approaches to model quality in model-based software development-a review of literature. Inf Softw Technol 51(12):1646–1669
Moody DL (2005) Theoretical and practical issues in evaluating the quality of conceptual models: current state and future directions. Data Knowl Eng 55(3):243–276
Moor JH (1978) Three myths of computer science. Br J Philos Sci 29(3):213–222
Norman DA (1986) Cognitive engineering. In: Norman DA, Draper SW (eds) User centered system design. Lawrence Erlbaum Associates, Hillsdale, pp 31–61
Phillips JF (1999) Truth and inference in fiction. Philos Stud 94(3):273–293
Reich BH, Benbasat I (2000) Factors that influence the social dimension of alignment between business and information technology objectives. MIS Q 24:81–113
Ryle G (1949) The concept of mind. Hutchinson, London
Smith BC (1991) Limits of correctness in computers. Academic, London
Stubblefield WA (2002) Narrative and meaning in ordinary interactive software. In: *Storytelling and collaborative activities workshop (CSCW 2002)*. Citeseer
Toon A (2010) Models as make-believe. In: Frigg R, Hunter M (eds) Beyond mimesis and convention. Springer, Dordrecht/London, pp 71–96
Walton KL (1990) Mimesis as make-believe: on the foundations of the representational arts. Harvard University, Cambridge

Chapter 31
An Ontological Analysis of Metamodeling Languages

Erki Eessaar and Rünno Sgirka

1 Introduction

Metamodeling systems (meta-CASE systems) are used to create new modeling systems (CASE systems), which allow developers to model systems by using one or more modeling languages; to test the models; and possibly to generate program code based on the models. These modeling languages are usually domain-specific languages (Kelly and Tolvanen 2008). Each metamodeling system provides a *metamodeling language* (metalanguage) for specifying modeling languages. There are many different metamodeling systems and hence also metamodeling languages. It raises a question, whether some of the metamodeling languages are better than others and how to find it out. Specification of a formal language should specify semantics, abstract syntax, concrete syntax, and serialization syntax of the language (Greenfield et al. 2004). In this chapter, we concentrate our attention to the evaluation of the abstract syntax of languages.

Livingstone (2008) argues that a programming language should be conceptually simple and hence have characteristics like parsimony, straightforwardness, generality, orthogonality, and uniformity. Siau and Rossi (1998) have proposed a set of evaluation methods of information modeling methods. This kind of methods can be used to analyze different modeling languages – not only information modeling languages. One of the proposed nonempirical methods is *ontological analysis*. Guizzardi (2005) presents the framework for performing the ontological analysis of artificial modeling languages. There exist examples of ontological analysis of modeling languages. These analyses use *foundational ontologies* (also known as upper ontologies or top-level ontologies), which specify domain-independent categories and are theoretically well founded (Guizzardi et al. 2008). For instance, Bunge-Wand-Weber (BWW) ontology (Wand and Weber 1990) has been used to

E. Eessaar (✉) • R. Sgirka
Department of Informatics, Tallinn University of Technology, Tallinn, Estonia
e-mail: eessaar@staff.ttu.ee; runno.sgirka@gmail.com

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_31,

analyze and redesign UML 1.3 (Opdahl and Henderson-Sellers 2002) and Architecture of Integrated Information Systems (ARIS) (Green and Rosemann 1999). More recently, Unified Foundational Ontology (UFO) has been used to analyze and redesign Software Process Ontology, which is a domain ontology (Guizzardi et al. 2008), and UML 2.0 (Guizzardi and Wagner 2010). Guizzardi and Guizzardi (2010) use UFO to design an agent-oriented engineering language for the ARKnowD methodology.

In this chapter, we apply the ontological analysis method (Guizzardi 2005) to the metamodeling languages. The *first goal* of this chapter is to investigate how to perform ontological analysis of metamodeling languages and whether the results of the analysis give language designers sufficient information to improve the quality of metamodeling languages. Therefore, we present the results of a small experiment, during which we analyzed two metamodeling languages by using a foundational ontology. One of the languages is used in a web-based and database-based metamodeling system WebMeta (Eessaar and Sgirka 2010), which we have developed over time. Hence, the *second goal* is to find out whether the metamodeling language of WebMeta needs improvement and, if it does, then propose improvements that are ontologically well founded.

The rest of this chapter is organized as follows: In Sect. 2, we discuss how to perform ontological analysis of metamodeling languages. In Sect. 3, we present the results of an ontological analysis of two metamodeling languages and suggest improvements to one of the languages. Finally, we conclude and point to the further work with the current topic.

2 Ontological Analysis of Metamodeling Languages

It is possible to specify the abstract syntax of a modeling language by using a metamodel (Greenfield et al. 2004). If we use UML class diagrams to represent a metamodel, then modeling constructs are represented as classes and relationships between the modeling constructs are represented as associations or generalizations. In this chapter, we call the metamodel of a metamodeling language a meta-metamodel, based on the MOF metamodeling architecture (Meta Object Facility).

Each metamodeling language is a domain-specific modeling language that is used to specify modeling languages. These modeling languages will be used to create models, which specify static and/or dynamic characteristics of very different subject areas. For instance, Kelly and Tolvanen (2008) present five examples of domain-specific languages, which were implemented by using the same metamodeling language: IP telephony and call processing, insurance products management, microcontroller applications, mobile phone applications, and digital wristwatch applications.

The modeling constructs in a metamodeling language are very generic, because they must facilitate definition of very different languages, which are used in many different domains. Foundational ontologies describe very general concepts like

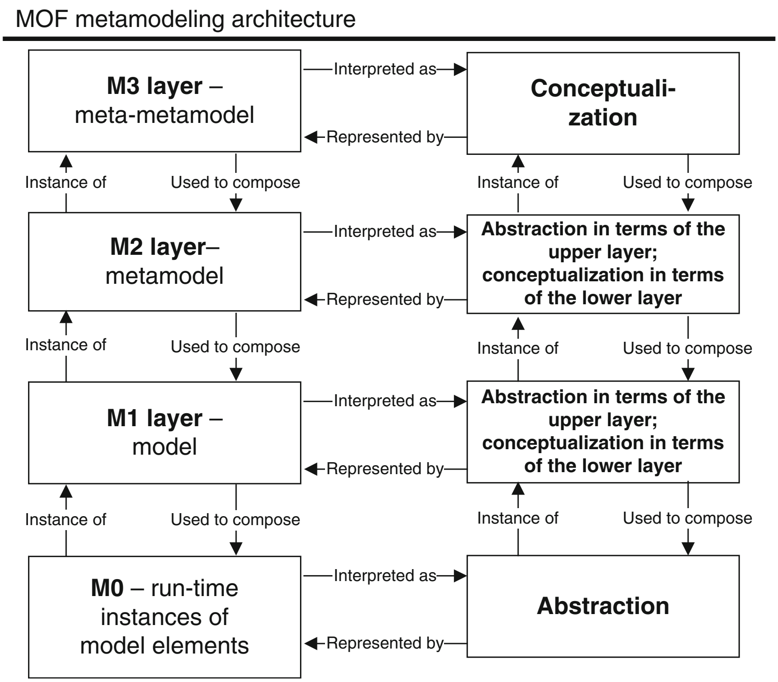


Fig. 31.1 Correspondence between four layers of the MOF metamodeling architecture and conceptualization and abstraction

Entity, *Particular*, and *Universal*. Therefore, foundational ontologies are the suitable basis for the ontological analysis of metamodeling languages.

The elements that make up a *conceptualization* of a domain D are used to articulate *abstractions* of certain state of affairs in reality (Guizzardi and Wagner 2010). Each ontology is a formal specification of a domain conceptualization.

Any number of meta-layers, greater than or equal to two, are permitted by the MOF 2.0 metamodeling architecture (Meta Object Facility). For instance, UML infrastructure specification (OMG Unified Modeling Language™ uses a four-layer metamodel hierarchy, where M3 is the highest layer. Figure 31.1 presents the correspondence between four layers of the metamodeling architecture and conceptualizations and abstractions. For instance, a certain conceptualization of the domain of modeling language design can be constructed by considering concepts such as *Monadic Universal* and *Relation*, among others. These concepts are represented by the modeling constructs in a metamodeling language.

By using these concepts, it is possible to articulate a domain abstraction that a certain modeling language allows us to model *Actors*, *Use cases*, and relationships between *Actors* and *Use cases*.

On the other hand, these abstractions also constitute a domain conceptualization of the domain of use-case modeling. These concepts are represented by the modeling constructs in a modeling language (in this case, the language for creating

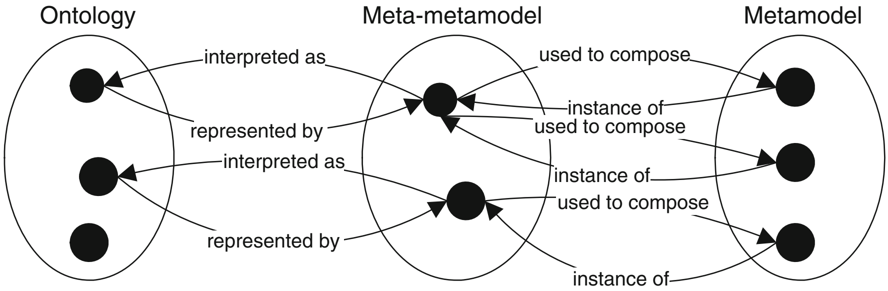


Fig. 31.2 Example of mappings between the elements of an ontology, a meta-metamodel, and a metamodel, which has been created based on the meta-metamodel

use-case models). By using these concepts, it is possible to articulate a domain abstraction that, for example, an actor *Client* is associated with a use-case *Make an order* in a particular model. On the other hand, these abstractions also constitute a domain conceptualization of the domain of a particular information system. These concepts are represented by the model elements in a model. By using these concepts, it is possible to articulate a domain abstraction that, for example, an actor *John Smith* makes an order with number *O110234*.

For each layer of the metamodeling architecture, there are corresponding languages, which are used to create models that correspond to this layer. It is possible to perform the ontological analysis of all these languages. In this chapter, we are interested in the ontological analysis of metamodeling languages, which correspond to the M3 level of the architecture.

Ontological analysis of a metamodeling language L means that one has to compare the metamodel of L with an ontology O to find possible violations of the following desired properties of L: completeness, soundness, lucidity, and laconism (Guizzardi and Wagner 2010). If L has these properties, then the metamodel of language L and ontology O is isomorphic, and it should reduce the problems of using L.

A metamodeling language L is *complete* in terms of a domain D if and only if every concept in the ontology O of that domain is represented in a modeling construct of L (Guizzardi 2005; Guizzardi and Wagner 2010). Each metamodeling language is used to specify zero or more modeling languages (see Fig. 31.2). Hence, in case of evaluating completeness of metamodeling languages, one could *also* investigate whether it is possible to represent every concept in O in the metamodel of at least one language, which is created by using the metamodeling language. For instance, one of the concepts in the UFO ontology is *Role* (Guizzardi and Guizzardi 2010). During the analysis of completeness of a metamodeling language L based on the UFO ontology, one could investigate, whether L allows language designers to create a modeling language L' where one of the modeling constructs represents the concept *Role* in the UFO. We do not conduct this analysis in this chapter.

A metamodeling language L is *sound* in terms of a domain D if and only if every modeling construct in L has an interpretation in terms of a domain concept in the ontology O (Guizzardi 2005; Guizzardi and Wagner 2010). A modeling construct *c* in a modeling language L', which is created by using L, might not have a corresponding concept of O that provides its interpretation. Metamodeling languages should not be too restrictive and should allow language designers to define this kind of modeling constructs in modeling languages. However, it is a violation of the soundness property in case of L', and one has to perform ontological analysis of L' to find the problem.

A metamodeling language L is *lucid* in terms of a domain D if and only if every modeling construct in L represents at most one domain concept in O (Guizzardi 2005; Guizzardi and Wagner 2010).

A metamodeling language L is *laconic* in terms of a domain D if and only if every concept in the ontology O of that domain is represented at most once in the metamodel of L (Guizzardi 2005; Guizzardi and Wagner 2010).

In addition, one has to evaluate, whether the meta-metamodel of L follows all the constraints, which have been specified in the ontology O.

3 An Experiment

In this section, we present the results of an ontological analysis of two metamodeling languages. A problem, which limits the selection of metamodeling languages for the analysis, is that complete specifications of the abstract syntax of some of the languages are not publicly available.

We selected the following languages, which have been developed in-house by our university: a metamodeling language that is used in our web-based and database-based metamodeling system WebMeta (ver. 0.5) (Eessaar and Sgirka 2010) (see Fig. 31.3) and a metamodeling language, which is proposed to use in the context of evolutionary information systems (Roost et al. 2007) (see Fig. 31.4).

We selected Unified Foundational Ontology (UFO) as the foundational ontology, which is used as the basis in the ontological analysis. A reason is that it has been recently used in other ontological analysis as well. Another reason is that the ontology is documented by using diagrams that resemble UML class diagrams, and it simplifies the ontological analysis process.

Rosemann et al. (2004) suggest that one has to set the scope of an ontological analysis of a language L by selecting a subset S of concepts of the ontology O. Only the concepts that belong to S will be used in the analysis. The subset must contain only these concepts of O that are relevant in terms of the language L metamodel. Metamodeling languages are used to specify the structure of modeling languages. Metamodeling languages are not used to specify behavior or social concepts. Hence, we used in the ontological analysis a subset (a compliance set) of UFO, namely, *UFO-A: An Ontology of Endurants*. However, if one wants to analyze whether it is possible to represent every concept in the foundational ontology O in the metamodel of at least one language, which is created by using the metamodeling language, then one has to use O completely and not only a subset of O.

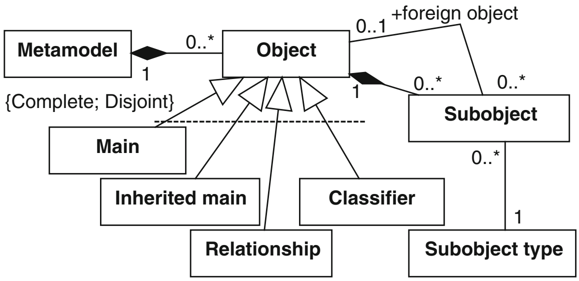


Fig. 31.3 Meta-metamodel of the metamodeling system WebMeta (ver. 0.5)

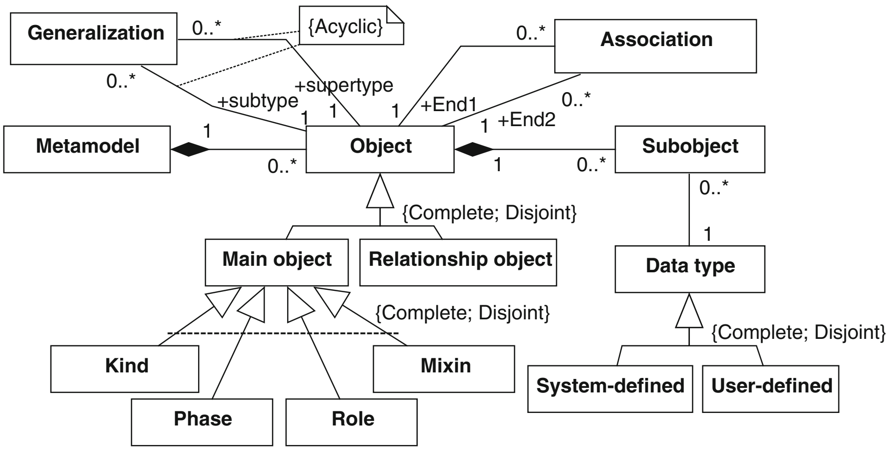


Fig. 31.4 A redesign meta-metamodel of the WebMeta metamodeling system

A problem is that the UFO ontology is still in active development and has not yet completely stabilized. Therefore, the descriptions of UFO elements are somewhat different in different articles. However, UFO-A is more mature and stable compared to other parts of UFO (Guizzardi and Wagner 2010). The current analysis is based on the UFO-A specification, which is presented in (Guizzardi and Wagner 2010). We do not present in this chapter a formal characterization of UFO-A due to the lack of space.

Rosemann et al. (2004) suggest that in order to improve the quality of the ontological analysis of a language L, the metamodel of L and an ontology O, which is used in the ontological analysis, must be represented by using the same language. In case of this analysis, we used the meta-metamodels, which were represented by using UML class diagrams. The ontology (UFO-A) that is used in the analysis is represented in (Guizzardi et al. 2008; Guizzardi and Wagner 2010) by using diagrams, which resemble UML class diagrams.

Table 31.1 Mappings between the modeling constructs of the WebMeta (ver. 0.5) metamodeling language and concepts of the UFO-A ontology

Metamodeling language	Ontology
Classifier object	Quality structure
Inherited main object	Object universal, Basic formal relation
Main object	Object universal
Metamodel	Abstract set
Object	Universal
Relationship object	Material relation
Subobject	Quality universal, Relator universal
Subobject type	Quality structure

3.1 WebMeta Metamodeling Language

The meta-metamodel of the metamodeling language, which is the basis of WebMeta metamodeling system (Eessaar and Sgirka 2010), specifies the following classes: *Metamodel*, *Object*, *Main, Inherited main*, *Classifier*, *Relationship, Subobject*, and *Subobject type* (see Fig. 31.3). Each *Metamodel* specifies the abstract syntax of a modeling language (that belongs to the M2 layer; see Fig. 31.1). *Classifier* is used to specify modeling language constructs that help modelers to characterize some other modeling language constructs. *Relationship* is used to specify modeling language constructs that relate other modeling language constructs. *Main* and *Inherited main* are used to specify modeling language constructs that are not classifiers or relationships. *Inherited main* can be used to specify a modeling language construct based on inheritance from some other *Main* or *Inherited main* modeling language construct. *Subobject* is used to specify the properties of modeling language constructs and associations between modeling language constructs. Examples of *Subobject type* are integer, varchar(100), and boolean.

The metamodeling language of WebMeta *is sound*, because every modeling construct in the metamodeling language has an interpretation in terms of a domain concept in the UFO-A ontology (see Table 31.1). For instance, the ontology element *Abstract set* provides interpretation to the language construct *Metamodel*. Metamodel is a set of model elements, which together specify the abstract syntax of a language.

Each data type is a named, finite set of values (Date 2006). Hence, we agree with Guizzardi and Wagner (2010) that the ontology element *Quality structure*, which is a subclass of *Set* in the UFO-A ontology, is the ontological interpretation of the *Subobject type* construct in the metamodeling language. Each *Classifier object* is conceptually a set of values that are used to characterize model elements at the M1 layer of the MOF metamodeling architecture. For instance, a use-case modeling language could allow modelers to specify the importance of each particular use case. One could define a classifier object *Priority* that belongs to the *Use case* metamodel. The values that belong to the classifier object and can be used to characterize use cases could be “low,” “medium,” and “high.” Hence, we think that the ontological interpretation of the modeling construct *Classifier object* must be the same than the interpretation of the modeling construct *Subobject type*. The modeling construct *Subobject type* represents system-defined (simple) data types. The modeling construct *Classifier object* represents user-defined data types (more precisely, *enumerated types*).

The metamodeling language of WebMeta *is not complete*, because there are concepts in the UFO-A ontology that are not represented by a modeling construct of the metamodeling language.

For instance, there are concepts like *Kind*, *Role,* and *Phase*, as well as *Concrete particular* that are not represented by a modeling construct of the metamodeling language.

The metamodeling language of WebMeta *is not lucid* because there are modeling constructs in the metamodeling language, which represent more than one domain concept in the ontology (see Table 31.1). The modeling construct *Inherited main* represents *Object universal* and *Basic formal relation*. If we use *Inherited main* construct in a metamodel, then it means that there is an inheritance relationship in the metamodel. Inheritance belongs to the category *Basic formal relation* (Guizzardi and Wagner 2010).

The modeling construct *Subobject* represents the ontology concepts *Quality universal* and *Relator universal*. *Subobjects* are used to represent properties of objects. For instance, if we specify use-case modeling language, then the fact that each use case must have a name would be specified by defining the subobject *name* of the object *Use case*. Each *name* is existentially dependent of one single particular – a *Use case*. *Subobjects* are also used to represent relationships between objects. The fact that each use case should be associated with a primary actor could be specified by defining the subobject *primary actor* of the object *Use case*. The subobject would have the foreign object *Actor*. Each *primary actor* is existentially dependent on a plurality of particulars – a *Use case* and an *Actor*.

The metamodeling language of WebMeta *is not laconic*, because there are some concepts in the UFO-A ontology which are represented by more than one modeling construct in the metamodeling language (see Table 31.1). The ontology concept *Object universal* is represented by the modeling constructs *Main object* and *Inherited main object*. For instance, one can define the modeling construct *Use case* by creating a main object. However, one can also define the modeling construct *Use case* by creating the main object *Model element* and then creating the inherited main object *Use case* based on the main object *Model element*. The concept *Quality structure* is represented by the modeling constructs *Subobject type* and *Classifier object*. Hence, while creating a metamodel, one has to decide whether to specify the possible values of a property of a language construct by using a predefined *Subobject type* or by specifying a new set of values.

Figure 31.4 presents the *first version* of a redesigned meta-metamodel of the WebMeta metamodeling language. Our goal is not to present *detailed* description of the new language but to *illustrate* useful results of ontological analysis.

Quality structure is a subclass of *Particular* according to the UFO-A ontology. However, *Classifier object* is specified as a subclass of *Object* (the ontological interpretation of which is *Universal*) in the original meta-metamodel (see Fig. 31.3). Hence, we propose to redesign the metamodeling language in a way that *Classifier object* will not be a subclass of *Object* any more. Instead, in the new model, we have class *User-defined (type)* that is a subclass of *Data type*. For the sake of clarity, we propose to rename *Subobject type* to *System-defined (type)*.

Main object has in the new model subclasses that represent the ontology concepts, which are the subclasses of *Object universal* in the UFO-A ontology.

It improves the completeness of the metamodeling language and makes it possible to enforce the constraints, which are prescribed by UFO (Guizzardi 2005) and regulate the relations between these different types of universals. For instance, a language designer could define use-case modeling language construct *Actor* by using the metamodeling language construct *Kind*. One could also define use-case modeling language construct *Primary actor* by using the metamodeling language construct *Role*. However, definition of a generalization relationship, according to which *Primary actor* is the supertype of *Actor*, must be prohibited based on a constraint that is defined in the UFO (Guizzardi 2005).

We have added constructs *Generalization* and *Association* to the metamodeling language to increase the lucidity of the language. The construct *Subobject* is now only used to specify the properties of model constructs in the metamodels (*Quality universals* in terms of the UFO-A ontology).

An ontological analysis of the redesigned metamodeling language (based on the UFO-A ontology) shows that the language is sound and lucid. However, the language is not complete and not laconic. The language is not laconic because the concept *Quality structure* in the ontology is represented by three modeling constructs: *Data type*, *System-defined data type*, and *User-defined data type*. The language is not complete because, for instance, the concept *Concrete particular* is not represented in a modeling construct of the metamodeling language.

3.2 A Metamodeling Language Proposed in Roost et al. (2007)

The metamodeling language, which is proposed in Roost et al. (2007), is very generic. The meta-metamodel has only one class – *Element* (see Fig. 31.5).

The metamodeling language *is not complete* because only the most generic concept of the ontology (*Entity*) is represented by a construct (*Element*) in the metamodeling language.

In addition, the relationships in the meta-metamodel represent some (but not all) *Basic formal relations* in the UFO-A ontology.

The metamodeling language *is sound*. The concept *Entity* in the ontology provides the interpretation to the modeling construct *Element* in the metamodeling language. The metamodeling language *is lucid* because the modeling construct *Element* in the metamodeling language represents exactly one concept (*Entity*) in

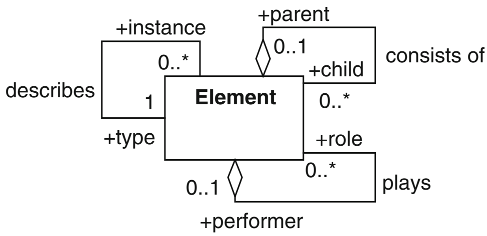


Fig. 31.5 Meta-metamodel of a metamodeling language, which is proposed to use in the context of evolutionary information systems (Roost et al. 2007)

the ontology. The metamodeling language *is laconic* because every concept in the ontology is represented at most once in the meta-metamodel of the language.

Based on the meta-metamodel, an *Element* can be an instance of itself. It violates a disjointness constraint, which is specified in UFO-A, according to which an *Entity* cannot be a *Particular* and a *Universal* at the same time.

3.3 Discussion

It is hard (and in our view unnecessary) to achieve completeness in terms of a complete foundational ontology in case of metamodeling languages. The reason is that each metamodeling language has to provide only a small number of generic modeling constructs. The constructs have to be generic because they will be used to specify different modeling constructs (with different semantics) of different modeling languages. Hence, we used in the analysis only a subset of a foundational ontology – *UFO-A: An Ontology of Endurants*. However, UFO-A specifies the concept *Concrete particular* and its subclasses. A metamodeling language L does not have to contain the constructs that represent these concepts because most of the constructs of L (except *Data type* and its subclasses) represent *universals*, which can be *instantiated* at the lower layer of the metamodeling architecture.

It might be a good idea to change the representation of the meta-metamodel of a language, without changing the semantics of the model, to facilitate the ontological analysis. For instance, in WebMeta (ver 0.5), the database schema, which implements the meta-metamodel, is created according to the model (a) of Fig. 31.6. However, for the analysis, we presented the existence of different types of objects by using subclasses (see part b of Fig. 31.6) and a constraint {Complete; Disjoint}.

The constraint shows that each instance of a superclass (*Object*) belongs to exactly one of the subclasses of *Object*. In case of model, (b) it is easier to map constructs of a metamodeling language with the concepts in an ontology.

The ontological analysis detected only few problems in a very generic and flexible meta-metamodel (see Sect. 3.2). However, in case of this very generic meta-metamodel, one could more easily violate the constraints that are specified in the foundational ontologies, while defining a metamodel. Ontological analysis is used to evaluate only one aspect of a language (its abstract syntax), and it should not

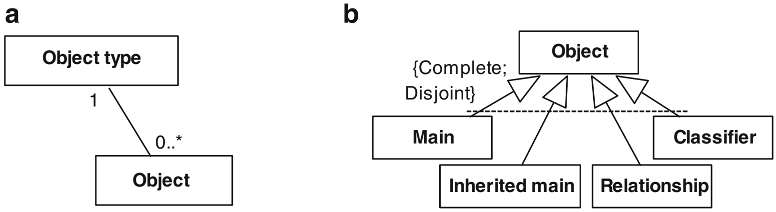


Fig. 31.6 Different representations of a meta-metamodel

be the only evaluation method of metamodeling languages. A problem of the ontological analysis is that it is somewhat subjective in nature and depends on the selection of ontology as well as on the understanding of the ontology and metamodeling languages by the evaluators. However, even in this way, it can give useful information about the deficiencies of a particular language (see Sect. 3.1).

Laarman and Kurtev (2010) use a simple foundational ontology (four-category ontology) to construct an ontologically well-founded metamodeling language (ontology-grounded metalanguage). This chapter complements the paper (Laarman and Kurtev 2010) by suggesting the use of ontological analysis to improve the *existing* languages.

4 Conclusions

In this chapter, we investigated how to conduct an ontological analysis of metamodeling languages. As an example, we performed an ontological analysis of the metamodeling language of our own metamodeling system WebMeta (ver. 0.5) (Eessaar and Sgirka 2010) and a metamodeling language described in (Roost et al. 2007). We used a subset of the Unified Foundation Ontology (namely, UFO-A) as the basis of the analysis. We found several problems of the metamodeling languages and presented a redesigned meta-metamodel of the metamodeling language of WebMeta. We conclude that it is possible and useful to conduct ontological analysis of metamodeling languages. It would help language designers to find problems of languages and compare languages in terms of the number and severity of the problems. However, the investigation of completeness of a metamodeling language would probably lead to the conclusion that the language is incomplete.

Future work should include similar analysis based on the bigger set of metamodeling languages. In addition, it would be necessary to find out whether the ontological analysis of the same metamodeling languages based on other foundational ontologies would give the same results. We also have to continue the improvement of the metamodeling language of WebMeta.

Acknowledgments This research was supported by European Social Fund's Doctoral Studies and Internationalisation Programme DoRa.

References

Date CJ (2006) The relational database dictionary. A comprehensive glossary of relational terms and concepts, with illustrative examples. O'Reilly, Sebastopol

Eessaar E, Sgirka R (2010) A database-based and web-based meta-CASE system. In: International conference on systems, computing sciences and software engineering. Springer, Dordrecht, pp 379–384

Green P, Rosemann M (1999) An ontological analysis of integrated process modeling. In: CAiSE'99. LNCS vol 1626. Springer, Berlin, pp 225–240

Greenfield J, Short K, Cook S, Kent S (2004) Software factories: assembling applications with patterns, models, frameworks, and tools. Wiley, Indianapolis

Guizzardi G (2005) Ontological foundations for structural conceptual models. Telematica Instituut Fundamental Research Series No. 15. Ph.D. thesis, University of Twente

Guizzardi RSS, Guizzardi G (2010) Applying the UFO ontology to design an agent-oriented engineering language. In: ADBIS'10. LNCS vol 6295. Springer, Berlin, pp 190–203

Guizzardi G, Wagner G (2010) Using the unified foundational ontology (UFO) as a foundation for general conceptual modeling languages. In: Theory and applications of ontology: computer applications. Springer, Dordrecht, pp 175–196

Guizzardi G, Falbo R, Guizzardi SS (2008) Grounding software domain ontologies in the Unified Foundational Ontology (UFO): the case of the ODE software process ontology. In: XI Iberoamerican workshop on requirements engineering and software environments, Recife, Brazil

Kelly S, Tolvanen JP (2008) Domain specific modeling enabling full code generation. A Wiley-Interscience Publication, Hoboken

Laarman A, Kurtev I (2010) Ontological metamodeling with explicit instantiation. In: SLE 2009. LNCS vol 5969. Springer, Heidelberg, pp 174–183

Livingstone D (2008) Simplicity. Systemist 30:16–39

Meta Object Facility (MOF) Core Specification. Version 2.4 Convenience, ptc/2010-12-08

OMG Unified Modeling Language^{TM} (OMG UML), Infrastructure. Version 2.4, ptc/2010-11-16

Opdahl AL, Henderson-Sellers B (2002) Ontological evaluation of the UML using the Bunge–Wand–Weber model. Softw Syst Model 1(1):43–67

Roost M, Rava K, Veskioja T (2007) Supporting self-development in service oriented information systems. In: 7th WSEAS international conference on applied informatics and communications, Athens, Greece, pp. 52–57

Rosemann M, Green P, Indulska M (2004) A reference methodology for conducting ontological analyses. In: ER 2004. LNCS vol 3288. Springer, Heidelberg, pp 110–121

Siau K, Rossi M (1998) Evaluation of information modeling methods-a review. In: Thirty-first Hawaii international conference on system sciences, 5th edn. IEE Computer Society, Los Alamitos, pp 314–322

Wand Y, Weber R (1990) An ontological model of an information system. IEEE Trans Softw Eng 16(11):1282–1292

Chapter 32
On XML Document Transformations as Schema Evolves: A Survey of Current Approaches

Jakub Malý, Irena Mlýnková, and Martin Nečaský

1 Introduction

The *eXtensible Markup Language* (XML) (Bray et al. 2008) is currently one of the most popular meta-formats for data representation on the Web. To enable data exchange, it is crucial to restrict structure of the exchanged XML documents by a set of rules, so that each communicating party is able to understand them. This restriction is achieved via *XML schemas*.

As user requirements change during the life cycle of the system, so do the XML schemas. This *evolution* of XML schemas should be reflected in their instances – the changes made to XML schemas during evolution should be *propagated* to the set of XML documents present in the system. Without any tools to help, the old and the new schema need to be examined by a domain expert, each change must be identified, analyzed, and all the relevant parts of the system modified, respectively. The documents conforming to the old version of each schema must be updated in order to conform to the new version. This process, called *revalidation*, can be both time-consuming and error prone when performed manually. In fact, a significant portion of the operations could be performed automatically. This is the reason why frameworks and systems for schema evolution and document revalidation have emerged.

In this work, we propose a general set of operations that an applicable general evolution framework should support, we list the favorable architectonic/design features of such framework, and we examine the extent of support for these operations in current systems. This chapter should serve as a useful source of information for users of the systems, their vendors, as well as researchers interested in the area.

J. Malý (✉) • I. Mlýnková • M. Nečaský
Faculty of Mathematics and Physics, XML Research Group, Department of Software Engineering, Charles University in Prague, Malostranské náměstí, 118 00 Praha 1, The Czech Republic
e-mail: maly@ksi.mff.cuni.cz; mlynkova@ksi.mff.cuni.cz; necasky@ksi.mff.cuni.cz

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_32,

Outline The rest of this chapter is organized as follows: In Sect. 32.2, we list the operations that an evolution framework should support. In Sect. 32.3, we name desired general capabilities of such framework. In Sect. 32.4, we describe three main architectures. In Sect. 32.5, we study how the existing approaches met the stated requirements. And Sect. 32.6 concludes.

2 Requirements for an XML Schema Evolution Framework

In this section, we define universal required properties and capabilities of an evolution framework.

In the following text, we will use the notion $\mathcal{S}$ for an XML schema and $\mathcal{S}'$ to denote the schema evolved from $\mathcal{S}$. There are several schema languages being used. Three most popular – DTD (Sperberg-McQueen et al. 2000), XML schema (Thompson et al. 2004), and RELAX NG (Clark and Makoto 2001) – are based on *regular tree grammars* (Murata et al. 2005); Schematron (ISO 2005) is a *rule-based language* and is usually used as a support for a schema expressed via one of the previous three languages (i.e., a schema in a *grammar-based language* defines structure and nesting, and a Schematron schema specifies additional integrity constraints and rules).

The following definitions are independent of notation used for expressing the schemas (particular XML schema languages), but for the purposes of this chapter, we will expect schemas expressed using regular tree grammars. By *schema evolution* we mean conducting certain operations upon the existing schema until reaching the desired final state – new version of the schema. A grammar for a schema $\mathcal{S}$ will be denoted $G_{\mathcal{S}}$.

2.1 Preliminary Definitions

First, we briefly introduce the notion of validity of an XML document against an XML schema expressed as a regular tree grammar. For more details we refer to our earlier work (Nečaský and Mlýnková 2010).

Definition 1 (Set of Conforming Documents, Validity). For schema $\mathcal{S}$ the *set of conforming documents* $\mathcal{T}(\mathcal{S})$ equals to the language $\mathcal{L}(G_{\mathcal{S}})$ generated by a grammar $G_{\mathcal{S}}$. We will say that XML document T is *valid* against $\mathcal{S}$ if $T \in \mathcal{T}(\mathcal{S}) = \mathcal{L}(G_{\mathcal{S}})$.

The degree of difference between two versions of a schema can vary greatly – the newer version can only fix some names, change the arrangement of elements or tweak the data types to be more accurate. On the other hand, the new version of the system can bring new attributes and subtrees or change the structure of the documents. We distinguish two special relationships which allow to compare two version of the same schema called *backward* and *forward compatibility*. In general, however, they can be incomparable.

Definition 2 (Backward, Forward Compatibility). $\mathcal{S}'$ is called *backward-compatible* when all documents valid against $\mathcal{S}$ are also valid against $\mathcal{S}'$, that is, $\mathcal{T}(\mathcal{S}) \subseteq \mathcal{T}(\mathcal{S}')$. $\mathcal{S}$ is called *forward compatible* when all documents valid against $\mathcal{S}'$ are also valid against $\mathcal{S}$, that is, $\mathcal{T}(\mathcal{S}') \subseteq \mathcal{T}(\mathcal{S})$.

The motivation for our work is integration and/or modification of components accustomed to the old version of the system. The first part of the problem is deciding whether the changes in the schema may cause problems with these components.

Definition 3 (Invalidated Set of Conforming Documents). We say that the set of conforming documents $\mathcal{T}(\mathcal{S})$ of schema $\mathcal{S}$ was *invalidated in the new version* (or just *invalidated*) if: $\exists T \in \mathcal{T}(\mathcal{S}) : T \notin \mathcal{T}(\mathcal{S}')$. If no such T exists, then $\mathcal{S}'$ is backward compatible.

For an invalidated set of conforming documents, the process of adjusting the invalid documents to the new version is called *revalidation*. Formally:

Definition 4 (Revalidation). A function *revalidate*: $\mathcal{T}(\mathcal{S}) \rightarrow \mathcal{T}(\mathcal{S}')$ is called *revalidation function* iff $\forall T \in \mathcal{T}(\mathcal{S}) \setminus \mathcal{T}(\mathcal{S}')$: *revalidate* $(T) \in \mathcal{T}(\mathcal{S}')$.

2.2 Required Edit Operations

Changes in the evolving schema can be categorized into four groups according to the character of a change (the classification is similar to Nečaský and Mlýnková 2009): *addition* (a new content is added), *removal* (an existing content is removed), *migratory* (elements, attributes or whole subtrees are moved to another location), and *sedentary* (an existing content is modified). An evolution framework should provide means to allow the user to perform operations in the schemas/ documents listed in Table 32.1. Supporting sequence (i), choice (ii), and set (iii) content model means allowing to specify that a content of an element is a sequence of other elements in a given order (i), one of the given options (ii), or a sequence of elements, but in an arbitrary order (iii).

3 Architecture Requirements

Schema editing is just a preliminary step toward a robust evolution framework. In this section, we will study more general features defining its architecture.

Separation of Evolution of Schemas and Documents It is highly useful when schema evolution and document revalidation are separated phases of system evolution. The main advantage is that it allows to perform more operations in one cycle. When the framework creates (or helps to create) a *revalidation script* as an output of the schema evolution phase (and such a script can be then applied on the documents,

Table 32.1 Table of required operations

Add an element/attribute	Addition
Remove an element/attribute	Removal
Rename an element/attribute	Sedentary
Move a leaf element/attribute	Migratory
Convert attributes to elements and vice versa	Migratory
Control occurrences (cardinalities) of elements/attributes	Sedentary
Specify a content model (sequence, choice, set) for contents of elements	Sedentary
Move a subtree in the document	Migratory
Specify data types for values of attributes/text nodes	Sedentary
Specify mixed content	Sedentary

in this scenario, application of the revalidation script acts as *revalidate* function from Definition 4), it can be used to revalidate documents not present in the system in the moment. These may come from other communicating systems that have not adopted the new version yet.

Normalization before Propagation If more than one operation in each evolution cycle is allowed, the sequence of operations should be normalized to eliminate repetitive and redundant operations.

Inheritance The framework should also provide some possibility of complex content definition reuse such as inheritance and composition.

Conceptual Model, Data Semantics Furthermore, the framework should be able to link the constructs in the schema to a conceptual model. If so, also changes in the semantics besides changes in the structure can be correctly identified and revalidated. A change at the schema level can sometimes be propagated to the documents in various ways. For example, when `time-spent` attribute is moved from `Task` element to its parent element `Project` in the schema, its value in document x should be equal to the sum of all the values in `Task` elements in $T \in \mathcal{T}(\mathcal{S})$. Another example is when a value of the attribute in the new version $\mathcal{S}'$ should be obtained from the values of some attributes in the old version $\mathcal{S}$ (again, at the schema level, a new attribute was added, but the propagation to the documents is more complex).

Advanced Content Generation When definitions of new attributes/leaf elements with text content are added to the schema, corresponding attributes/elements need to be created in the documents from T (if the added attributes/elements are not optional). When the respective data cannot be obtained from the revalidated document itself (as in case of the example from the previous paragraph), it must be generated. Prevailing approach to this issue is to let the user specify the content. However, the system can have the required content stored already, only in some other component (e.g., a relational database or other XML document). Framework designed upon a conceptual model can exploit this – for example, by automatically generating an SQL query that retrieves the required data for the revalidated document.

Technical Aspects Finally, there are other, more general aspects that should be taken into consideration, such as whether the framework has a *formal background*, whether it is *independent of the implementation language* (both schema language and revalidation script language), and whether it provides some form of *graphical notation*. In addition, other features may be supported by evolution frameworks, such as namespaces, substitution groups (from XML schema language), etc., but we believe that these are mainly technical tools not required at the abstract level.

4 Approaches to Revalidation

For the goal of determining whether $\mathcal{T}(\mathcal{S})$ was invalidated, the system must recognize and analyze the differences between $\mathcal{S}$ and $\mathcal{S}'$. There are three possible ways to recognize changes:

(a) Only a single change allowed with immediate propagation
(b) Recording of the changes as they are conducted during the design process
(c) Comparing the two versions of the diagram.

We will briefly describe the consequences of choosing the first trivial approach and then discuss in more detail the other two. Either way has both advantages and disadvantages, and the choice between the two significantly influences the capabilities of any revalidation algorithm.

Immediate Propagation The basic approach is to allow only a single change between $\mathcal{S}$ and $\mathcal{S}'$ which is immediately propagated to $\mathcal{T}(\mathcal{S})$. In particular, the tool offers a finite set of evolution primitives, the user selects one, and as the schema is evolved, the documents are revalidated right away. As this approach does not support conducting multiple operations in one evolution cycle, revalidation is not effective (with more operations, the set $\mathcal{T}(\mathcal{S})$ will be revalidated over and over, each time with only a small local change). Moreover, tools that choose this approach offer a limited set of primitives, which results in the necessity of using more primitives for one logical operation from Table 32.1 (e.g., “rename element” requires “remove element” and “add element”). This results in unwanted loss of data. Since each primitive is propagated immediately, such an approach can never meet the requirement for separation of the schema evolution and document revalidation phases (they are interleaved).

Recording Changes An evolution system that uses the recording technique usually provides some kind of a command that initiates the recording, and after issuing this command, all operations carried out by a user over the schema are recorded. The user starts with schema $\mathcal{S}$ and initiates recording. All conducted operations leading to $\mathcal{S}'$ are recorded until the user is satisfied with the new schema. When the desired schema $\mathcal{S}'$ is reached, the user finishes recording, and the system has all the information about the changes made – the sequence of performed operations.

When the recording is finished, the system can normalize the sequence, for example, by eliminating operations that cancel each other or by replacing groups of operations by other groups that lead to the same result but in a more straightforward way. The normalizing rules must be defined in the system.

Here is the outline of the key characteristics of recording approach:

- The recorded set should be normalized to eliminate redundancies (repeated changes in the same location, etc.).
- Once the evolution process is started, the old version $\mathcal{S}$ cannot be easily changed.
- A user may want to interrupt his/her work at some point and continue in another session. The sequence of recorded changes would have to be stored and recording resumed later.
- When the user wants to retrieve the sequence for reverse process, (s)he will have to either start with the new version $\mathcal{S}'$ and record the operations needed to go back to the old version $\mathcal{S}$ again, or the system will have to be able to create an inverse sequence for each sequence of operations.
- When the evolved schema comes from an outer source, the sequence of operation changes cannot be retrieved directly; the user must start with his/her old version $\mathcal{S}$ and manually adjust it to match the new schema $\mathcal{S}'$.
- Recorded set provides enough information to propagate changes in the schema to the documents; there are no ambiguities or possible misinterpretations of operations.

Schema Comparison The last approach is to base the change detection on comparison of two versions of the schema. The user can work with both schemas independently until (s)he is satisfied with them. The change detection algorithm then takes the two schemas as an input and compares them. The result of the comparison is a list of differences between the schemas.

The characteristics of the comparison approach are as follows:

- There is no need to look for redundancies; the set of changes is always minimal.
- Both old version $\mathcal{S}$ and new version $\mathcal{S}'$ can be edited without limitations. The system may contain an arbitrary amount of versions, any pair of versions can be compared. This can increase efficiency, for example, when a document valid against version $\mathcal{S}$ is revalidated against $\mathcal{S}''$ – the script obtained from comparing $\mathcal{S}$ and $\mathcal{S}''$ can be more efficient than the concatenation of the scripts for transformations $\mathcal{S} \to \mathcal{S}'$ and $\mathcal{S}' \to \mathcal{S}''$.
- The process of schema evolution can be arbitrarily stopped and resumed.
- The reversed operation can be easily handled by the same algorithm, only with the two schemas on the input swapped.
- A schema from an outer source can be imported into the system and serves as an input to the change detection algorithm.
- Without additional information, some type of changes cannot be distinguished (rename vs. add and remove, move vs. add and remove); methods for mapping discovery are necessary.

If we compare features of recording and comparison approaches, the last item in the former list speaks in favor of recording approaches, but the assumption requires a strict discipline from the user. If (s)he moves an attribute from element E_1 to element E_2, the change will be processed correctly during revalidation (the value will be copied). But if (s)he removes the attribute from E_1 and adds the same attribute to E_2, the value will be lost although the user's intent was to move the attribute. If the user decides to correct his/her operation, (s)he has to either undo the operation (and all operations conducted afterward) or modify the recorded sequence which is nontrivial in a general case and providing such option would significantly raise the complexity of the framework.

The comparison approach is more versatile and flexible than the recording approach, but it must resolve ambiguities that may emerge (mentioned in the last item of the list). The final choice of interpretation of the changes may be left up to the user or decided by a heuristic.

5 Support in Existing Evolution Frameworks

In this section, we examine the existing evolution frameworks and their capabilities to fulfill the requirements stated in Sect. 32.2.

X-Evolution System *X-Evolution* (Guerrini et al. 2007) is an example of a system built upon graphical editor for creating schemas in the XML schema language. It uses *immediate propagation*. Backward compatibility is decided particularly by marking each evolution primitive as backward-(in)compatible. The set of operations is limited to insertions and deletions, which need to be used for all more complex operations (like move, adding a wrapping element, etc.). A part of the set of available operations is rather technical (switching between a local and global definition of a type, etc.). It considers only elements without attributes and recognizes choice/sequence/set content models. *X-Evolution* does not use any conceptual model for the schemas.

In Guerrini and Mesiti, further describe the *XSchemaUpdate language*. Using statements in this language, a new content can be created with non-default values.

The system is implemented as a schema visualization tool, where the user can select and construct which she/he wants to evolve and then select the desired evolution primitive from the set of available primitives. Revalidation of $\mathcal{T}(\mathcal{S})$ is triggered right after an evolution primitive is selected.

XML Evolution Management (XEM) (Su et al. 2002) is another *immediate propagation* approach to manage schema evolution, but this time, DTD is used as a schema language[1]. It deals both with changes in DTD and XML documents. Both DTD and instance XML documents are represented in the system as directed acyclic graphs

[1] Not to be confused with W3C *XQuery Update Facility* W3C

and the evolution primitives are defined as operations on these graphs. For each DTD altering primitive, a resulting data change is defined in terms of the primitives altering instance documents.

The set of proposed primitives is proven to be sound and complete in the terms of being able to transform any DTD to any other DTD; however, there are primitives for addition and removal, but none for moving or renaming. Any change in DTD can thus be expressed via the proposed primitives, but when these are propagated to the valid XML documents, they lead to removing a significant part of the XML document and recreating it again. This applies even for small, local changes. For example when an element needs to be renamed, it must be removed first (which can only be done after its subtree is removed) and then added under new name. If the root element is removed, the whole XML document is first deleted and its structure recreated again. The same holds for moving content. In this process the structure is created properly by the algorithm, but the data is lost.

Since *XEM* works with DTDs, its capabilities are restricted (e.g., no support for set content model, limited support for cardinalities). Again, as in case of *X-Evolution*, the conceptual model is not utilized.

XEM-Tool, which implements XEM, uses object-oriented mapping for manipulating DTDs and instance XML documents. The input DTD is processed by *DTD Manager* component which builds a DTD graph and then generates Java classes (`ELEMENT` definitions are translated to classes, `ATTRIBUTE` definitions to properties, parent-child relationships are stored in a *children* vector of each class). Execution of a DTD change primitive triggers regenerating the Java source files and their recompilation.

CoDEX system (Conceptual Design and Evolution of XML Schemas) (Klettke 2007) is an example of a schema evolution framework using the *recording approach*. The changes made in the visualization of the schema are logged, and when the evolution process is finished, the resulting changes are performed in the XSD.[2] Documents associated with the evolved schema are then updated according to the XML schema evolution steps.

The recorded design steps are minimized and normalized due to the fact that the design process conducted by the user is not often straightforward. This is achieved via built-in rules for combining and reducing the design steps. When the evolved schema is not backward compatible, XML update operations are generated.

In contrast to the previous approaches, *CoDEX* allows the user to make a sequence of changes in each evolution cycle (both *XEM* and *X-Evolution* allow only a single primitive). Moves and renames are supported, but the tool does not support multiplicity. Especially migratory operations (moving elements, attributes, and subtrees from one location in the document to another) can become complex when multiplicity is introduced (e.g., what should happen when an attribute is moved from element E_1 to E_2 in the schema, when there are multiple instances of both elements in the revalidated document?).

[2] XML schema definition

The approach does not use conceptual model, and the authors themselves give an example why this can lead to unexpected results: the user moves element `address` from element `owner` to element `producer`. The structure of both the elements `address` is the same (they would probably refer to the same type), but semantically the meaning is different (the address of the producer is not the same as the address of the owner), and moving the contents of `address` from element `owner` to element `producer` would create valid, yet semantically incorrect document.

Evolution using UML Class Diagrams The *Unified Modeling Language* (UML) (Object Management Group 2007) is nowadays widely used for *platform-independent modeling* of system infrastructure and data. XML schema derivation from UML class diagrams is supported by commercial tools (e.g., Sparx Systems: Enterprise Architect). The schema derivation can be fully automatic or adjusted by applying *stereotypes* provided in a UML profile.

Adding support for schema evolution (again using the *immediate propagation*) is proposed in Domínguez et al. [2005]. The authors utilize UML class diagrams for the conceptual model and define straightforward mapping between building blocks of class diagrams and XML schema constructs; the translation is fully automatic. Without any means to alter the mapping and thus tailor the resulting XSD, this approach gives the user very limited control over the result. The possibility of having several different XSDs sharing the same concepts mapped to the same conceptual diagram is not supported.

The evolution operations are conducted upon the class diagram instead upon the (visualized) XML schema. The set of evolution primitives consists of basic operations for working with UML diagrams (i.e., addition or deletion of classes, attributes, and associations).

The propagation of changes proceeds in two steps. First, the *mapping rules* keeping correspondence between UML and XML constructs are updated to reflect the changes. Deduced changes in the set of translation rules serve as an input for the next step – propagation of changes to the XSDs and XML documents. Schema evolution (propagation of changes to the XSDs) and function *revalidate* (propagation to documents) are both implemented using XSL (Kay 2007).

XSLT Visualization Authors in Kwietniewski et al. [2010] use the *schema comparison* technique. The source and target schema are represented as XSDs, the schemas themselves are evolved outside the system, and only data revalidation is considered. The algorithm for mapping discovery starts at roots of the source and target XSD and continues recursively (the routine attempts to match sets of children of two already matched nodes). The mapping is "best-effort" and partial. When the elements of the source schema $\mathcal{S}$ and target schema $\mathcal{S}'$ are mapped, an XSLT script is generated on the basis of this mapping. Since the mapping is, in a general case, only partial, the produced XSLT script does not guarantee correct revalidation (the output document will not be valid against the target schema). The system expects the user to adjust the XSLT script manually with the aid of XSLT script

Table 32.2 Operations supported/requirements met by *X-Evolution*, *XEM*, *CoDEX*, evolution using UML, and XSLT visualization approach, respectively

X-Evolution	*XEM*	*CoDEX*	*viaUML*	*XSLTviz*
Revalidation approach (immediate propagation, recording, comparison)				
IP	IP	REC	IP	COMP
Operations				
Add element/attribute				
Elements only	✓	✓	Elements only	✓
Remove element/attribute				
Elements only	✓	✓	Elements only	✓
Rename element/attribute				
Elements only	–	✓	Elements only	✓
Move leaf element/attribute				
–	–	Without cardinalities	?	Elements only
Convert attributes to elements and vice versa				
–	–	–	–	?
Support for cardinalities				
✓	DTD limitations	–	✓	–
Support for set/sequence/choice content model				
✓	Set not supported	✓	✓	✓
Move subtree				
–	–	✓	?	✓
Specify data types for values of attributes/text nodes				
–	–	?	–	–
Mixed content support				
–	–	–	–	–
Other features				
Multiple operations in one cycle/separated evolution of schemas and docs				
–	–	✓	–	✓
Normalized sequence of operations				
–	–	✓	–	Not needed
Inheritance				
–	–	–	✓	?
Advanced content generation				
Partly	–	–	–	–
Ability to recognize semantics of operations with elements/attributes				
–	–	–	–	–
Formal background				
✓	✓	✓	✓	–
Language independent				
✓	–	–	✓	–
Graphical notation				
✓	–	✓	✓	✓

visualization. The generated XSLT script, which conducts document revalidation, follows the structure of the target schema and the XSLT visualization together with the visualization of $\mathcal{S}$, and $\mathcal{S}'$ helps the user with the process.

1:1 mapping between the elements of the source and target schema is not sufficient to encompass some specific types of evolution operations. For example, splitting one attribute into several new attributes is impossible to describe – the algorithm excludes pairs of already mapped elements, so 1:N mapping is never defined.

Even a successful finding of a mapping between two elements does not always allow straightforward revalidation. For example, moving an attribute from element e_c to its parent element e_p does not allow a simple copying when the schema allows e_p to have multiple subelements e_c.

5.1 Support for Operations: Summary

Table 32.2 summarizes how the aforementioned systems meet the requirements for an evolution framework. It can be seen that none of the examined systems fully complies to the requirements. The requirement for recognizing semantics is met by none of them in particular (and is acknowledged as lacking by the authors of *CoDEX*).

6 Conclusion

In this chapter, we have compared and evaluated several approaches to XML document revalidation according to given comparison criteria. We identified several features for which the support in the existing approaches is not sufficient.

The whole problem of XML data revalidation is nontrivial and deserves proper support from the tools. However, none of the existing frameworks provides full support for the operations we consider necessary. The major open problems in the area of schema evolution/document revalidation we see in:

- The lack of support of conceptual models and semantics of revalidation
- Missing support for integrity constraints definition and utilizing constraints in revalidation
- Further automation of the whole revalidation process when new data is needed (aiding the designer with adding new data, suggesting queries to retrieve the data)

Also, none of the studied frameworks provides support for the newer XML schema languages like RELAX NG or Schematron.

References

Bray T, Paoli J, Sperberg-McQueen CM, Maler E, Yergeau F (2008) Extensible markup language (XML) 1.0, 5th edn. W3C. http://www.w3.org/TR/REC-xml/

Clark J, Makoto M (2001) RELAX NG specification, Oasis. http://www.oasis-open.org/committees/relax-ng/spec-20011203.html

Domínguez E, Lloret J, Rubio AL, Zapata MA (2005) Evolving XML schemas and documents using UML class diagrams. In: DEXA, Lecture notes in computer science, vol 3588. Springer, Berlin, pp 343–352

Guerrini G, Mesiti M, XML schema evolution and versioning: current approaches and future trends

Guerrini G, Mesiti M, Sorrenti MA (2007) XML schema evolution: incremental validation and efficient document adaptation. In: XSym. Lecture notes in computer science, vol 4704. Springer, Berlin, pp 92–106

ISO (2005) Information technology document schema definition languages (DSDL) part 3: rule-based validation schematron. ISO/IEC 19757-3

Kay M (2007) XSL transformations (XSLT) version 2.0. W3C. http://www.w3.org/TR/xslt20/

Klettke M (2007) Conceptual XML schema evolution – the CoDEX approach for design and redesign. In: Workshop proceedings datenbanksysteme in business, technologie und web (BTW 2007), Aachen, pp 53–63

Kwietniewski M, Gryz J, Hazlewood S, Van Run P (2010) Transforming XML documents as schemas evolve. Proc VLDB Endow 3:1577–1580

Murata M, Lee D, Mani M, Kawaguchi K (2005) Taxonomy of XML schema languages using formal language theory. ACM Trans Internet Technol 5(4):660–704

Nečaský M, Mlýnková I (2009) Five-level multi-application schema evolution. In: DATESO, pp 90–104

Nečaský M, Mlýnková I (2010) When conceptual model meets grammar: a formal approach to semi-structured data modeling. In: WISE 2010. Lecture notes in computer science, vol 6488. Springer, Berlin, pp 279–293

Object Management Group (2007) UML infrastructure specification 2.1.2. http://www.omg.org/spec/UML/2.1.2/Infrastructure/PDF/

Sparx Systems: Enterprise Architect. http://www.sparxsystems.com.au/products/ea/

Sperberg-McQueen CM, Bray T, Paoli J (2000) Document type declaration

Su H, Kramer DK, Rundensteiner EA (2002) XEM: XML evolution management. Technical report WPI-CS-TR-02-09

Thompson HS, Beech D, Maloney M, Mendelsohn N (2004) XML schema part 1: structures, 2nd edn. W3C. http://www.w3.org/TR/xmlschema-1/

W3C, XQuery update facility 1.0 specification. http://www.w3.org/TR/xquery-update-10/

Chapter 33
Tool Support for Checking Consistency of UML Model

Ruta Dubauskaite and Olegas Vasilecas

1 Introduction

Sometimes, the models of different aspects (processes, states, structure, etc.) are not interrelated, and even more, contradictory information can be provided in them. For example, it is possible that elements created in one model are not used in another model. Expression of an IS through various models is related to the problem of consistency ensuring of different models. Consistency means that the structures, features, and elements that appear in one model are compatible and in alignment with the content of other models (Rozanski and Woods 2005).

The problem of consistency checking of IS model is important when several analysts and designers model the same system; they can use different terms for the same concept. If the information system is large and complex, the risk of consistency conflict in the models is bigger. Therefore, the question of ensuring consistency is even more relevant.

IS models, expressed in UML, are chosen for detail analysis because UML is likely the most popular modelling language (Silingas and Butleris 2009), and there are many modelling tools supporting UML (Shen et al. 2002). The second reason why UML was chosen is because it was developed by OMG (Object Management Group), which also introduces MDA (Model-Driven Architecture) (Lucas et al. 2009). Consistency of UML model is especially important in MDA, for automatic transformation initial model to specific model and finally code generation tasks (Rozanski and Woods 2005; Berkenkötter 2008). Finally, UML is the most popular semiformal modelling language (Berkenkötter 2008). It is considered as the standard for the object-oriented modelling (Mokhati et al. 2007), and usually, object-oriented methods are used for a detailed analysis and following implementation.

R. Dubauskaite (✉) • O. Vasilecas
Vilnius Gediminas Technical University, Sauletkio al. 11, Vilnius, Lithuania
e-mail: ruta.dubauskaite@vgtu.lt; olegas.vasilecas@vgtu.lt

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_33,

The rest of this chapter is structured as follows: Further sections introduce approaches of ensuring UML model consistency, describe checking consistency using UML CASE (computer-aided software engineering) tools and present the suggested approach for consistency checking of UML model.

2 Related Works

2.1 Approaches of Ensuring UML Model Consistency

The analysed approaches of checking UML model consistency can be divided to two groups. The first group of approaches is based on constraints. Constraints are defined for UML metamodel, and UML models are checked according to these constraints. The authors of papers (Shen et al. 2002; Berkenkötter 2008; Chen and Motet 2009; Egyed 2007) suggest defining constraints for every different aspect model. But constraints among different models are not defined. The Eyged approach (Egyed 2007) uses constraints on relationships of different aspect models. But the defined consistency rules are hard coded to UML analyser tool and cannot be analysed. The second group of approaches is based on formal models (Mokhati et al. 2007; Van Der Straeten et al. 2003; Pakalnickiene and Nemuraite 2007). Authors of these approaches propose transformation of semiformal UML models to formal models. In this case, it is suggested to detect consistency conflicts using inference mechanism of formal language. Despite being precise of formal models, they are less understandable in comparison with semiformal languages, which can be presented using diagrams (Vavpotič and Bajec 2009).

2.2 Analysis of UML CASE Tools

In this section, we analyse several CASE tools that can be used for developing UML model. The examples of commercial UML tools are *Gentleware Poseidon for UML* (can be found in www.gentleware.com/download.html), *IBM Rational Rose* (can be found in www-01.ibm.com/software/rational/), *Sybase PowerDesigner* (can be found in www.sybase.com/products/modelingdevelopment/powerdesigner), *NoMagic MagicDraw* (can be found in www.magicdraw.com/), *Microsoft Visio* (can be found in office.microsoft.com/en-us/visio/), IBM Rational *System Architect* (can be found in www-01.ibm.com/software/awdtools/systemarchitect/) and others. While *ArgoUML* (can be found in argouml.tigris.org/), *Open ModelSphere* (can be found in www.modelsphere.org/), *StarUML* (can be found in http://staruml.sourceforge.net/en/) and *Umbrello UML Modeller* (can be found in http://uml.sourceforge.net/) are open source UML tools.

The usage of these tools allows us to create and modify the model of information system; it is especially important for large systems. The question of quality, especially of consistency of IS model, is also very important.

This chapter presents the results of analysis *MagicDraw UML*, *PowerDesigner*, *Poseidon for UML*, *Rational System Architect* and *Microsoft Visio* tools from perspectives of checking consistency of IS model and extension of these tools. Commercial tools were chosen for detail analysis because they are widely used; besides, they usually have more facilities compared with open source tools. Our decision also depended on economical factors (university has licences of *MagicDraw* and *PowerDesigner*) and personal factors (experience of working with the tools). *Microsoft Visio* tool is incorporated to widespread *Microsoft Office* package; therefore, it is worth attention. *Poseidon for UML* and *IBM Rational System Architect* are chosen accidentally, and trial versions are explored.

2.2.1 Magic Draw UML 15.0

MagicDraw has facility to check the produced models according validation rules. Validation rule captures some condition, which must be checked against the model. Validation rules are specified as invariant constraints in the model.

There are three types of constraints that *MagicDraw* can evaluate:

- Classifier level constraints
- Constraints on metaclasses
- Constraints on stereotypes

Classifier level constraints are placed on the classes, data types, and other classifiers of the model. These constraints are evaluated on all the instances of these classifiers. The example of this classifier constraint is *Limit of Credit cannot be bigger than 500*.

Constraints on metaclasses are placed on one of the classes in the UML *Standard Profile:UML2 Metamodel*. This constraint is evaluated on all the model elements of that kind. For example, if constraint is placed on *Actor* metaclass, then this constraint applies to all the actor elements in the model. An example of rule (specified in OCL2.0), which mandates that *all actor names in the model must be capitalised* is provided below (MagicDraw 2008a):

```
context Actor inv capitalize:
let startwith:String = name.substring(1,1) in
startwith.toUpper() = startwith
```

The third group of constraints is constraints on stereotypes. Stereotype is specific metaclasses on existing UML metamodel element (OMG 2007b). Constraints on stereotypes are applied to all the model elements that have this stereotype applied to them.

Constraints can be classified not only to types but also they are grouped according to application domain. *MagicDraw UML 15.0* has eight suites of rules for checking:

- Completeness of model
- Correctness of model
- XML, DDL schemes, Java, C++ specific models and other domains

Validation suites, available for validating, depend on what profiles your model includes. Profile is package of UML stereotypes, tagged values and constraints for modelling concrete aspect of system (OMG 2007b), for example, graphic user interface. In this chapter, the analysis of completes and correctness suites is presented in detail, because they are the most related with our research. Both completeness and correctness suite include 13 rules; they are in *UML Standard Profile*.

Completeness suite has a collection of rules, which check that model is more or less finished; there are no gaps; and elements have the essential information fields filled in. Examples of completeness rules are the following:

- *No initial state should have incoming transitions.*
- *Instances should have classifier specified.*
- *Actors should be named.*
- *Call message should have operation assigned.*
- *Specify parameter types for operation.*

Completeness of model depends on correctness of model and vice versa. Correctness suite has a collection of rules, checking for common mistakes while modelling in UML2. The examples of these rules, realised in *MagicDraw* are the following:

- *Should be one constrained element for OCL2.0 constraints.*
- *All the features in the interface should be public.*
- *Leaf element should not be abstract.*
- *Lower multiplicity value should be lower than upper multiplicity value.*
- *If at least one operation of classifier is abstract, classifier should be abstract.*

The automatic validation process can be activated by command *Validation* from *Analysis* menu.

The previous analysed rules are provided as OCL 2.0 expressions in *MagicDraw*. In general, *MagicDraw* supports 2 languages for expressions that can be evaluated:

- OCL2.0 is used for validation rules, specified in OCL language. Only OCL invariants are evaluable in MagicDraw. Other types of OCL constraints (def, init, derive, pre, post, body) can be modelled for documentation purposes.
- Binary is used for the more advanced expressions, which are not easily expressed in OCL. These expressions are written in *Java*, compiled and specified

in the *MagicDraw* classpath. Then these expressions can be specified as validation rule expression (MagicDraw 2008b).

MagicDraw tool can be extended with new rules, expressed in OCL or *Java* code. When rules are OCL invariants, then package with constraints might be exported as module and used by any other project of IS modelling. If the rules are expressed in *Java* code, it is necessary to create a plug-in for *MagicDraw*, which can be also reused for checking models of other systems.

The analysis of *MagicDraw UML 15.0* from the perspective of checking IS model consistency shows tools have 26 rules for checking completeness and correctness of IS model. The most of rules check one aspect model.

2.2.2 PowerDesigner 15.0

There are 34 OOM (object-oriented model) specific type checks available by default in *PowerDesigner* tool. The examples of types of constraints are use case, class, message, state, event, activity and association.

Examples of constraints implemented in *PowerDesigner* tool are the following:

- *Name of class has to be unique.*
- *Actor/use case has to be linked to any object.*
- *Data type has to be assigned to attribute.*
- *Circular dependency cannot be among packages.*

New validation rules can be defined by user using graphic user interface. However, the rules can be used only for the purpose of documentation. It means that rules are not included to automatic process of model validation, which is activated by command *Check Model* from *Tools* menu.

The rules that are documented in model are not used in the process of model checking. Therefore, if we want to include new rules to process of model validation, it is necessary to extend the *PowerDesigner* tool with new profile (Sybase 2008b). Custom checks (profile) are model checks, written in *VBScript*. Custom checks are listed with standard model checks in the *Check Model Parameters* dialogue box. A more detailed analysis about *MagicDraw* and *PowerDesigner* is provided in the article (Dubauskaitė and Vasilecas 2009b).

2.2.3 Poseidon for UML 8.0 and Visio 2003

Poseidon for UML 8.0 and *Visio 2003* provide elements, defined in UML metamodel, for creating model. But most of the constraints in UML specification defined (OMG 2007b) are not implemented in the tool. There is no ability to check consistency of information system model according to explicitly defined constraints on one aspect model or on different aspect models. *Poseidon for UML* can be extended with new features developing plug-in, written in *Java* language (Boger et al. 2006), while *Microsoft Visio* tool has possibility to include new macros, written in *Visual Basic Language* or include new plug-in (sometimes called add-in or add-on), written in .NET language.

2.2.4 Rational Software Architect 8.0.2

Rational Software Architect has the facility to check that it complies with the defined constraints (IBM 2011a). The validation function is called by right clicking the model and then clicking *Validate*. The user can identify the rules that cause errors, warnings, or reminders when he validates a UML model or diagram (IBM 2011b). All implemented constraints are hard coded, and they are not shown to the designer. According to documentation of *Rational Software Architect 8.0*, this version of the tool is extended with two validation rules for sequence and class diagrams:

- *Private operation call is not allowed.*
- *Message signature operation should be contained in target lifeline classifier* (IBM 2010).

Rational Software Architect tool can be extended with new rules creating a plug-in (Zwanziger et al. 2011). The two tool-supported languages are OCL (Object Constraint Language) and Java (Miic 2005).

2.2.5 Comparative Analysis of CASE Tools

Table 33.1 shows a summary of the analysed *NoMagic MagicDraw UML*, *Sybase PowerDesigner*, *Poseidon for UML*, *IBM Rational System Architect* and *Microsoft Visio* tools. Criteria of comparison are the following:

- Model in conformity with the metamodel. Possible values are "+" (almost conform) and "partially" conform. If model is in conformity with UML, the metamodel is checked according to one rule: *Name of class has to be unique*. If the tool does not allow creating class with the same name in the model, then it is assumed that the model almost conforms to the metamodel, "almost" because it is not checked if all constraints defined in the metamodel are implemented in the tool. If the tool allows creating two classes with the same name, it is assumed that the model partially conforms to the metamodel, "partially" because tool does not implement all the constraints defined in the specification of UML but provide meta-elements defined in the metamodel.
- Constraints (consistency rules) for one aspect model. Constraints are defined at metamodel level for one aspect model, for example, for class diagram.
- Consistency rules among two and more different aspect models. Constraints are defined on relationship of different aspect models, for example, between class diagram and sequence diagram.

Table 33.1 The comparison of CASE tools from checking consistency viewpoint

Compared CASE tools / Comparison criteria	MagicDraw UML 15.0	Power Designer 15.0	Poseidon for UML 8.0	Rational Software Architect 8.0.2	Visio 2003
1. Model in conformity with metamodel	+	+	Partially	Partially	Partially
2. Constraints (consistency rules) for one aspect model	+	+	–	+	–
3. Consistency rules among two and more different aspect models	+	–	–	+	–
4. Language for extension of tool, for expressing/implementation rules	OCL, Java	Visual Basic	Java	OCL, Java	Visual Basic (for macros), . NET (for plug-in)
5. Technique of tool extension with new rules of consistency checking	Module, plug-in	Plug-in	Plug-in	Plug-in	Macros, plug-in

- Language for expressing/implementation rules. The language can be used for expressing extension of tool and for adding constraints of one aspect model or on relationship of different aspect models.
- Technique of tool extension with new rules of consistency checking. Examples are developing module or plug-in or macros or using other techniques for the extension of the tool with new rules.

Despite of the existence of many tools, it is not easy to develop a model that conforms to UML metamodel. Besides, not all available tools have the facility to check consistency of IS model, and almost all defined constraints are for one aspect model.

2.3 The Results of Analysis

The analysis of related work shows that the approaches solve inconsistency of model problems in some extent. Formal models are often too difficult to understand to be used in practice. Semiformal UML models are widely used, but their constraints are proposed only for one model, and relationships among models are not defined. The Eyged approach (Egyed 2007) includes constraints on relationships of different aspect models, but the defined consistency rules are hard coded to CASE tool and are not expressed in formal language, which would be independent from platform-specific features.

MagicDraw UML 15.0 and Rational Software Architect 8.0.2 have constraints on relationships of different aspect models, expressed in formal language. But the existence of the CASE tools does not fill the gap of the approach of ensuring IS model consistency which would not be dependent on concrete language or tool.

The authors of this chapter suggest extending the approach based on constraints of UML metamodel by adding consistency rules among different aspect models. Consistency rule means constraint on relationship of different aspect models. It is suggested consistency rules to express in OCL in order the approach would be more general and more applicable in various tools.

The following section presents the suggested approach in detail.

3 The Suggested Approach of Ensuring UML Model Consistency

The novelty of the work lies in the fact that we suggest a new method for ensuring consistency of information system model:

- We proposed to check consistency of original semiformal UML model.
- The approach includes both constraints (consistency rules) on relationships of different aspect models and constraint on different aspect models.
- These consistency rules are expressed explicitly in OCL.
- Consistency rules are defined for UML metamodel.

The structure and relationships of elements of the proposed approach are shown in Fig. 33.1 using UML class diagram.

Consistency of IS model, expressed in UML, is checked according to the defined consistency rules. Consistency rules describe conditions that all UML models must satisfy them to be considered valid (Egyed 2007). Consistency rules, expressed in OCL, constrain every aspect model and relationships of different aspect models. Aspect model is part of IS model. Consistency rules are defined for UML metamodel. IS model also conforms to UML metamodel. Diagrams that visualises all or part of IS model are based on UML metamodel too. Diagrams are included to our suggested approach of IS consistency because the developer often models information system using diagrams, but in order for diagrams to present unambiguous aspects of the IS system, first, the IS model should be consistent.

The process of checking the IS model consistency is presented by authors of this paper (Dubauskaite and Vasilecas 2009a). The main activities are checking model according consistency rules and appending list of violations of consistency. The removal of detected consistency conflicts can allow improvement of the consistency of the UML model.

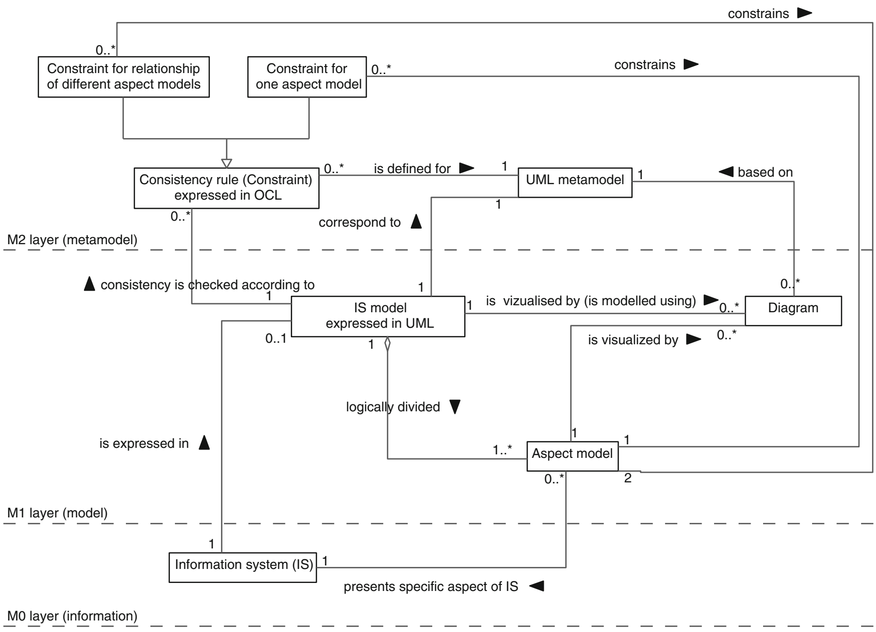


Fig. 33.1 Approach of ensuring consistency based on rules

4 Experiment

A simple experiment is presented in this section to illustrate usage of suggested approach and demonstrate the functionality of created software prototype.

After analysis of UML metamodel specification and specific information system models, we come to a conclusion that it is necessary to define operation, which determines movement of object from one state to another state. Operation is defined in class model. Therefore, consistency rule ID1 associates two different aspect models: class and state models.

Consistency rule ID1 requires defining transition of states by specifying operation, in which execution causes the changes of state. Only one suggested consistency rule is presented in this chapter because of space limitations. More information about consistency rules and their implementation is provided in our paper (Vasilecas et al. 2011).

Figure 33.2 below shows elements of UML metamodel for IS *Class* and *Protocol State Machines* models that are associated by consistency rule ID 1. Dashed lines with arrows present relationships of UML metamodel and IS model. Elements of different aspect models associated by consistency rule 1 are shown using dashed line without arrows. According to UML metamodel, which part is in Fig. 33.2, operation is not mandatory for protocol transition. But in practical situations, it is needed to know what operation execution causes the transition of states.

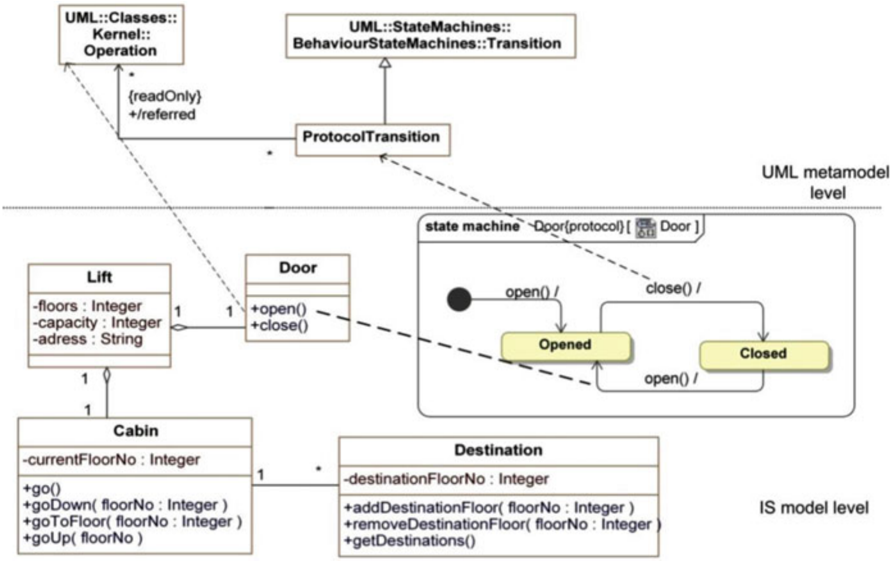


Fig. 33.2 Fragments of UML metamodel for IS static structure (class) and behaviour (protocol state machine) models

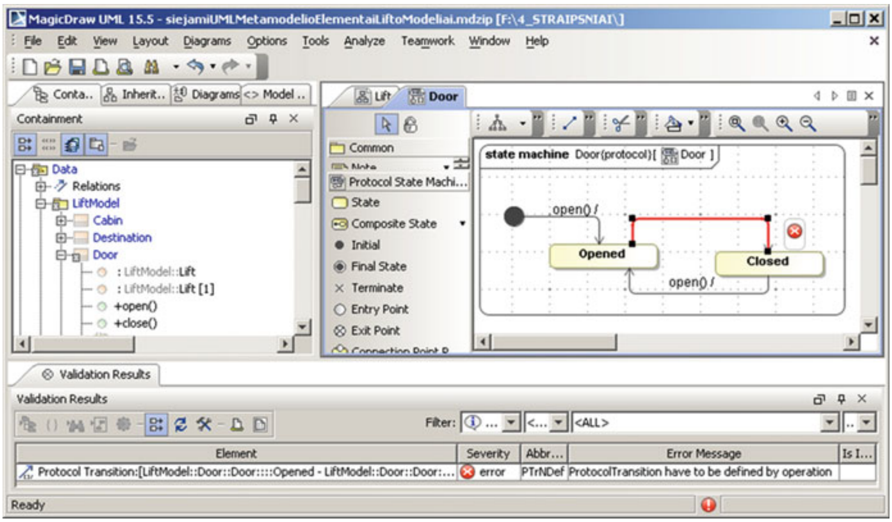


Fig. 33.3 Checking IS model using software prototype which implements our approach of ensuring consistency

The suggested consistency rules were implemented in MagicDraw UML tool. The usage of implemented prototype of consistency module of MagicDraw UML tool is presented in Fig. 33.3.

UML model is validated according to every consistency rule. If consistency conflict is detected, then list of consistency violations is appended with message of error or warning. Left column of Fig. 33.3 provides UML model, developed using MagicDraw UML 15.5 tool, and the right column of Fig. 33.3 visualises the state model by protocol states diagram. The diagram represents possible states of a class Door. Door is a part of lift business system. Accurate modelling of business system is necessary in order software, which is developed or generated automatically according to model and manages business system correctly. The process of checking of business model, expressed using UML, is executed automatically (activated by command validate model). The detected consistency conflicts are shown in the bottom of right column in validation results section of Fig. 33.3.

The last step of ensuring consistency of UML model is modifying of IS model according to detected consistency conflicts.

If method *close* of class *Door* is added to transition of *Door* protocol states model, then consistency of lift system model would be improved.

More details about our suggested approach, consistency rules, and prototype of software that implement the method of consistency checking are provided in (VeTIS 2009).

5 Conclusions

The analysis of related work demonstrates that issue of inconsistency of UML model is still open and relevant. UML model consistency can be checked using constraints for one aspect model. Another group of approaches uses inference mechanism of formal language and includes constraints both on every aspect model and on relationships of different aspect models. But formal models are often too complex to be used in practice. The Eyged approach (Egyed 2007) uses constraints on relationships of different aspect models, but they are hard coded to UML analyser tool.

The research of UML CASE tools shows that several tools have the facility to check consistency of one aspect model. Two of five analysed tools used constraints on relationships on different aspect models. But only MagicDraw UML tool allows reviewing of the constraints on one aspect model and constraints on relationships.

Based on performed analysis, we concluded that it is meaningful to check consistency of UML model and define constraints both on different aspect models and among models. It is also suggested to define constraints at metamodel level and express in OCL. Checking consistency of original UML model allows do not perform additional task of UML model transformation to formal model; in such way, time for detecting consistency conflicts is shortened. Consistency rules expressed in OCL and defined for metamodel are unambiguous, more general and more applicable in various tools.

The experiment shows that the suggested approach is able to detect inconsistencies automatically and in such way makes easier work for the designer.

Besides detected and fixed consistency conflicts in earlier phases of IS life cycle, it allows to reduce cost of IS development.

References

Berkenkötter K (2008) Reliable UML models and profiles. Electron Notes Theor Comput Sci (ENTCS) 217:203–220

Boger M, Graham E, Köster M (2006) Poseidon for UML: user guide. URL: http://cs.adelaide.edu.au/docs/PoseidonUsersGuide.pdf. 13 Apr 2011

Chen Z, Motet G (2009) A language-theoretic view on guidelines and consistency rules of UML. In: Model-driven architecture foundations and applications: LNCS 5562. Springer, Berlin, pp 66–81

Dubauskaitė R, Vasilecas O (2009a) Ensuring models consistency in the OMT, Booch, and OOSE object-oriented methods, information sciences, Lithuania, vol 50. Vilnius University Publishing House, Vilnius, pp 160–167. (CEEOL)

Dubauskaitė R, Vasilecas O (2009b) UML taisyklių modelių darnos tikrinimo galimybės, naudojant MagicDraw UML ir PowerDesigner įrankius (Checking consistency of UML models using MagicDraw UML and PowerDesigner Tools). Informatika: 12-osios Lietuvos jaunųjų mokslininkų konferencijos "Mokslas – Lietuvos ateitis" pranešimų rinkinys (Proceedings of the 12th conference of Lithuanian's researchers) (in press)

Egyed A (2007) Fixing inconsistencies in UML design models. In: Proceedings of the 29th international conference on software engineering (ICSE), Minneapolis, USA. IEEE Computer Society, New York, pp 292–301

IBM (2010) Rational Software Architect – what's new in the 8.0 releases. URL: http://www.ibm.com/developerworks/wikis/display/RSA/RSA+v8+-+What%27s+new. 19 Apr 2011

IBM (2011a) Validating UML models and diagrams [disc]. In: Rational Software Architect 8.0.2 help system

IBM (2011b). Identifying validation rules for UML models [disc]. In: Rational Software Architect 8.0.2 help system

Lucas FJ, Molina F, Toval A (2009) A systematic review of UML model consistency management. Inf Softw Technol 51:1631–1645

MagicDraw (2008a). MagicDraw architecture made simple: user's manual [disc]. In: MagicDraw UML 15.0 help system

MagicDraw (2008b) MagicDraw architecture made simple: OPEN API user's guide [diskas]. iš MagicDraw UML 15.0 vartotojų vadovai

Miic D (2005) Authoring UML profiles using Rational Software Architect and Rational Software Modeler: technical documentation. URL: http://www.ibm.com/developerworks/rational/library/05/0906_dusko/. 19 Apr 2011

Mokhati F, Gagnon P, Badri M (2007) Verifying UML diagrams with model checking: a rewriting logic based approach. In: Proceedings of the seventh international conference on quality software, USA, pp 356–362

OMG (2007b) OMG Unified Modeling Language (OMG UML), superstructure, v2.1.2, OMG document: formal/2007-11-04. Available at http://www.omg.org/docs/formal/07-11-02.pdf. Last visit 4 Jan 2009

Pakalnickiene E, Nemuraite L (2007) Checking of conceptual models with integrity constraints. Inf Technol Control 36(3):285–294

Rozanski N, Woods E (2005) Software system architecture. Addison-Wesley, London, p 546

Shen W, Compton K, Huggins J (2002) A toolset for supporting UML static and dynamic model checking. In: Proceedings of the 26th international computer software and applications conference (COMPSAC 2002), prolonging software life: development and redevelopment, Oxford, England. IEEE Computer Society, Washington, DC, pp 147–152

Silingas D, Butleris R (2009) Towards implementing a framework modelling software requirements in MagicDraw UML. Inf Technol Control 38(2):153–164

Sybase (2008) Customizing and extending power designer [disc]. PowerDesigner 15.0: help system

Van Der Straeten R, Simmonds J, Mens T, Jonckers V (2003) Using description logic to maintain consistency between UML models. In: UML 2003: LNCS 2863. Springer, Berlin/Heidelberg, pp 326–340

Vasilecas O, Dubauskaitė R, Rupnik R (2011) Consistency checking of UML business model. Technol Econ Dev Econ 17(1):133–150

Vavpotič D, Bajec M (2009) An approach for concurrent evaluation of technical and social aspects of software development methodologies. Inf Softw Technol 51(2):528–545

VeTIS (2009) Business rules solutions for information systems development. Project reg. No. B-07042, Lithuanian State Science and Studies Foundation

Zwanziger A, Holliday C, Herden S, McMackin T (2011) Use the topology editor in Rational Software Architect to add a custom validator: tutorial. URL: http://public.dhe.ibm.com/software/dw/rational/pdf/use-the-topology-editor-in-rational-software-architect-to-add-a-custom-validator.pdf. 19 Apr 2011

Chapter 34
Object-Oriented Analysis with Data Flow Diagram

Vaclav Repa

1 Introduction: Historical Background

In the era of the structured methods of information system development, that is, in the late 1980s, the data flow diagram (DFD) was regarded as the key tool for the IS development. In addition to the data model, which defines the contents of the database, DFD represented the second major dimension of the information system – its functionality. The structured methods contained also rules for applying the diagram and related analytical techniques, like the technique for the functional model development based on the analysis of events (event partitioning approach) by Edward Yourdon (1989).

In the late 1980s, the principles of object-oriented programming, which have been taking the shape from the beginning of that decade, were used analogically in the field of analysis; a number of new methodologies were created. These methodologies were aimed at the application of object-oriented thinking in the field of analysis in the traditional (structured) extent. A number of shortcomings, notably coming from the application of often irrelevant programming concepts into a qualitatively different field of analysis, were quite understandable regarding the novelty and obvious progressiveness of this way of thinking. Nevertheless, these new methods were a big move forward because of their ability to overcome some crucial problems in the structured approach using a new paradigm. In particular, the OMT (Yourdon 1989) masterfully combines the knowledge of structured approaches to new dimensions of thinking, open by the object paradigm. Data flow diagram occupied a key position in this method.

However, in the UML, the object-oriented modelling standard, created in the mid-1990s, DFD is not present. This diagram was replaced there by the Use Case Diagram. Unfortunately, the Use Case Diagram covers just a small part of the DFD scope.

V. Repa (✉)
Department of Information Technology, University of Economics,
W.Churchill sqr. 4, 130 67 Prague 3, Czech Republic

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_34,

While the DFD aims to cover the full functionality of the system Use Cases are describing only the user interface. Instead of the detailed analysis of functional elements and their relationships, the UML sees the functional content of an information system as a black box which we can examine only by its inputs and outputs. We must rely on the fact that all the functionality of the system is sufficiently described only with the object model (Class Diagram) in the form of mutually connected individual objects which represent the groups of isolated operations (methods). Unfortunately, such an approach cannot sufficiently describe the whole functionality of the IS. Therefore, the subsequent methodologies, built on UML, gradually reduced their interest from the information system as a whole to the so-called application software only. Consequently, the original starting point of the analysis – conceptual model of a real system, supplemented by a functional model of the IS – has been replaced with a set of "user requirements." This reduction was necessary for the methodologies implicitly using the UML.

The sad consequence of the above-mentioned facts is the current inability of developers to fully adopt the newer phenomenon of business processes. From the structured methods point of view, the theory of business process modelling and reengineering clarifies the fundamental link between the real (business) system and its supporting information system, just in the field of the functional model. It shows that the original idea that the functional model describes the dynamics of business systems is not correct. Dynamics of business systems is modelled by the business process model; the functional model describes information system content. Concerning business processes, current development methodologies oscillate between two extremes: the rejection (there is no space for the business system in heads of developers) and total usurping (the business process is what takes place on the computer).

During the rest of the 1990s, Data Flow Diagram gradually disappeared from analysis supporting tools, and at the beginning of the new millennium, no new CASE tool supported this diagram.

However, already in the middle of the first decade, the interest in the DFD has begun to rise noticeably and even began to get back into the CASE tools (such as Power Designer v.12.5). How to assess this fact? Is it only an insignificant detail, or is it a manifestation of a deeper objective needs coming back on the evolution spiral? The following text aims to find the rhetorical answer to this question.

This chapter is divided into six sections. After this introduction, the essentials of DFD are remembered in Sect. 2. Then the methodology for information modelling of organisations is introduced in Sect. 3 in order to explain the context of the functional model as a part of system models. Section 4 describes relationships to other system models in detail and discusses the problem of the system models consistency. In Sect. 5, the way of implementation of the DFD as a specialisation of the UML diagrams is roughly outlined in order to illustrate the nature of the DFD. The sixth section contains the topic summary and some conclusions.

2 What is DFD

Data flow diagram displays a functional model of the information system, describing the functions and their links. Function determines what processes are conducted in an information system in order to be a faithful model of supported reality. These processes of information system reflect the real business processes. Nevertheless, functional model is not a process model (even if it describes processes). DFD describes functions which exist in the system and substantial connections between them via the data flows. It is a static description. Described processes are processes of information system, processes which are modelling, not the business processes which are modelled. Just a static description of functions (non-procedural) is needed during the analysis, as all the procedural aspects of the reality behaviour are fully modelled by the business processes description. Procedural aspects of the behaviour of an information system are not the subject of analysis of business reality. Process description of the specific information system processes is realised in the phase of the system design (see, e.g. Deployment Diagram from the UML). The functional model of the system thus represents the basic analytical terms of reference and starting point of system design.

Figure 34.1 shows the data flow diagram describing how the data flow truth the information system functions from its inception (the "Terminator") to the final consumption ("Terminator"). These natural data streams (given by the nature of the real system) are interrupted on their way by saving them to the "data stores" because of the need to wait for more information on other events in order to allow its interpretation in the context of the information stored.

DFD has evolved from the so-called activity diagrams used in the SADT methodology (Marca and McGowan 1987).[1] SADT (Structured Analysis and Design Technique) is a methodology, used since about the mid-1970s. DFD is the most thoroughly methodically seized in the work of Edward Yourdon (Yourdon 1989) (see also www.yourdon.com).

The DFD has following basic elements: Function, Data Flow, Data Store, and Terminator (external entity)

Function is an essential element of the diagram. It represents the process of data processing. According to the principle of modelling, the function is an element of the real-world behaviour model. Nevertheless, it cannot be considered a business process, as it used to be often but erroneously considered. The function is a process, running in the information system, which itself is a specific model of the real (business) systems (including business processes). The function thus represents a unit of behaviour (performance) of an information system. Thus, function of the IS is never related to "physical" elements of the real system behaviour – business

[1] Note that the SADT activity diagram has (except the name) nothing common with the UML diagram of the same name, which is rather an analogy of classic flowchart for the description of the algorithm. This can be seen also in Chap. 5, dealing with "physical" nature of data flow diagram.

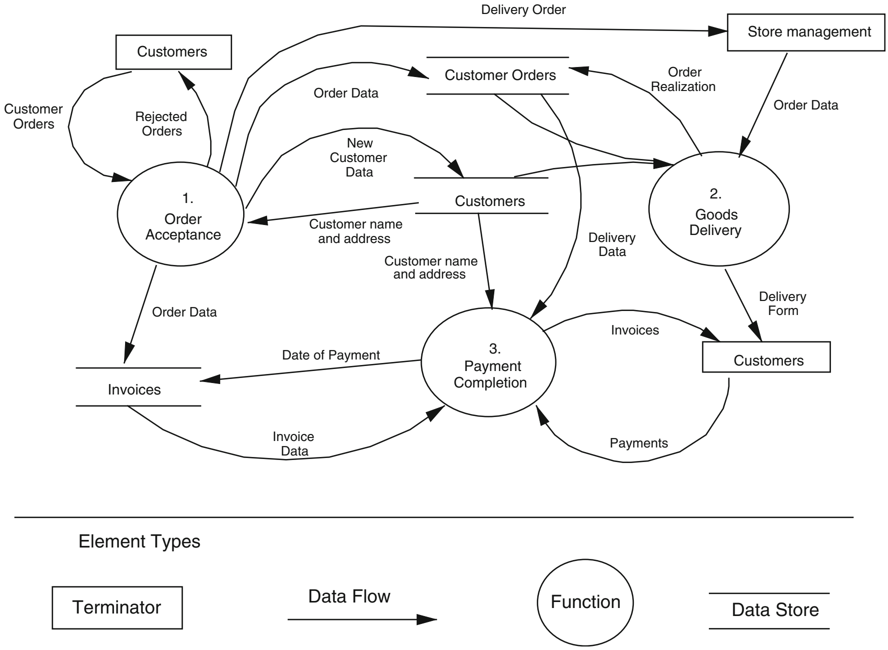


Fig. 34.1 Data flow diagram (Source: Yourdon 1989)

processes 1:1, but rather M:N, since typically many business processes need to meet the same IS function to support only part of their needs. Physically, the function represents the transformation of data, which leads to the production of an output (input to output transformation).

The difference between the business process and the IS function is clearly manifested by lately very popular service-orientation principle in the IS development. This principle is based on the fact that it is necessary to strongly distinguish between the elements of the IS functionality which are relatively stable and typical and their temporal combinations which follow from the natural (business) processes and their specific, often temporary, needs. Function as an element of the DFD thus represents the abstract content of the service. Consequently, the general criteria for the granularity of elementary functions in the DFD should be principally the same as criteria for the granularity of services from the enterprise architecture theory. From the IS/ICT management point of view and in the context of the enterprise architecture, these facts are discussed in greater detail in (Voříšek et al. 2011).

Traditionally, there are two basic kinds of processes described in the DFD:

Data process (function) expresses the physical transformation of data, that is, the change in data representation, or change of some part of the data, that is, the change in data values, the creation of new data. The main task of the function (process data) is to process (transform) the data.

Control Process expresses the control algorithm (cross-time continuity) of functions in a certain part of the system. It is used to capture the real-time characteristics of the application. In contrast to the function, the task of the control process is not to produce data. In data processing, systems thus control processes lose their meaning, they should not be considered in the analysis phase, but only in the design phase (see the deployment diagram from UML). It can be shown that, regarding the principles of object orientation, the presence of the information system processes in the analytical models is generally pointless. For these reasons, control processes are no more addressed in this chapter.

DataFlow represents an abstraction of any form of data transfer in the system and to/from it. Data flows contain data that are processed and stored in the system. Even though data flow diagram was originally developed for description of the flow of documents, materials, goods, raw materials, etc., it is not suitable to use this diagram for this purpose as it unnecessarily creates a hotbed of analytical errors. For this purpose, different, more appropriate tools have been created, such as the A-graphs of the ISAC methodology (see Lundeberg et al. 1981) or business process modelling tools (see Řepa 2004).

DataStore is an abstraction of any form of data storage in information systems. DataStore is a depository of data (data stored for later use). It is represented with two parallel lines. In the technical symbolism, this symbol represents the break, which points out that the data storage means interrupting the flow of data over time. This fact has a far-reaching importance in process modelling (e.g. see the events analysis technique described in Yourdon 1989).

DataStore as a "place for temporary storage of data" is used wherever there is a delayed (asynchronous) transmission of data between processes. It expresses only the fact that the data are kept (i.e. the flow is interrupted at the time) and says nothing about the particular form of storage (which is an issue of the IS implementation). DataStore is a secondary (passive) element in the diagram – data flows to and from must always be performed by functions.

Terminator is an object that does not belong to the described system but its substantial surrounding area. The Terminator (beginning or end of the data flow and data source, and the location and purpose of the data consumption) shows an external source or destination of data (sometimes also called an external entity – an object). It thus reflects the surrounding (real-world) system with which the information system communicates.

Functionality of the system is described not only with one DFD but with the hierarchical system of diagrams. Each function can be described in greater detail as a separate chart on the lower level (i.e. more detailed one). In such a system of diagrams, some elements of the interface functions are naturally repeating. This fact raises the general risk of inconsistencies models of both levels (i.e. situations where a detailed model describes the same interface element other than a higher-level model). This is called the consistency of diagrams hierarchy.

3 Functionality of the IS in the Context of an Information Model of Organisation

The following chapters are based on ideas of the methodology for information modelling of organisations, which is described in (Řepa 2010). As illustrated in Fig. 34.2, this methodology builds the information system on the model of the real (business) system, which consists of a conceptual business model (business structure of the system), and the model of business processes (business system behaviour). To specify the content of the information system, it is necessary to complete this model of reality with the functional model of information system in the form of a data flow diagram (see Fig. 34.2).

Functionality of an information system means the substance of its work, that is, what activities it supports and how (what data/information it provides). It is clear that, in terms of the principles of modelling, the functionality of an information system is also a reflection of the real system – reflection of real processes which are performed in order to insure the will to the creation, elimination, or support, etc., of real objects and their essential relations. In this sense, the functional model of information system should be derived from the model of real processes and model objects. However, in the information system itself – as part of a real system – the procedural and structural elements of reality manifest themselves in a specific way: as the data/information on real interrelated events, the knowledge of which is embedded in the intrinsic properties of an information system. The information system receives the information on real event in the form of its input (data flow) to combine it with other known events in order to derive information from

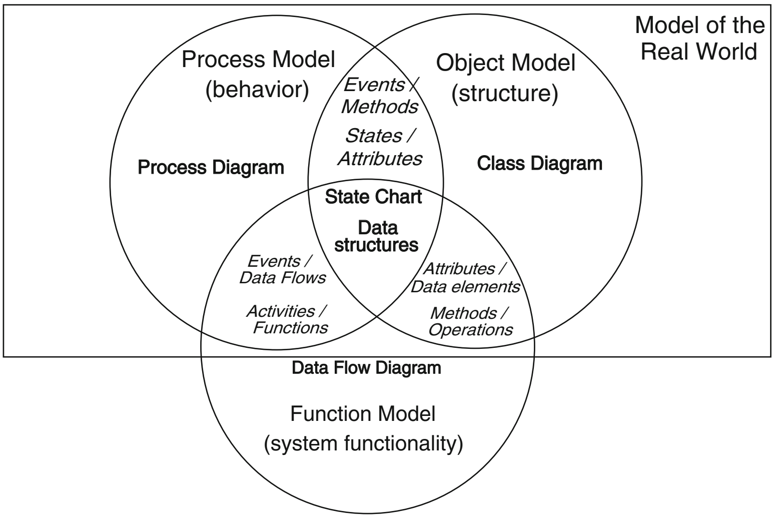


Fig. 34.2 The three components of the content model of information system

their relationship. Combining information about different time-independent (asynchronous) but factually related phenomena requires storing the information (i.e. remembering the event and waiting for the next future, factually related, events). To do this, the information system must "know" the reality. More precisely, the system must be able to assume primary (essential) continuity of basic (essential) real events; these expected relations must be "coded" in its structure.

Figure 34.2 describes the elements in a functional model which coincides with the model objects. They are both attributes of individual objects in the information system implemented in the form of data elements and structures (which reflect the interaction of objects), both methods of objects implemented in the form of information system operations and their structures (also reflecting the relationship of objects). The figure also describes the elements in which the functional model overlaps with the model of processes. Here are the events to which particular processes are reacting, realised in the information system in the form of data flows and structures (which reflect the interaction of processes – their communication), the activity of the processes implemented in the form of operations and information system functions and structures (reflecting the communication between processes as well).

The figure also shows common intersection of all three models – the common denominator of all bilateral common elements are the data structures (reflecting the attributes of objects and events and their essential relations) and also the essential regularities, reflecting the essential continuity of activities and events (the life cycles of objects).

The actual content of the IS function is a kind of specific (specialised) model of the real system in both its dimensions. Then, however, the functional content of the information system must be strictly distinguished from the form of its implementation (which is given generally by the technology environment and particularly by the implementation environment). Thus, the function model of the IS in terms of information system remains the model of content – technological, and implementation models will be derived from it. Some authors have therefore called it the essential model (e.g. E. Yourdon in his pivotal work (Yourdon 1989)).

4 Consistency of DFD and Other Information Model Diagrams

The need to ensure consistency results from the fact that one system is described by several models, which partially overlap in content (see the intersections of sets in Fig. 34.2). This means that one issue can be described in several different ways, and it is necessary to ensure that these different views are compatible. Therefore, the methodology formulates a set of rules governing relations between diagrams: Class Diagram (CD), Process Diagram (PD), and possibly state-transition diagram (STD, Statechart). These rules also substantiate the meaning of a functional model as an integrated part of information system models, as well as they manifest the strong need for instrumentation as UML, if it meets its responsibilities in the field of analysis (and not only the fulfilment of "user requirements"), discussed in the opening chapter.

4.1 Consistency of DFD and Object Model

Class Diagram (CD) represents in terms of DFD a general view of the database of the modelled system. It contains the data elements to store – the classes and their attributes – and a set of operations that can be carried out over such data in the form of an object's methods. DFD is, on the other hand, a model that captures the system interaction with the environment, its response to external events, and records which data structures are influenced by processes in the system due to their reactions to trigger events.

The points of contact of these two diagrams are primarily data and also, to some extent, the operations performed on the data (although not in relation 1:1).

In maintaining consistency of data between the DFD and the CD, it is therefore particularly necessary to maintain the link between each DataStore and the structure of some classes and their relationships in the CD. It is appropriate to maintain a link between class methods and data flows in the DFD. As this consistency link is obviously much less tight, it can be omitted if there are technical difficulties with the modelling tool.

The following set of rules describes aspects of the relations between the model classes (CD) and data flow diagram (DFD):

(a) Each elementary DataStore in the DFD must be represented in the CD as a class or association, or a combination of both.
(b) Attributes of each elementary DataStore in DFD must be a data structure composed from attributes of classes which represent this DataStore in the CD.
(c) Methods of each elementary function in the DFD must be an algorithmic structure composed from the methods of classes, which in the CD represent DataStore, associated by data flows with this function.

4.2 Consistency of DFD and Process Model

Process diagram (PD) is a model of the real (business) processes. Seen through the eyes of an information system, the processes are running across the system, and the information system provides them with the information support. As a model of the real (business) process, the Process Diagram does not address the structure of an information system but the transformation of the real inputs to outputs. It also monitors the resources consumed in this transformation.

The activities that are displayed in PD have some form of their image in the processes in DFD, although the relationship is not 1:1 but rather 0:N. The reason for this misty relationship is particularly the different focus and character of the two charts – one describes a business processes and a second the structure of functions. However, the activities of the business process should have its image in the DFD.

Since the process model and DFD are working with events, the same events should appear in both diagrams, or there would be some relationship among the events of the DFD and the PD at least.

The following set of rules describes aspects of the relations between the process model (PD) and the data flow diagram (DFD):

(d) Each elementary input DataFlow from the Terminator (i.e. from outside the system) in the DFD must correspond to an event specified in (some) business process (processes) in the PD.
(e) Each state of each process in PD must correspond to certain (some) elementary DataStore (DataStores) in the DFD and vice versa (every elementary DataStore from the DFD must correspond to certain (some) state (states) of the process (processes) in the PD. It is an M:N correspondence.

4.3 *Consistency of DFD and STD*

State-Transition Diagram (StateChart) describes the general life cycle of objects of the class. It enumerates the states to which the object of the class can come and possible transitions between these states. Each transition is described by the action that occurs on an event from the outside world. The transition is caused by an event that must have an image even in DFD and PD. STD is a basic tool for describing the relationships between object and process models of reality (see previous chapter, including rules of consistency). From the DFD point of view, the STD is thus hidden behind these models and may not be, in terms of functionality of the system, taken into account.

Figure 34.3 illustrates some basic relationships between the models described above.

5 Implementation of Data Flow Diagram in the UML

In the basic methodology (Řepa 2010), the data flow diagram is implemented as a specific profile of the UML, based on the Class Diagram. Function, DataStore, and Terminator are represented by specific stereotypes of a Class; DataFlow is represented by specific stereotype of Association. The natural hierarchical structure of diagrams is implemented as a link between the function (class of stereotype <<function>>) and associated data flow diagram (class diagram of stereotype <<data flow diagram>>).

In the previous text, the need for maintaining consistency of data flow diagram with other diagrams have been discussed. DFD, implemented in the UML on the base of the Class Diagram, can easily be linked with STD, CD, as well as design-oriented diagrams (see UML).

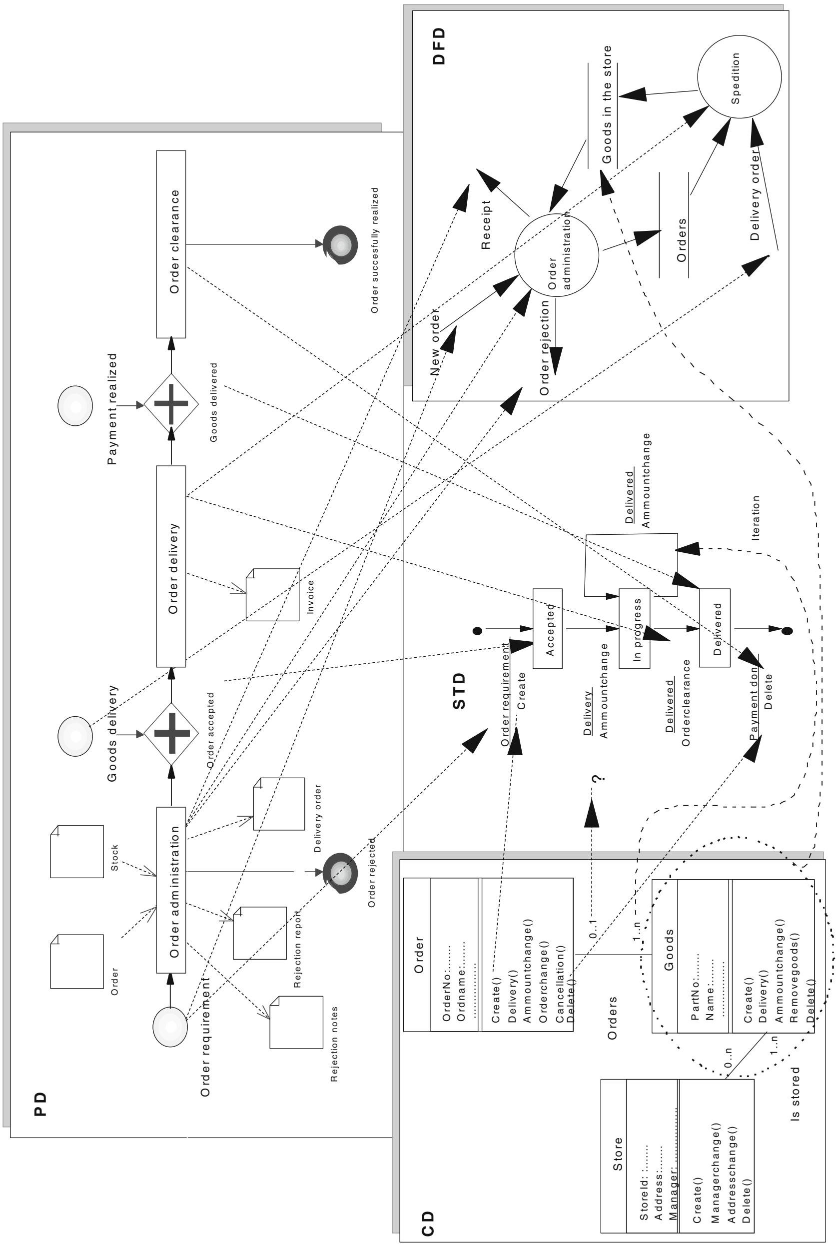


Fig. 34.3 Consistency of the model of processes, objects, and DFD

Specialisation of the Class Diagram to DFD is implemented by introducing four standard stereotypes:

DFD element	UML metaclass	Specialisation (basic restrictions) of the metaclass
DataStore	Class	Standard methods of reading, writing, and deleting (destructor)
Function	Class	Methods are irrelevant
Terminator	Class	Methods are irrelevant
DataFlow	Association	Oriented, single, or two way

For such a stereotyped model of classes, the following rules are valid:

- DataStore must have at least one input DataFlow and one output DataFlow.
- DataFlow may only connect Function with Function, Function with DataStore, or Function with Terminator.
- There must be an event associated to the DataFlow between Terminator and Function.
- For every Function there must be at least one associated DataFlow.

6 Conclusions

As the previous chapter shows, the "physical" essence of the data flow diagram is the *conceptual model of information system functions*. This knowledge is dramatically different from the general notion that the nature of this model is procedural, expressed even by the father of the structured methodology Edward Yourdon in (1989). This idea led to other, generally accepted misconceptions, such as "functional model is a model of the business processes".

Taking into account the Yourdon's precisely formulated rules for the use of DFD as well as his technique of analysis of events, all in the context of current knowledge in the procedural dimension of business systems, it is obvious that the function model is naturally structural (i.e. object oriented, not process oriented).

In addition to simplicity and natural consistency of the implementation of DFD on the base of the Class Diagram described in the previous chapter, the correctness of the idea of a conceptual nature of DFD if confirmed by the methodologies for object-oriented analysis evolution itself. In particular, there are some general classifications of types of analytical objects (see Fowler 1997 for instance), such as functional objects (alias functions), data objects (alias data stores), and business objects (terminators/conceptual entities).

The data flow diagram and its methodological particulars, like the rules of use and techniques for linking with other diagrams, represent a large methodology value which should remain in the further development of methodologies maintained. Anyway, given the objective nature of values, there is no risk that this value would disappear at all; it is rather a matter of time when it appears in some modified form. It is certainly better to use the knowledge already invented instead of reinventing the wheel. The certain way of non-reinventing the wheel outlines also this chapter: just realise the basic context and overcome stereotypes in thinking given by different, just seemingly contradictory paradigms. As it is usual in the evolution process, the future is in the synthesis, rather than in simple negation.

Acknowledgements The work presented in this chapter has been supported by the Czech Science Foundation in the grant project No. P403/10/0303 Enterprise Architecture as Management Principle for SMEs.

References

Fowler M (1997) Analysis patterns: reusable object models. Addison-Wesley, Reading

Lundeberg M, Goldkuhl G, Nilsson A (1981) Information systems development – a systematic approach. Prentice-Hall, Englewood Cliffs

Marca D, McGowan C (1987) Structured analysis and design technique. McGraw-Hill, New York. ISBN 0-07-040235-3

Řepa V (2010) Consistency in the business system model. In: Družovec TW, Jaakkola H, Kiyoki Y, Tokuda T, Yoshida N (eds) Information modelling and knowledge bases XXI. Ios Press, Amsterdam, pp s. 247–s. 257, 438 s. ISBN 978-1-60750-089-6

Řepa V (2004) Business processes and objects in an information systems development methodology. In: Abramowicz W (ed) Business Information Systems – BIS 2004, Poznan, 21–23 Apr 2004. ISBN 83-7417-019-0

Rumbaugh J, Blaha M, Premerlani W, Eddy F, Lorensen W (1990) Object-oriented modelling and design. Prentice Hall, Englewood Cliffs. ISBN 0-13-629841-9

Voříšek J, Jandoš J, Feuerlicht J (2011) SPSPR model – framework for ICT services management. J Syst Integ 2(2):3–10, ISSN 1804-2724

Yourdon E (1989) Modern structured analysis. Prentice Hall, Englewood Cliffs

Chapter 35
Toward Understanding Contradictions in Enterprise System Implementations: Insights from a Case Study

Stig Nordheim

1 Introduction

Implementing an enterprise system (ES) in an organization is both challenging and expensive (Seddon 2005). In ES implementations, value conflicts occur between stakeholders (Allen 2005), and a dialectic perspective thus explains important aspects of the ES implementation process (Besson and Rowe 2001; Nordheim and Päivärinta 2006; Robey et al. 2002; Sia and Soh 2007). A dialectic perspective views change as the result of contradictory values competing for domination (Van De Ven and Poole 1995).

This study focuses on an ES different from ERP systems. The case is a combination of technical solutions, including an integration of enterprise content management with collaboration solutions and personal productivity tools. A number of software systems were integrated, and dialectics occurred in the implementation process. Having established main contradictions in an ES implementation process, one needs to explore them further in order to understand them. Thus, a rather fundamental research question is raised:

How can we understand contradictions in enterprise system implementations?

To understand and explain contradictions is a necessary prerequisite for being able to deal with them in a constructive way.

This chapter is organized as follows: Sect. 2 presents background literature. After a description of the research method in Sect. 3, Sect. 3 describes the case. Section 4 presents the findings on how to understand contradictions. Section 5 discusses the findings, concluding with implications for ES research and practice.

S. Nordheim
University of Agder, Postboks 4224604 Kristiansand, Norway
e-mail: Stig.Nordheim@uia.no

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_35,

2 Background Literature

The term enterprise system (ES) has traditionally been related to ERP systems (Davenport 1998). In addition to ERP, the range of available enterprise systems is growing and includes customer relationship management (CRM), supply chain management (SCM), and enterprise portals (Ward et al. 2005). Another enterprise-wide system is enterprise content management (ECM) (Päivärinta and Munkvold 2005; Smith and McKeen 2003).

Enterprise systems may in a wide sense be viewed as a response to a need aptly summarized by Markus (1997), who stated that the world around us is demanding that we find ways to satisfy unique needs with generic software and components. This is no small challenge for organizations, as research on ES implementation has established (e.g., Soh and Sia 2005).

2.1 Dialectics

Dialectics is one way to view organizational change, as the result of contradictory values competing for domination (Van de Ven and Poole 1995). Dialectics is a way of thinking that is based on contradictions. A contradiction can be viewed as a relation between two opposite aspects of a phenomenon, called thesis and antithesis, where antithesis is the negation of the thesis. The two aspects of a contradiction are intrinsically related yet opposite and distinct from one another (Van de Ven and Poole 1995). In dialectical theory, stability and change are explained by reference to the balance of power between the two opposing entities. A thesis (A) may be challenged by an antithesis (not-A), and the resolution of the conflict becomes a synthesis (which is not not-A). By its very nature, the synthesis is a novel construction that departs from both the thesis and the antithesis. This synthesis, in turn, becomes a new thesis as the dialectical process continues. Dialectics is about dynamics and is one way of explaining development and change (Van de Ven and Poole 1995).

Dialectical reflection is a way to understand a situation (Israel 1979). Dialectical theory provides insights into IS development but does not treat relationships between organizations and IS as determinate, causal connections. Instead, it examines them as emerging through social constructions (Sabherwal and Newman 2003). The mutual adaptation between the technology and the organization is far from a planned change, and the outcomes of the process are difficult to predict (Wei et al. 2005).

2.2 Commonality and Variability: A Vendor Challenge

One way to achieve the challenge to satisfy unique needs with generic software and components (c.f. Markus 1997) is by means of two fundamental design concepts which are a characteristic of most software packages. That is the distinction between commonality and variability (Bühne et al. 2005). These two fundamental design concepts are also used by ES vendors (Leishman 1999). In a previous study of ES vendors, one major ES vendor challenge was found, and this was termed a dialectic of design (Nordheim 2007). This challenge is to find an optimal balance between stability and change, between commonality and variability. For an ES customer, there are two important implications of the challenging dialectic of ES design. First, if ES variability is unable to meet the customer's requirements, a considerable pressure is applied to make the customer adapt to the system. ES vendors were also found to simultaneously promote both commonality and variability in a dialectic rhetoric vis-à-vis a potential customer (Nordheim 2007).

2.3 Understanding Contradictions: Previous Research

One contradiction related to ERP implementation is termed a dialectic of learning (Robey et al. 2002). The dialectic of learning occurs between the old knowledge embedded in business processes and practices associated with legacy systems and the new business processes and practices that ERP is designed to support.

Other research on ERP contradictions is Besson and Rowe's (2001), who found that during and after the shakedown phase, targeted organizational outcomes are often not realized because of job and governance conflicts among the users and other stakeholders. Hence, the dialectical view often dominates the shakedown phase, with task, resource, and power conflicts among the stakeholders (Besson and Rowe 2001).

An ES entails many stakeholders who typically have multiple and often conflicting objectives and priorities and rarely agree on a set of common aims (Sedera et al. 2004). Stakeholders with divergent interests can play a vital role in ES implementation success (Boonstra 2006). Therefore, value conflicts occur between stakeholders in ES implementation processes (Allen 2005).

Another stream of research has focused on ERP misfits by applying a dialectic conceptualization (Soh et al. 2003; Soh and Sia 2005; Sia and Soh 2007). According to this research stream, a misfit emerges between the features of an ERP package and the specific requirements of an organization adapting the package. This misfit between the ERP's structures and the structures of the implementing organization may be solved either by modifying the package or changing the organization (Soh and Sia 2005).

To summarize the ERP literature, it does provide some answers to the research question. We may understand contradictions in ES implementation as (1) old knowledge versus new knowledge (Robey et al. 2002), (2) conflicts between stakeholders in the shakedown phase (Besson and Rowe 2001), (3) contradictions may be understood as conflicting objectives and priorities between stakeholders (Allen 2005), and (4) misfits between structures in the ES and in the organization (Soh et al. 2003; Sia and Soh 2007).

3 Case Background and Description

Statoil is a technology-based international energy company that primarily focuses on upstream oil and gas operations. Statoil's headquarters are in Norway, and the largest shareholder is the Norwegian state with a majority of the shares. The company is the third largest exporter of crude oil in the world, with approximately 25,600 employees in 33 countries at the time of study.

Statoil was one of the world's largest users of Lotus Notes/Domino in the 1990s (Munkvold and Tvedte 2003). They launched a major ECM and collaboration development program from 2002 to 2007, which is the object of study. The project included collaboration and information sharing across organizational and geographical boundaries, with access to corporate information. Each employee in Statoil is attached to team sites to conduct their knowledge work and collaboration. All relevant information for a project or team is accessible to the team sites, with all documents being managed throughout their life cycles.

The technologies that constitute the enterprise system include Meridio ECM combined with Stratify content classification (taxonomy) software. This is closely integrated with the FAST Search & Transfer software. Microsoft SharePoint portal server combined with Microsoft Live Meeting constitutes important collaboration software, and personal productivity tools include Microsoft Office. The solution spans a wide range of technologies that are tightly integrated. Statoil employees and partners collaborate in the team sites, and their information objects are managed in a way that is transparent to the user. The team sites, corporate-wide integrated storage, and search engines are accessed through a corporate portal. The case is therefore a complex ES solution. For Statoil, this ES implementation project is an effort comparable to their ERP implementation in the 1990s.

The project was organized with the CIO as sponsor, a steering committee, a project group with a project manager, and a reference group with representatives from the business units. In addition, there were a quality assurance group and a solution handover group. The steering committee consisted of process owners and IS/IT managers from different business units having a customer role. The sponsor represented corporate management, was financially and commercially responsible, and chaired the steering committee. The project group was staffed by corporate services IS/IT. Vendor and other consultants were involved as needed, but Statoil staff did a major part of the work. Informants came from the project group, steering committee, corporate user representatives, reference group, and quality assurance group.

3.1 Data Collection and Analysis

Some of the reviewed literature establishes that contradictions emerge through social interaction between individuals and groups. This implies that the phenomenon under study fits with interpretive assumptions, that our knowledge of reality,

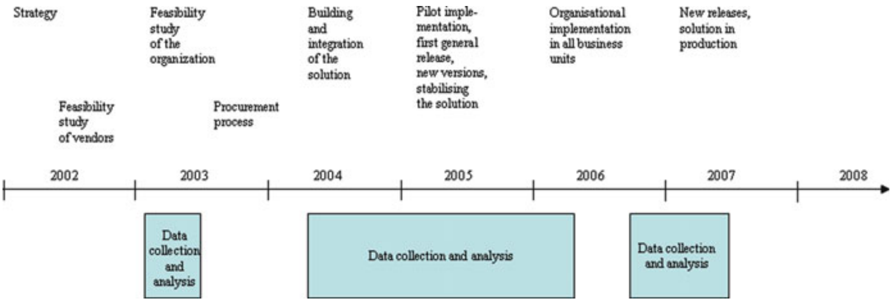


Fig. 35.1 The main project phases of the Statoil case above the timeline, together with research activities below

including the domain of human action, is a social construction by human actors (Walsham 2006). The process of implementing an enterprise system in an organizational context is time-consuming, and to understand how this process unfolds, a longitudinal study has been viewed appropriate. The interpretive research approach has therefore been guided by Pettigrew's (1995) advice on longitudinal research on change.

The ES implementation project lasted from 2002 to 2007 and has been studied from an outside observer viewpoint in three distinct periods. The time frame of the project and the research activities are outlined in Fig. 35.1. Above the timeline are the main project phases, and below are the phases of research.

The data collection activities included a total of 23 interviews with 15 persons and about 840 pages of documents analyzed. To get access to information-rich informants, a partial "snowball" or chain sampling of informants was used (Patton 1990). The principal data collection method was in-depth, semi-structured interviews, combined with document analysis. To remain both focused and open, a general interview guide approach was combined with an informal conversational interview, to "go with the flow" (Patton 1990). The interviews typically lasted 45 min. All interviews were audiotaped and transcribed.

The research heeded Pettigrew's (1995) stance on how to study organizational change in context, namely, that it requires multilevel analysis and processual analysis. Multilevel analysis was done at (1) project level, (2) group level with groups of users having diverging interests, and (3) individual level. The interviews were coded, and a deductive analysis was based on the interview guides. Data reduction by tables and displays (Miles and Huberman 1994) were made. A visual mapping strategy (Langley 1999) was also used.

3.2 The Dialectic of Adaptation and the Appointment of a Special Role

In this case, one main contradiction was found, termed the dialectic of adaptation. It is described in detail by Nordheim and Päivärinta (2006) and emerged between the features of the ES packages versus the organization-specific requirements addressed by the organization. The thesis in this contradiction was an “out-of-the-box” implementation of commercial software. The antithesis was an implementation fulfilling Statoil’s requirements concerning solution integration and user experience. The contradiction may be phrased as follows: “we ought to implement commercial software as it is, out-of-the-box” versus “no, we ought to implement the software so that we fulfill our requirements concerning solution integration and a simplified user experience.” The contradiction manifested as 42 critical “issues” that jeopardized the entire project. They had to be solved before the project could continue (Nordheim and Päivärinta 2006).

As a result of and in a direct response to the dialectic of adaptation, a special role was established. Termed a corporate user representative, this role became an important way to deal with the dialectic of adaptation. A description of the corporate user representative role and how this role dealt with the contradictions is described in two previous publications (Nordheim 2008; Nordheim and Nielsen 2008).

4 Findings: Different Perspectives on Contradictions

In this case, contradictions can be understood from the following perspectives:

- A pre-implementation perspective, in view of ES vendor challenges
- An early chartering phase perspective, by a focus on perceived customization needs
- A project phase perspective, as the contradictions manifested as critical issues

The understandings of contradictions from these perspectives are presented in the following.

4.1 Understanding Contradictions in View of ES Vendor Challenges

As pointed out in Sect. 2.2, a basic challenge that ES vendors face was termed a dialectic of design (Nordheim 2007). This challenge is to find an optimal balance between stability and change, between commonality and variability. By comparing the contradictions involved in the dialectic of adaptation with the vendors’ challenging dialectic of design, commonality and variability are underlying the issues in the contradictions.

Commonality was underlying the governing principle of using standard software “out-of-the-box.” As the project manager during the most intense project phase stated:

> We were required to solve everything "out-of-the-box," as a general principle. As we unpacked the solution "out-of-the-box" and saw how it worked, we quickly found out that we had to do quite a lot. But there has always been a tension between the steering committee and the project concerning how far we should go with the local adaptation.

The limits of commonality can clearly be seen in the "out-of-the-box" pilot installation of the purchased products on Statoil's own ICT infrastructure. This resulted in a number of concerns, conceptualized as 191 "issues" identified for resolution during the implementation. The issues represented "weaknesses, challenges and uncertainties" in the solution, and 42 of the issues were categorized as highly critical (Nordheim and Päivärinta 2006).

Variability solved some of the issues, as they only required a simple configuration task. However, custom components were needed to supplement the limitations of commonality. The custom components were strictly prioritized, addressing solution integration and user experience. The limits of the built-in commonality and variability may also be illustrated by some statements by the corporate user representative on his tensions with different project groups:

> Is this really not possible to solve, with the products we bought?
>
> We have a problem here, what we bought turned out to be less flexible than we thought.
>
> We have to change the scope, [we need to] do things differently, and the solution will be different from what we decided.

By analyzing the dialectic of adaptation in terms of the ES vendors' commonality/variability design challenges, these were clearly at the heart of the dialectic of adaptation. Commonality was largely expressed by the CIO's governance principle of using standard software "out-of-the-box." Variability related to solution integration and user experience was a particular challenge, and the case shows that variability was supplemented with custom components to achieve integration and user experience beyond the built-in variability of the software. The vendor challenges (Nordheim 2007) are therefore found to be underlying the dialectic of adaptation.

4.2 Exploring Manifestations of Contradictions Early in the Chartering Phase

An attempt to explore perceived customization needs was made early in the chartering phase (Nordheim and Päivärinta 2004). In retrospect, this initial analysis can be viewed as an early attempt to understand possible latent contradictions that later surfaced in the ES implementation. Customization is certainly not recommended (e.g., Beatty and Williams 2006), but perceived needs for customization was here only used as a way to explore and anticipate possible contradictions early in the chartering phase. The study concluded that perceived needs for customization were mainly seen in the following areas: nonfunctional integration with existing software, simplification of user interfaces, and functional adaptation and simplification (Nordheim and Päivärinta 2004).

It turned out that exactly these customization needs later surfaced in contradictions. Integration and user experience later turned out to be key issues in the dialectic of adaptation (Nordheim and Päivärinta 2006). This shows that one could thus begin to understand important contradictions by focusing on perceived customization needs early in the chartering phase, before any contradictions emerged. With the benefit of hindsight, it can be seen that a focus on perceived customization needs early in the chartering phase was useful. This focus helped to uncover important issues that later emerged in the antithesis of the dialectic of adaptation.

4.3 Understanding Contradictions: Previous Findings from the Case

The case study has previously been analyzed in terms of a hybrid development approach involving three “motors” of change (Nordheim and Päivärinta 2006). Although not explicitly focused on understanding contradictions, this study presents three important insights worth including here.

The first understanding of the dialectic of adaptation is its emergence. The dialectic of adaptation could be traced back to the project strategy. However, this dialectic was only latent in the beginning. It fully emerged in the project phase. The second understanding is that contradictions are only part of the picture. Viewed in terms of Van de Ven and Poole’s (1995) four motors of change, a hybrid theoretical pattern of change was found best suited to describe the ES implementation. In other words, contradictions must in this case be understood as one of several drivers of change in the project organization. The third understanding of contradictions is that a stakeholder perspective is also important to understand the dialectic of adaptation. Diverging viewpoints on functionality surfaced within the steering group, and representatives from different domains advocated their needs. The emerging contradictions led to sharpened prioritizations of implementation issues and organizational learning among the stakeholders (Nordheim and Päivärinta 2006).

Table 35.1 summarizes the different understandings of contradictions obtained from this case study and shows how the dialectic of adaptation may be understood from these perspectives.

Table 35.1 Perspectives for understanding contradictions in an ES implementation

Understanding	Description
Pre-implementation considerations: the vendors' dialectic of design	The dialectic of adaptation may be understood in view of the vendors' challenging dialectic of design, to find an optimal balance between commonality and variability. Commonality and limited variability were underlying the thesis of the dialectic of adaptation
Early chartering phase exploration: perceived customization needs for the ES	The dialectic of adaptation may early in the chartering phase be understood by exploring perceived customization needs. These needs included integration and user interface issues and were later core issues in the dialectic of adaptation
Project phase: emergence	The dialectic of adaptation emerged from being latent in the project strategy to surface as critical issues in the project phase
Project phase: hybrid pattern	The dialectic of adaptation emerged in combination with the teleological and life cycle motors of change
Project phase: stakeholders	The dialectic of adaptation occurred as representatives from different solution domains advocate their needs. Different user groups want to get "their" components prioritized first

5 Discussion

This chapter summarizes understandings from one case. The research question was as follows: how can we understand contradictions in enterprise system implementations? Given the dialectic of adaptation as a main contradiction, Table 35.1 summarizes different perspectives on this complex phenomenon.

The finding that the vendors' challenging dialectic of design was underlying the dialectic of adaptation is a new contribution. It helps to understand why package-organization misalignments occur (Sia and Soh 2007). Of a more practical nature is the finding that an early focus on perceived customization needs surfaced issues in later contradictions. Issues in the dialectic of adaptation could be understood early in the chartering phase by focusing on perceived customization needs, and this is a new contribution.

Previous research on this case has established that contradictions emerged in the project phase, as part of a hybrid theoretical pattern of organizational change and as the result of diverging viewpoints among user groups (Nordheim and Päivärinta 2006). Taken together, the findings from this case help us to understand some of the complexity underlying contradictions.

The point of understanding contradictions is to be better able to deal constructively with them. In this case, the establishment of a corporate user representative role became an important way to deal with the dialectic of adaptation (Nordheim 2008; Nordheim and Nielsen 2008).

These findings should have relevance beyond this case, although there are some limitations on how the findings may be generalized: (1) the case was a combination of collaboration and information management technologies, (2) the time of study is mainly the chartering and project phases, with only a few interviews early in the shakedown phase, (3) the informants mainly represent the project perspective, and (4) the focus is to a large extent limited to the dialectic of adaptation. However, the understandings of the dialectic of adaptation are related to the enterprise-wide nature of these systems and should therefore be applicable to other enterprise systems.

Although the different perspectives presented here contribute to our understanding of contradictions in enterprise system implementations, the research question is still relevant. There may be other perspectives for understanding contradictions than the ones presented here, and future research could well focus on other contexts and later phases of ES implementation.

A conscious strategy of looking for contradictions and pursuing a constructive synthesis could help manage large-scale ES projects. An early focus on perceived customization needs may as in this case help to uncover potential future contradictions in the project. One should also expect diverging viewpoints among user groups in the project phase, and one may anticipate these by appointing a carefully selected corporate user representative, as done in this case.

To conclude, two new understandings have been presented: (1) by considering the larger context of vendor challenges, the dialectic of adaptation can be viewed as manifestations of the vendors' dialectic of design and (2) by considering limits of variability in an early chartering phase, an early focus on customization needs helped to surface critical issues early. This chapter also contributes by putting together a larger picture of how contradictions in an ES implementation can be understood. Understanding contradictions involves several dimensions, and an openness to these could help identify and deal with contradictions in future ES projects.

References

Allen JP (2005) Value conflicts in enterprise systems. Inf Technol People 18(1):33–49

Beatty RC, Williams CD (2006) ERP II: best practices for successfully implementing an ERP upgrade. Commun ACM 49(3):105–109

Besson P, Rowe F (2001) ERP project dynamics and enacted dialogue: perceived understanding, perceived leeway, and the nature of task-related conflicts. SIGMIS Database 32(4):47–66

Boonstra A (2006) Interpreting an ERP-implementation project from a stakeholder perspective. Int J Project Manage 24:38–52

Bühne S, Lauenroth K, Pohl K (2005) Modelling requirements variability across product lines. In: 13th IEEE international conference on requirements engineering, Paris, France. IEEE Computer Society, Los Alamitos, pp 41–50

Davenport TH (1998) Putting the enterprise into the enterprise system. Harv Bus Rev 76 (4):122–131

Israel J (1979) The language of dialectics and the dialectics of language. Munksgaard, Copenhagen

Langley A (1999) Strategies for theorizing from process data. Acad Manage Rev 24(4):691–712
Leishman DA (1999) Solution customization. IBM Syst J 38(1):76–97
Markus ML (1997) The qualitative difference in IS research and practice. In: Proceedings of the IFIP TC 8 8.2 international conference on information systems and qualitative research, Philadelphia. Chapman & Hall, London, pp 11–27
Miles MB, Huberman AM (1994) Qualitative data analysis: an expanded sourcebook. Sage, Thousand Oaks
Munkvold BE, Tvedte B (2003) Implementing a portfolio of collaboration technologies in Statoil. In: Munkvold BE (ed) Implementing collaboration technologies in industry. Springer, London, pp 81–107
Nordheim S (2007) Towards a dialectic understanding of enterprise systems – vendor challenges and contradictory rhetoric. In: Magyar G et al (eds) Advances of information systems development: new methods and practice for the networked society, vol 1. Springer, New York, pp 11–22
Nordheim S (2008) Corporate user representatives and the dialectics of enterprise systems: a quest for social actors with political skill. In: Proceedings of the 41st Hawaii international conference on system sciences, Big Island, Hawaii. IEEE Computer Society, Los Alamitos. (CD-ROM)
Nordheim S, Nielsen PA (2008) Enterprise system implementations: organizational influence processes for corporate user representatives. In: Proceedings of the 16th European conference on information systems, Galway, Ireland. (CD-ROM)
Nordheim S, Päivärinta T (2004) Customization of enterprise content management systems: an exploratory case study. In: Proceedings of the 37th Hawaii international conference of system sciences, Big Island, Hawaii. IEEE Computer Society, Los Alamitos. (CD-ROM)
Nordheim S, Päivärinta T (2006) Implementing enterprise content management: from evolution through strategy to contradictions Out-of-the-Box. Eur J Inf Syst 15:648–662
Päivärinta T, Munkvold BE (2005) Enterprise content management: an integrated perspective on information management. In: Sprague RH JR (ed) Proceedings of the 38th Hawaii international conference on system sciences, CD-ROM. IEEE Computer Society, Los Alamitos
Patton MQ (1990) Qualitative evaluation and research methods. Sage Publications, Newbury Park
Pettigrew AM (1995) Longitudinal field research on change. In: Huber GP, Van de Ven AH (eds) Longitudinal field research methods: studying processes of organizational change. Sage Publications, Thousand Oaks, pp 91–125
Robey D, Ross JW, Boudreau M-C (2002) Learning to implement enterprise systems: an exploratory study of the dialectics of change. J Manage Inf Syst 19(1):17–46
Sabherwal R, Newman M (2003) Persistence and change in system development: a dialectical view. J Inf Technol 18(2):69–92
Seddon PB (2005) Are ERP systems a source of competitive advantage? Strateg Change 14:283–293
Sedera D, Gable GG, Chan T (2004) Measuring enterprise systems success: the importance of a multiple stakeholder perspective. In: Proceedings of the 13th European conference on information systems, Turku, Finland
Sia SK, Soh C (2007) An assessment of package–organisation misalignment: institutional and ontological structures. Eur J Inf Syst 16:568–583
Smith HA, McKeen JD (2003) Developments in practice viii: enterprise content management. Commun AIS 11(33):1–26
Soh C, Sia SK (2005) The challenges of implementing vanilla versions of enterprise systems. MIS Q Exec 4(3):373–384
Soh C, Sia SK et al (2003) Misalignments in ERP implementation: a dialectic perspective. Int J Hum-Comput Int 16(1):81–100
Van de Ven AH, Poole MS (1995) Explaining development and change in organizations. Acad Manage Rev 20(3):510–540
Walsham G (2006) Doing interpretive research. Eur J Inf Syst 15:320–330

Ward J, Hemingway C, Daniel E (2005) A framework for addressing the organisational issues of enterprise systems implementation. J Strateg Inf Syst 14:97–119

Wei H-L, Wang ETG, Pei-Hung J (2005) Understanding misalignment and cascading change of ERP implementation: a stage view of process analysis. Eur J Inf Syst 14(4):324–334

Chapter 36
Requirements Elicitation with Web-Based Focus Groups

Carla Farinha and Miguel Mira da Silva

1 Introduction

Requirements are the heart of Information Systems Development since the earliest days of computing (Avison and Fitzgerald 2006). Requirements engineering is an important and crucial process of development because requirements determine how the system will operate (Davey and Cope 2008; Apshvalka et al. 2009; Zowghi and Coulin 2005). Nevertheless, problems still exist (Nuseibeh and Easterbrook 2000; Preece et al. 2002; Christel and Kang 1992; Saqid et al. 2010; Hossenlopp and Hass 2007) and are major causes for the high failure rate of these projects (T.S. Group 2009).

Requirements elicitation aims to identify requirements through intense communication between stakeholders and analysts. This complex and error-prone communication shows that stakeholders are not always clear in describing what they need and analysts have difficulties understanding business concepts (Nuseibeh and Easterbrook 2000; Apshvalka et al. 2009; Burg 1996; Preece et al. 2002; Pfleeger and Atlee 1998). The errors in this activity become expensive and hard to fix (Kitzinger 1994), costing around 80–100 times less than if discovered at the implementation stage (Crabtree et al. 2000). As a result, the requirements elicitation activity is usually accepted as a critical activity of the requirements engineering process (Davey and Cope 2008; Apshvalka et al. 2009; Geisser and Hildenbrand 2006; Engelbrektsson et al. 2000).

Many directions of recent research focus the social nature of requirements elicitation, leading to the usage of social sciences approaches (Zowghi and Coulin 2005;

C. Farinha (✉)
Opensoft, Rua Joshua Benoliel, 1, Lisbon, Portugal
e-mail: carla.farinha@opensoft.pt

M.M. da Silva
IST/INOV, R. Alves Redol, 9, Lisbon, Portugal
e-mail: mms@ist.utl.pt

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_36,

Pfleeger and Atlee 1998), including ethnography (Crabtree 1998; Crabtree et al. 2000), interviews (Al-Rawas and Easterbrook 1996; Davey and Cope 2008) or group work (Farinha and Mira da Silva 2009; Davidson 1999; Saqid et al. 2010; Engelbrektsson et al. 2000).

We have been investigating the use of focus groups to overcome major problems of requirements elicitation (Farinha and Mira da Silva 2009). Discussions offer an overview of the global system and needs, allowing to identify requirements and to negotiate conflicts more efficiently. This proposal was evaluated with Action Research that allows understanding and acting over the situation, developing skills of members to overcome that situation, and adding scientific knowledge (Coghlan and Brannick 2009; Bhattacharjya and Venable 2006; Kock et al. 1997; Koshy 2005).

In this chapter we present our most recent research in the field of requirements elicitation, including a proposal and results. We used web-based focus groups to involve all stakeholders in the identification of needs to an Information System. This chapter explains the proposal, presents experiments, and discusses results. Conclusions demonstrate that this method avoids limitations of conventional focus groups, including dominant users or difficulties gathering relevant stakeholders.

2 Requirements Elicitation Trends

Information Systems Development begins with requirements engineering that identifies, analyzes, specifies, verifies, and validates activities (Avison and Fitzgerald 2006; Burg 1996; Maté and Silva 2005; Tsumaki and Tamai 2005).

The next sections present social approaches to elicit requirements as a recent trend of investigation and also present group work as promising approaches.

2.1 Social Approaches

Requirements elicitation is one of the most critical activities of requirements engineering (Zowghi and Coulin 2005; Preece et al. 2002; Saqid et al. 2010) for many reasons. First, the activity relies on a complex and error-prone communication between stakeholders and analysts. Second, stakeholders are not always clear about what they want or need. Finally, analysts have difficulties understanding the business concepts (Nuseibeh and Easterbrook 2000; Avison and Fitzgerald 2006; Burg 1996; Pfleeger and Atlee 1998). For example, stakeholders describing a task with business concepts may be misunderstood by analysts. Analysts don’t share the business concepts and stakeholders may not clearly express needs because the task is obvious for them.

The social nature of requirements elicitation has led research to the usage of social sciences approaches (Farinha and Mira da Silva 2009; Davidson 1999; Kitzinger 1994; Engelbrektsson et al. 2000). For example, Zowghi and Coulin (2005) aggregated methods in eight groups: prototyping, goals, scenarios, viewpoints, domain, ethnography, interview, and group work. Their categorization was based on their review of the literature, practical experience, and observation in requirements elicitation research and practice. They also presented an understanding of which of the techniques and approaches can be used in cooperation with (complementary) or instead of each other (alternative). Ethnography, interview, and group work are the alternative groups that actively involve stakeholders.

Ethnographers observe people in their natural environment and translate stakeholders' activities and interactions (Crabtree 1998; Dix et al. 2004; Preece et al. 2002). Crabtree et al. (2000) described ethnomethodologically informed ethnography as a methodology for information science research, studying how it may be used to inform systems design. They reveal limitations, including risk of incorrect interpretation, impossibility of identifying new requirements, or difficulty of generalizing results. Sommerville (2007) believes that ethnography is incomplete, being useful as a complement to other requirements elicitation techniques.

Interviewing is an informal interaction (Zowghi and Coulin 2005) where analysts explore needs, asking stakeholders about the system in use and the system to be (Sommerville 2007; Hossenlopp and Hass 2007). Davey and Cope (2008) consider interviews as best practice for requirements elicitation. Despite the improvements that their research results demonstrate, they admit that requirements elicitation is problematic and more research is needed about the nature of conversations in the field. Goguen and Linde (1993) survey and evaluate techniques for eliciting requirements of computer-based systems, including interviews. They consider interviewing limited by the stimulus-response interaction and by the need of participants to share basic concepts and methods. Sommerville (2007) states that interviewing is unsuitable to this activity by its own.

Group work gathers stakeholders to collaborate reaching solutions about an identified problematic situation. Typical limitations of group work are dominant participants, biased opinions, high logistic costs, and difficulties on gathering stakeholders at the same time and place (Zowghi and Coulin 2005; Maté and Silva 2005).

In summary, Zowghi and Coulin (2005) consider three main alternative groups of methods that involve stakeholders in requirements elicitation: ethnography, interview, and work group. Ethnography is incomplete and has relevant limitations, including impossibility of identifying new needs with observation of daily activities and difficulty on generalizing results (Sommerville 2007).

Discussions of group work have advantages over individual interviews: more complete overview, richer information, and possible resolution of conflicts to reach a consensual or agreed solution (Zowghi and Coulin 2005).

2.2 Group Work

There are many group work methods, such as brainstorming and workshops, which include JAD, creativity workshops, and focus groups.

Brainstorming engages different stakeholders in informal discussions to rapidly generate ideas without focusing on any one. It is often used to develop the preliminary mission statement for the project but not to explore requirements (Zowghi and Coulin 2005).

Joint Application Development (JAD) discussions focus on needs of business and users rather than technical issues to make decisions. JAD differs from brainstorms since main goals of the system are already established before the discussion (Zowghi and Coulin 2005). Davidson (1999) studied three organizations in which JAD was used, realizing improvements in systems development. However, JAD was considered difficult to sustain in practice. The author admits that the research doesn't represent all companies using JAD and it is possible that others achieved more substantial benefits. Coughlan and Macredie (2002) also presented two different studies of JAD in practice. Results demonstrate that JAD forces a rigid user-designer interaction and is weak at acquiring knowledge and fairly complicated to use if followed to the letter.

Creativity workshops intend to discover and invent system requirements through creative thinking after the system boundaries specification and before use cases specification. The workshops are incorporated in the RESCUE (Requirements Engineering with Scenarios for User-Centered Engineering) process that integrates different modeling and analysis processes in parallel through concurrent subprocesses (Maiden and Robertson 2005). Maiden and Robertson (2005) used RESCUE creativity workshops to discover stakeholder and system requirements. Results show that the overall process was a success, providing outputs for subsequent requirements processes. Nevertheless, not all of the workshop sessions were a success.

Focus groups are discussion groups facilitated by a specialist who follows a guide to orientate the discussion around key questions (Kitzinger 1994; Engelbrektsson et al. 2000; Krueger and Casey 2000). The preparation of such sessions requires defining groups (size and composition) and procedures (number of sessions and moderator guide). This method differs from other group-based methods because of the group special characteristics: homogeneous and focused on key topics to collect inductive and natural information (Krueger and Casey 2000). Engelbrektssonn et al. (2000) studied methodological considerations with focus groups. They stated that an efficient choice of participants and mediating tools is important to enhance the requirements elicitation activity, but more research is needed to validate the results. Also, Farinha and Mira da Silva (2009) have applied focus groups in real-world experiments to evaluate the use of focus groups to better elicit requirements of information systems. The results show that stakeholders discuss different perspectives about the system as a whole and collaborate to formalize the requirements according to their needs, but dominant users and analysis costs were serious limitations. Kasirun and Salim (2009) proposed a requirements elicitation tool based on a forum that employed focus groups. Results demonstrated that a web-based tool supports shared involvement and eases requirements elicitation, but further research is needed to prove results.

Resuming, many researchers have been studying requirements elicitation problem using social approaches, namely, group work. However, all admit that the problem still exists and much more empirical work is needed (Coughlan and Macredie 2002; Kitzinger 1994; Engelbrektsson et al. 2000).

3 Groupware Systems

Requirements elicitation involves intense communication activities (Burg 1996; Pfleeger and Atlee 1998), demanding a high level of collaboration (Whitehead 2007). Nevertheless, traditional techniques to elicit requirements gather stakeholders at the same time and place, which is difficult (Apshvalka et al. 2009; Avison and Fitzgerald 2006; Davidson 1999; Hossenlopp and Hass 2007), particularly in large projects (Herbsleb 2007; Todd and Cleland-Huang 2008). This limitation (Herbsleb 2007; Whitehead 2007; Todd and Cleland-Huang 2008) and the belief that computer-supported collaborative work allow more flexible and efficient organization (Borghoff and Schlichter 2000) claim to distribute projects globally.

Groupware systems (or collaborative tools) allow the cooperation of all stakeholders in several phases of the software process, as we intended to evaluate, including in requirements engineering. Choosing a groupware system depends on the types of tasks to accomplish, on the inventory of existing software and hardware infrastructures, and on the experience and capabilities of the team (Brown et al. 2006).

Herbsleb (2007) studied a desired global development, arguing that coordination is a key. He considered challenges in several areas, including eliciting and communicating requirements. He concluded that a systematic understanding of what drives the need to coordinate and effective mechanisms for bringing are needed.

Whitehead (2007) presented an overview of the goals of collaboration in software engineering and a brief survey of existing collaboration tools, especially web-based tools. He finds it important to understand the collaborative nature of software engineering and low costs of communications with communication platforms to improve collaboration in the creation of software artifacts.

4 Web-Based Focus Groups

We propose web-based focus groups with all stakeholders. The following sections will explain foundations for our choice and the context of experiments.

4.1 Foundations

Requirements elicitation trends show that group work has potential. Focus groups were selected due to its characteristics: homogenous groups focused on key discussion topics to naturally collect information. Though no selection of participants exists in our proposal, it is a focus group because participants can freely express opinions; questions are focused and also moderated.

Web forums were chosen because of the tasks to accomplish, the inventory of existing infrastructures, and the experience of the team with the collaboration tool.

We expect to achieve a richer and complete overview of the system, promote cooperation to reach consensus over incoming conflicts, decrease the risk of dominant users and biased opinions, gather stakeholders, and reduce logistic costs. We also expect to involve all stakeholders to generalize needs and reduce the complexity of session analysis since no transcriptions are needed.

Our proposal was evaluated with Action Research that allows contributing with practical actions on the organization and generating knowledge about its context on real-world situations. The method requires studying the problematic situation, proposing alternative actions, applying one of the proposed alternatives in practice, and evaluating results (Coghlan and Brannick 2009; Bhattacharjya and Venable 2006; Kock et al. 1997; Koshy 2005).

4.2 Context

The proposal was evaluated on an enterprise of technological solutions. Following Brown et al. (2009), this enterprise develops B2B applications as custom-developed e-business systems, such as e-government systems to the public sector. It has around 60 employees and many clients. Employees' ages range from 23 to 40 years; 83% are men; the majority of them are Information Technology savvy. There is a Chief Information Officer, three Information System Directors, Information System managers, and professionals.

This enterprise has an old in-house Information System to manage its activities, including time reporting, project management, and financials. The time-reporting module is used to report the time that employees spend in a task of a project. The project management module monitors the current status of the projects, including allocated employees or tasks that an employee is performing. Finally, the financials module controls the financial plan of the client's project or contract. IS managers wanted to align these modules to the current needs.

All stakeholders of each module were invited to participate, not only the experts. Hence, time reporting involved the 60 employees of the enterprise, while the other modules involved around 14 employees (managers and directors). One discussion session was realized per module and the moderator guide was adapted to this web-based method. Two different techniques were evaluated:

Table 36.1 Characteristics of the collaborative forums

Collaborative forum	Comment oriented	Vote oriented	
Module characteristic	Time reporting	Project management	Financials
Participation	Comments	Comments and votes	Comments and votes
Participation period	20 days	20 days	20 days
# Questions of moderator guide	8	3	3
Sequential navigation	Yes	No	No
Anonymity	Yes	No	No

1. Comment-oriented collaborative forum
2. Vote-oriented collaborative forum

The comment-oriented collaborative forum consisted on a web-based focus group forum where each page had a question where stakeholders could comment. Following the orientations of a focus group moderator guide, these pages had an ideal sequential order to comment, but opening and ending questions were suppressed. Users were recommended to follow this sequentially but could freely navigate on the forum. Anonymity was integrated to gather real and unbiased opinions.

The vote-oriented collaborative forum consisted on a web-based focus group forum based on stakeholders' comments, votes on others' ideas, and the creation of new discussion topics. The moderator guide started with key questions to avoid an extensive discussion guide. No recommendations about navigation were given. Anonymity was not integrated to compare results with the other forum and because stakeholders had a similar profile, reducing the risk of biased opinions.

These web forums were available for a certain period of time. The moderator guides were written by project managers with topics they considered important (Table 36.1).

Our web-based focus groups didn't need transcriptions. Stakeholders' comments and votes were resumed on a report delivered to IS managers. Needs were enumerated in a descending order, according to the number of references or votes of the comment-oriented collaborative forum or vote-oriented collaborative forum.

5 Results

The comment-oriented collaborative forum involved all employees to identify needs of the time-reporting module. The vote-oriented collaborative forum involved 14 stakeholders (managers and directors) to identify needs and improvements of the modules they use (project management and financials).

The main results are presented in Table 36.2. The comment-oriented collaborative forum had eight questions, around 10.25 comments per discussion topic, and 15 identified needs. Comments were justified, even when comments approved another participant's perspective. Other results were measured from the tracking of the

Table 36.2 Measured indicators

Collaborative forum	Comment oriented	Vote oriented	
Module indicator	Time reporting	Project management	Financials
Comments per topic	10.25	18	11.30
Comments per user	–	4.79	1.36
Votes per user	–	7.14	
Identified needs	15	33	11

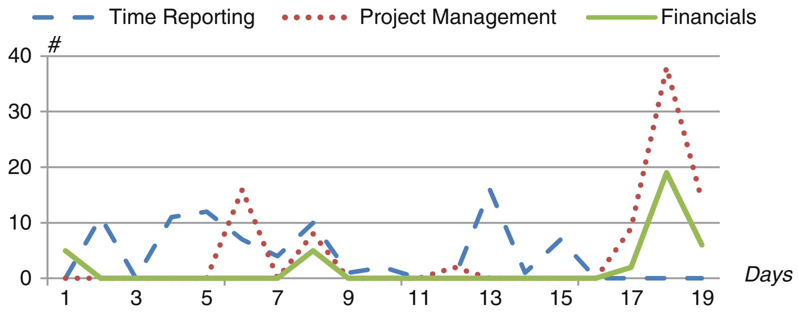


Fig. 36.1 Participation during the discussion period

originating computer and the log files. For example, each participant posted around 1.64 comments meaning that the majority of participants not only posted their perspectives but also came back to comment others' perspectives. Sixteen percent of the invited stakeholders did not participate. Some participants also posted figures and graphics as images obtained from a design tool to exemplify their perspectives.

The vote-oriented collaborative forum had three questions per module, around 18 comments per project management discussion topic, and 11.30 comments per financial discussion topic. No other discussion topics were opened by the participants although they could do it. The average comments per user were 4.79 in the project management module discussion and 1.36 in the financials module discussion. Moreover, the vote system verified around 7.14 votes per user. The project management module had 33 needs identified and the financials module had 11 needs.

Figure 36.1 presents the number of comments during the discussion period, revealing that the comment-oriented collaborative forum had a regular participation over time. The vote-oriented collaborative forum had a higher participation on the last days of the discussion, especially in the last 3 days.

After the discussions, employees were asked about the reasons to participate or not to participate if this was the case. The feedback involved 30% of the participants from the comment-oriented collaborative forum and 50% of the participants from the vote-oriented collaborative forum. Table 36.3 illustrates the results.

Table 36.3 Measured average indicators

Feedback type	Feedback
Nonparticipation reasons	No ideas, lack of time to participate, and recent employees with no experience
Positive aspects	Extensive to all employees, anonymity, simple participation rules, structured discussion, and open comments about discussion topics
Suggested improvements	Present just key discussion topics, give rewards to participants, support the vote system to avoid repeating ideas, suppress suggested answers, no anonymity to avoid unreasonable censures

6 Evaluation

All the stakeholders considered that this initiative was important and relevant.

Results show a higher participation rate in the vote-oriented collaborative forum that involved 14 stakeholders, having more comments per discussion topic in the project management module than in the financial module discussion. The time-reporting discussion had more invited stakeholders but fewer comments per discussion topic. Note that votes counted as participations, which may explain this result since participants prefer the easy voting system than spending time on writing or exposing detailed ideas.

The overall participation rate was fair enough and may be explained by the helping behavior. The participation would help Information System managers of the in-house Information System. Ng and Van Dyne (2005) investigated the helping behavior in work groups and demonstrated that group cohesion and norms of cooperation positively influence the helping behavior. In fact, we observed the unity of employees since they have almost a friendship relationship rather than a coworker relationship. Moreover, cooperative norms are encouraged by directors and managers to increase knowledge. Also Deckop et al. (2003) studied the helping behavior in organizations and suggested that employees help according to the organizational citizenship behavior they received from coworkers. Again, we verified that employees usually expressed readiness to help each other, demonstrating a high organizational citizenship behavior to each other.

The asynchronous communication allowed participants to express perspectives whenever they could, from wherever they were. Moreover, they had extra time to think about their perspectives, perceive others' opinions, and construct new ideas. Also, shy participants felt confident to contribute in an online forum.

Users of both forums did not answer to all of the discussion topics. In fact, key discussion topics were the most commented in the comment-oriented collaborative forum, while judgment of existing modules were the most commented and the most answered in the vote-oriented collaborative forum that followed. This shows that stakeholders prefer to answer key questions and to criticize existing systems. Criticisms are also useful to understand what is wrong and what should be improved.

Our proposal resembles regular threaded discussions (Cameron and Anderson 2006; Rizopoulos and McCarthy 2009). In fact, they have common characteristics, including online and asynchronous focused discussions with a moderator. However, major differences are the collaborative authorship (both the moderator and the participants can add new discussion topics), the choice of anonymity, and the voting system that web-based focus groups can have.

The identified needs were mostly technical requests. For example, users complied about the slow response of the system and about the inexistence of some buttons or certain information. This may be due to the participants' profiles: Information System professionals that understand the problems and tend to express technical opinions. Comments of less technical users were also less technical.

Anonymity promoted free answers but also less participation and conflicts. On the one hand, unexpected criticisms were revealed, bringing conflicting perspectives that were discussed to reach consensus. On the other hand, non-anonymity had more comments per user indicating a commitment sense of identified users. The lack of conflicts may mean that users agreed to each other or that they did not feel free to disagree. Some participants believe that anonymity allows free expression of ideas, while others feel that anonymity brings unreasonable criticisms.

Figure 36.1 shows that the comment-oriented collaborative forum had more regular participation than the vote-oriented one that had a higher participation rate at the end of the period. This fact may be explained with votes that count as participations: at the end of the period, there were more comments to vote. Actually, this voting system helps users not to repeat ideas and to prioritize already identified needs.

Results prove that our proposal helps overcoming identified limitations, including dominant users (simultaneous participations do not steal time of other participants), participation of all stakeholders from different locations and at different times, and generalization of results. It also simplifies the analysis process and avoided translations, forgetting relevant needs or having wrong interpretations.

7 Conclusion

Stakeholders enjoyed expressing ideas in a simple and structured web forum. They preferred not only the voting system but also the discussion of key questions avoiding an extensive discussion guide. Anonymity allows users making censorious comments but discourages disagreements and dismisses the commitment sense. As such, some participants prefer anonymity, while others think that comments should be identified to avoid unreasonable and thoughtless criticism.

This proposal allowed quickly eliciting needs from all stakeholders. Our experiments demonstrate that a web-based focus group method to elicit requirements not only is cheaper and quicker than the conventional focus group but also overcomes many problems of the conventional methods. It also seems well suited to the requirements elicitation activity since it promotes discussion among all stakeholders, gives richer information, and reveals and encourages resolution of conflicts.

Note that the demonstrated improvements are meant to apply not in general domains but in the maintenance/evolution of an existing Information System.

More research work is needed to confirm these results, applying a similar method in other projects. Moreover, from the feedback of stakeholders, other suggestions should be integrated in future web-based focus group sessions. First, rewards to participants should be given to encourage participation. Second, only key discussion topics should be initially provided, allowing addition of new discussion topics. Third, the voting system should always be present. Fourth, suggested answers should be removed so that users do not feel biased to answer.

References

Al-Rawas A, Easterbrook S (1996) Communication problems in requirements engineering: a field study. In: 1st conference on professional awareness in software engineering, Washington, DC, pp 47–60

Apshvalka D, Donina D, Kirikova M (2009) Understanding the problems of requirements elicitation: a human perspective. In: Barry C, Conboy K, Lang M, Wojtkowski G, Wojtkowski W (eds) Information systems development: challenges in practice, theory and education, vol 1. Springer, New York, pp 211–224

Avison D, Fitzgerald G (2006) Information systems development: methodologies, techniques and tools, 4th edn. McGraw-Hill Higher Education, London

Bhattacharjya J, Venable J (2006) An action research approach to strategic information systems planning in a non-profit organization. In: Ruth A (ed) Quality and impact of qualitative research, 3rd annual QualIT conference, Institute for Integrated and Intelligent Systems, Griffith University, Brisbane, pp 29–39

Borghoff UM, Schlichter JH (2000) Computer-supported cooperative work: introduction to distributed applications. Springer, New York

Brown MK, Huettner B, James-Tanny C (2006) Managing virtual teams: getting the most from wikis, blogs, and other collaborative tools. Wordware Publishing, Inc., Plano

Brown CV, DeHayes DW, Hoffer JA, Martin EW, Perkins WC (2009) Managing information technology, 6th edn. Pearson Prentice Hall, Upper Saddle River

Burg J (1996) Linguistic instruments in requirements engineering. IOS Press, Amsterdam

Cameron D, Anderson T (2006) Comparing weblogs to threaded discussion tools in online educational contexts. J Instr Technol Distance Educ 2(11):3–15

Christel M, Kang K (1992) Issues in requirements elicitation, Software Engineering Institute technical report. Carnegie Mellon University, Pittsburgh

Coghlan D, Brannick T (2009) Doing action research in your own organization, 2nd edn. Sage Publications Inc., Los Angeles

Coughlan J, Macredie R (2002) Effective communication in requirements elicitation: a comparison of methodologies. Requir Eng 7(2):47–60

Crabtree A (1998) Ethnography in participatory design. In: Participatory design conference, Seattle, pp 93–105

Crabtree A, Nichols D, O'Brien J, Rouncefield M, Twidale M (2000) Ethnomethodologically-informed ethnography and information system design. J Am Soc Inf Sci 51(7):666–682

Davey B, Cope C (2008) Requirements elicitation – what's missing? Issues Inf Sci Inf Technol (IISIT) 5:543–551

Davidson E (1999) Joint Application Design (JAD) in practice. J Syst Softw 45(3):215–223

Deckop JR, Cirka CC, Andersson LM (2003) Doing unto others: the reciprocity of helping behavior in organizations. J Bus Eth 47:101–113

Dix A, Finlay J, Gregory A, Beale R (2004) Human-computer interaction. Prentice Hall, New York

Engelbrektsson P, Yesil Ö, Karlsson I (2000) Eliciting customer requirements in focus group interviews: can efficiency be increased? In: 7th international product development management conference, Belgium

Farinha C, Mira da Silva M (2009) Focus groups for eliciting requirements in information systems development. In: UKAIS annual conference, p 20

Geisser M, Hildenbrand T (2006) A method for collaborative requirements elicitation and decision-supported requirements analysis. In: Ochoa S, Roman GC (eds) Advanced software engineering: expanding the frontiers of software technology, vol 219, IFIP International Federation for Information Processing. Springer, New York, pp 108–122

Goguen J, Linde C (1993) Techniques for requirements elicitation. In: International symposium on requirements engineering, San Diego, pp 152–164

Herbsleb JD (2007) Global software engineering: the future of socio-technical coordination. In: Future of Software Engineering (FOSE '07), IEEE Computer Society, Los Alamitos, pp 188–198

Hossenlopp R, Hass K (2007) Unearthing business requirements: elicitation tools and techniques. Management Concepts, Inc., Vienna

Kasirun ZM, Salim S (2009) Supporting collaborative requirements elicitation using focus group discussion. J Softw Eng Appl 3(3):59–70

Kitzinger J (1994) The methodology of focus groups: the importance of interaction between research participants. Sociol Health Illn 16:103–121

Kock N, McQueen R, Scott J (1997) Can action research be made more rigorous in a positivist sense? The contribution of an iterative approach. J Syst Inf Technol 1(1):1–24

Koshy V (2005) Action research for improving practice: a practical guide. Sage Publications, Inc., London

Krueger R, Casey M (2000) Focus groups: a practical guide for applied research, 3rd edn. Sage Publications Inc., Thousand Oaks

Maiden N, Robertson S (2005) Integrating creativity into requirements processes: experiences with an air traffic management system. In: 13th IEEE international requirements engineering conference, Paris, pp 105–114

Maté J, Silva A (2005) Requirements engineering for sociotechnical systems. Information Science Publishing, Hershey

Ng KY, Van Dyne L (2005) Antecedents and performance consequences of helping behavior in work groups: a multilevel analysis. Group Organ Manage 30(5):514–540, Sage Publications

Nuseibeh B, Easterbrook S (2000) Requirements engineering: a roadmap. In: International conference on software engineering, Limerick

Pfleeger S, Atlee J (1998) Software engineering: theory and practice, 3rd edn. Prentice-Hall, Upper Saddle River, pp 24–25

Preece J, Rogers Y, Sharp H (2002) Interaction design: beyond human-computer interaction. Wiley, New York

Rizopoulos LA, McCarthy P (2009) Using online threaded discussions: best practices for the digital learner. J Educ Technol Syst 37(4):373–383

Saqid M, Sahid M, Ahmad S (2010) Adding thread during software requirements elicitation and prioritization. Int J Comput Appl 1(9):50–54

Sommerville I (2007) Software engineering, 8th edn. Addison-Wesley, Wokingham

T.S. Group (2009) The Chaos report. The Standish Group

Todd KD, Cleland-Huang J (2008) A dynamic approach for managing stakeholder credentials for lightweight, collaborative, requirements elicitation. In: First international workshop on managing requirements knowledge, pp 42–46

Tsumaki T, Tamai T (2005) A framework for matching requirements engineering techniques to project characteristics and situation changes. In: Situational requirements engineering processes, Paris

Whitehead J (2007) Collaboration in software engineering: a roadmap. In: Future of Software Engineering (FOSE '07), IEEE Computer Society, Los Alamitos, pp 214–225

Zowghi D, Coulin C (2005) Requirements elicitation: a survey of techniques, approaches, and tools. In: Aurum A, Wohlin C (eds) Engineering & managing software requirements. Springer, Berlin, pp 19–46

Chapter 37
Constraint-Driven Approach to Support Input Data Decision-Making in Business Process Management Systems

María Teresa Gómez-López, Rafael M. Gasca, Luisa Parody, and Diana Borrego

1 Introduction

A business process consists of a set of activities that are performed in coordination in an organizational and technical environment (Weske 2007). The base of business process management systems (BPMS) is the explicit representation of business processes with their activities and the execution constraints between them. Compliance rules represent a natural step to include requirements between business functionality and data. For the design of a whole business process management (van der Aalst et al. 2003), it is necessary to design the model of activities and define the causal and temporal relationships between them (Walzer et al. 2008). Compliance rules can help to complete this information, since they can be used to validate business data (Chesani et al. 2008).

Organizations currently need to manage a great deal of data. This can be managed using BPMS, which permit to model tasks and business rules according to a data model. The evaluation of correctness and completeness of values of input data is very important from the point of view of organizations, since neither activity can work correctly using incorrect or corrupted data. But it is especially important when the data are introduced by humans in the instances of a business process.

This chapter takes constraint-driven view of compliance rule engines. We propose to use the constraint programming paradigm to represent and validate the relation between the dataflow variables in compliance rules. It provides a way to find out the possible valid values of the input data that are introduced by humans, in function of the decisions taken in the past, and the compliance rules that have to be satisfiable in the future.

In our proposal, compliance rules are used to analyse the values of the input data and dataflow, in different moments of the business process instance. For this reason,

M.T. Gómez-López (✉) • R.M. Gasca • L. Parody • D. Borrego
University of Seville, Avd Reina Mercedes sn, Seville, Spain
e-mail: maytegomez@us.es; gasca@us.es; lparody@us.es; dianabn@us.es

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_37,

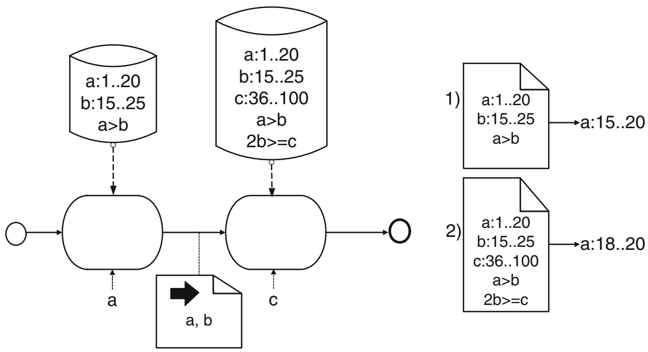


Fig. 37.1 Easy example of business process dataflow for data decision-making support

the use of constraint satisfaction problems for the analysis of the domain of the valid input data is very useful in order to help the user that is introducing input data in the process about the possible correct values of the input variables. The instantiation of the variables, which represent the data in an instantiation process, changes during the process execution; hence, the valid ranges of the variables will change too. In order to introduce the idea of this chapter, in Fig. 37.1 an easy example is depicted, where a very simple business process model is shown. In this process, the value of data a is introduced by humans in the first activity, and the value of variable b is an output value of an activity. In order to decide the valid values of the variable a, we can take into account the compliance rules related only with a and b (option 1), or all the compliance rules related directly or indirectly with a and b (option 2). Option (1) presents the possible values of the variable a if only the compliance rules related to the first activity are analysed, and option (2) shows how the possible valid values of a are reduced if the compliance rules related to the variable c are included in the analysis as well. For example, in option 1, it is possible to introduce the value 16 for the variable a and the value 15 for the variable b, being satisfiable that $\{a > b\}$, but there is no valid values for c to satisfy the compliance rule $\{2b \geq \mathrm{c}\}$. Then, the use of compliance rules to represent the relations between dataflow and input variables can help in the introduction of correct values by humans.

The use of compliance rule in a prognosis way can be decisive, since sometimes the humans take decisions in the business processes that result incorrect and when the nonconformities are identified too late. In order to know the valid range of the input variables, we use as compliance rule language and engine the constraint programming paradigm proposed in Teresa Gómez-López and Gasca [2008].

This chapter is organized as follows: Sect. 2 discusses some previous works related to our proposal. Section 3 presents the necessary definitions related to the proposal and an example where decision-making support has been used. Section 4 presents how constraint satisfaction problems can be used in decision-making support. Finally, conclusions and future work are presented.

2 Related Works

The importance of compliance data verification has been the focus of attention for numerous approaches, although most of them have analysed the compliance checking part of process model structure (Sadiq et al. 2005; Ly et al. 2008), or are related to mode checking (Liu et al. 2007; Awad et al. 2008,). Related to how to model data-aware compliance rules, works as Liu et al. [2007], Weber et al. [2008], Ly et al. [2010] and Awad et al. [2011], have defined graphical notations to represent the relationship between data and compliance rules by means data conditions. Also in Knuplesch et al. [2010] a preprocessing step to enable data-aware compliance checking in an efficient manner was presented. Governatori et al. [2008] introduces an approach for semantically annotating activities with preconditions and effects that may refer to data objects, and an efficient algorithm for compliance verification using propagation is also discussed. In contrast to this approach, we focus on analysing the validation on the input data in a business process instance. Most works are related to design and execution business process correctness. Our proposal starts with a correct designed process where an incorrect input instantiation of variables can produce that it works incorrectly.

In zur Muehlen and Indulska [2010], there is a depth analysis about the integration of rules and process modelling and the shortcomings of the existing solutions. Our work is based on Meng [2002], although we propose to separate the evaluation into an independent layer that checks the compliance rules oriented used as a contract that describes the behaviour of the activities in different moments of the business process instance, taking into account the compliance rules evaluated in the future.

The correctness of the values in business processes has been analysed in other proposals, for example, in Rajbhandari et al. [2008], where the "truth degree" of the data can be described by using fuzzy terms. But the mentioned proposal neither allows the monitoring of the process nor the compliance rules analysis. Our proposal is related to the monitoring idea developed in Beeri et al. [2007], which is oriented towards the definition of a monitoring language, although no data decision-making support is presented.

3 Formalization of Data Decision-Making Support for Business Processes

A business process model can be composed by using activities, compliance rules, decision rules and data objects from a process domain. In order to clarify the explanation, we use the well-known example of the organization of a conference, where a reduced model is presented in Fig. 37.2. This business process shows an example where decisions must be taken when there are many unknown variables.

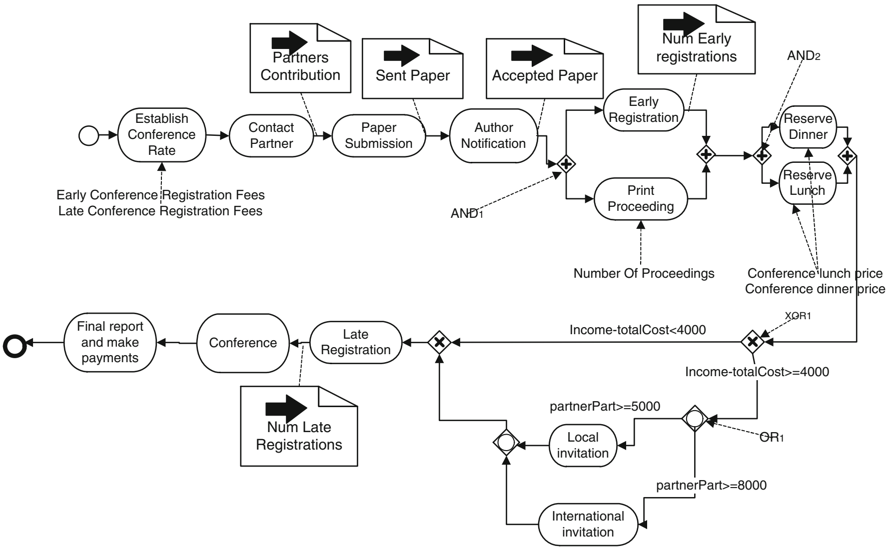


Fig. 37.2 Example of a conference organization process

First of all, the conference committee has to determine the early and late registration fees several months before the number of participants is known, and this decision cannot be changed after the call for paper has been done. Something similar happens to decide the number of proceedings that will be printed, where the final number of participant is unknown. Other decisions, such as to reserve the restaurant for dinner and lunch, have to be taken although the number of participants can have influence in the determination of the price and the restaurant. Although this information is unknown, it is necessary to make a reservation some months before the conference starts. In the example, five different input data participate in the data decision-making process (early conference registration fees, late conference registration fees, conference lunch price, conference dinner price and number of proceeding to print). There are other input variables, but their values are determined in a mandatory way, for example, the number of accepted paper or the final number of participants that belong as part of the dataflow. Obviously, these decisions can be taken for the conference committee, but how the committee can take into account all the possibilities that can happen and the possible branches that will be executed in function of the concrete values of each process instantiation?

The variables related in this example are:

totalCost, Income[0.. +∞] is the total cost and income of the conference, respectively
numParticipantEarly[50..200] is the number of assistant registered in the early period
numParticipantLate[10..100] is the number of assistant registered in the late period
numParticipant[0.. +∞] is the final number of participants
costPerParticipant[0.. +∞] is the cost per participant
EarlyRegistrationFees[100..800] is price of the early registration
LateRegistrationFees[200..1000] is price of the late registration
incomePartner[1000..10000] is the income of the companies whose sponsor the event
dinnerPrice[60..250] is the price of the dinner
lunchPrice[30..100] is the price of the lunch
proceedingPrice[5..20] is the price of print one copy of the proceeding
NumOfProceedings[0.. +∞] is the number of proceedings printed
fixCost[2000..3500] is the price of rent the hotel, publicity, cartels...
inviteSpeaker[0.. +∞] is the price of invite a speaker to the conference
inviteLocalSpeaker[0..2000] is the price of invite a local speaker to the conference
inviteInterSpeaker[0..4000] is the price of invite an international speaker to the conference

Although the values of the input data are unknown before the conference process starts, the committee has information about previous similar conferences. This information can be represented as constraint compliance rules to help in the datadecision-making process:

```
Cost: totalCost = numParticipat ∗ CostPerParticipant + fixCost +
   inviteSpeaker + 0.10 ∗ numParticipant*proceedingPrice
numParticipantC: numParticipant = numParticipantEarly
   + numParticipantLate
IncomeC: Income = numParticipantEarly*EarlyRegistrationFeesy
   + numParticipantLate*LateRegistrationFees + incomePartne
RelationPrices:
1.2*LateRegistrationFees ≤ EarlyRegistrationFees
   ≤ 1.5*LateRegistrationFees
RelationNumParticipant: 1.2* numParticipantEarly ≥ numParticipant
   ≤ 1.4* numParticipantEarly
LunchRelation: costPerParticipant*0.10 ≤ lunchPrice*3
   ≤ costPerParticipant*0.35
DinnerRelation: costPerParticipant*0.10 ≤ dinnerPrice
   ≤ costPerParticipant*0.60
```

(continued)

(continued)

```
ProceedingRelation:
costPerParticipant*0.05 ≤ proceedingPrice
  ≤ costPerParticipant*0.15
CostPerParticipantC: CostPerParticipant=3*lunchPrince
  +dinnerPrice+proceedingPrice
InvitationSpeakers: inviteSpeaker=inviteLocalSpeaker+inviteInter
Speaker
InvitationLocalSpeakers: 500 ≤ inviteLocalSpeaker ≤ 2000
InvitationInterSpeakers: 1000 ≤ inviteInterSpeaker ≤ 4000
Goal: Income*0.80 ≤ TotalCost ≤ Income
```

The constraints *InvitationLocalSpeakers* and *InvitationInterSpeakers* only are used if the activities *Local invitation* and *International invitation* are executed, respectively.

In this chapter, the concrete scenario is defined by means of a process formed by the following:

- $\mathbb{A}$ is a set of activities that defines the model of the process.
- $\mathbb{UA}$ is the set of user activities that is a subset of the activities $\mathbb{A}$ ($\mathbb{UA} \subseteq \mathbb{A}$) that are activities where a user performs the activity with the assistance of a software application and is scheduled, using the terminology proposed in BPMN 2.0 (OMG 2011).
- $\mathbb{CF}$ is a set of control flows (AND, OR, XOR, …) that describes the relation between the activities.
- $\mathbb{DO}$ is the set of data objects that flows in the process, each of them are defined as an input and/or output data of an activity.
- $\mathbb{ID}$ is the set of input variables introduced by users in the different $\mathbb{UA}$.
- $\mathbb{CR}$ is a set of compliance rules that can involve dataflow variables and/or input variables of the process used to validate the correctness of the process.
- *Scope*($\mathbb{CR}$) are the activities where $\mathbb{CR}$ can be evaluated after executing them because all the data objects involved in the $\mathbb{CR}$ are instantiated.
- $\mathbb{DR}$ is a set of decision rules associated to the control flows OR or XOR that describes the decisions that the process can take depending of the values of the variables in the dataflow.
- $\mathbb{D}$ is the set of data domains, where dom: $\mathbb{DO} \rightarrow \mathbb{D}$.

Considering the process example presented in Fig. 37.2, the components of the process following the previous definitions are:

- $\mathbb{A}$ = {Establish Conference Rate, Contact Contribution, Paper submission, Author Notification, …}.
- $\mathbb{UA}$ = {Establish Conference Rate, Print Proceeding, Reserve Dinner, Reserve Lunch}.
- $\mathbb{CF}$ = AND_1, AND_2, XOR_1, OR_1.

- $\mathbb{DO}$ = {Output(Contact Partners) = {Partner Contribution}, Output(Paper submission) = {Sent paper}, Output(Author Notification) = {Accepted papers}, Output(Early Registration) = {num Early Registrations}, Output(Late Registration) = {num Late Registrations}.
- $\mathbb{CR}$ are {totalCost[0 ... + ∞], numParticipantEarly[50 ... 200],..., invitedInterSpeaker[1000 ... 4000], ..., Cost, numParticipantC, ..., Goal }
- Some examples of Scope are: Scope(EarlyRegistrationFees[100 ... 8000])= {Establish Conference Rate, Contact Contribution, Paper submission, Author Notification, ...}, Scope(InvitationSpeakers)={Local invitation, International invitation} or Scope(lunchRelation)= {Reserve lunch}, Scope(RelationNumParticipant) = {Late Registration}.
- $\mathbb{DR}$ of the control flow XOR_1 are $\{Income - totalCost < 4000$, $Income - totalCost \geq 4000\}$, for the control flow OR_1 are {partnerParticipant $\geq$ 5000, partnerParticipant $\geq$ 8000}.
- $\mathbb{D}$ for the variables in the example is integer.

3.1 Compliance Rules Representation by Numeric Constraints

Although in the previous section compliance rules have been used, the syntax used in this chapter has not been introduced yet. In order to add compliance rules to a business process related to its data, it is necessary to add semantic information about the data correctness.

If the expressiveness of compliance rules is improved, the mentioned above characteristics will be improved too. For this reason, we propose the use of *Numeric Constraints* instead of *if ...then* axiom. The *Constraints* that we propose to define compliance rules can be expressed with the following grammar where `Variable` and `Value` can be defined for integer, natural or float domain:

```
Constraint := Atomic_Constraint BOOL_OP Constraint
  | Atomic_Constraint
  | '¬'Constraint
BOOL_OP:= '∨' | '∧'
Atomic_Constraint:= function PREDICATE function
function:= Variable FUNCTION_SYMBOL function
  | Variable
  | Value
PREDICATE:= '=' | '<' | '≤' | '>' | '≥'
FUNCTION_SYMBOL:= '+' | '−' | '∗'
```

These constraints make easier and more precise to handle numeric data (that can be represented as variables) that represent relations between variables. The use of constraints to represent business rules extends the formal semantics of business rules, since more knowledge can be inferred and the description has less limitation than using decision trees or a set of facts. For example, it is possible to add the following compliance rule:

$$(a + b + c = d \wedge c \leq a{*}0.10) \vee (a + b + c < d \wedge c \leq a_{*}0.15)$$

where

$$a[1 \ldots 100], b[1 \ldots 150], c[1 \ldots 100], d[5 \ldots 250]$$

`for Integer domain`

By using constraints to represent compliance rules, it is possible to infer new knowledge that is not explicitly described and although not all the related variables (data) are instantiated. Some examples of the inferred compliance rules are:

- $a \leq d$, $b \leq d$, $c \leq d$
- $c \leq d * 0.10$
- If $a = 10$ then $d[12...250] \wedge c = 1$

The knowledge of the third type has great quantity of possibilities, depending of the instantiation of the variables, and it is possible to evaluate a constraint although not all the variables are instantiated. It permits a decision-making task before all the variables are instantiated.

Also, the use of constraints enables integrity rules, derivation rules, reaction rules and production rules to be represented, and the evaluation of whether a set of data is correct for a business policy. The same rule can be of different types depending on the instantiated and known variables. For example:

$$a > b \wedge a + b = c$$

- a, b and c known: It transforms the constraint compliance rules in an integrity business rule.
- a and b known: It transforms the constraint compliance rules in a production business rule.
- a and c known: It transforms the constraint compliance rules in a derivation or reaction business rule, where b is obtained.

It permits to reuse the same compliance rules, avoiding the rewriting of them in different locations of the process and for decision-making support to know the possible valid values of the variables before they are instantiated as it is explained in the next section.

3.2 Dataflow Early Validation Using Compliance Rules

The adoption of business rules adds another tier to systems that automate business processes. Comparing the use of business rules to traditional systems, business rules approach has the following major advantages, analysed in a deeper way in Weber et al. [2009]: lowers the cost incurred in the modification of business logic; shortens development time; rules are externalized and easily shared among multiple applications; changes can be made faster and with less risk.

Although one of the features of compliance rules is the separation of business policy from process control flow, not all the compliance rules have to be related to the whole business process (McDermid 2003), but as it has been commented, it would be a good idea in data decision-making support to analyse the compliance rules that will be involved in the future depending on the model of the process.

We propose to change the initial definition of scope of a compliance rules presented in Sect. 3, checking the valid values before they are instantiated. In this chapter, we change the definition of scope data proposed in Delcambre et al. [2005], where scope data is described as data elements that can be defined which are accessible by a subset of the tasks in a case. We propose to enlarge the scope data definition to compliance rules:

$Scope(\mathbb{CR})$ are the activities where $\mathbb{CR}$ can be evaluated after execute them because some data objects involved in the $\mathbb{CR}$ are *related*.

If to know the possible valid values of *Early Registration Fees* and *Late Registration Fees*, only the constraints related to the range of the variables are taken into account, values as *Early Registration Fees* = 120 can be selected. And in future activities, the programme committee will realize that this was a wrong decision. For the example, the domain obtained for *Early Registration Fees* and *Late Registration Fees* are [134 ...800] and [201 ...960], respectively, that are obtained analysing not only the compliance rules related to the variables, all the constraints related to the variables related to them.

Then, in each, *User Activity* is necessary to know which compliance rules have to be included in the decision-making process and how the model of the process and the decision rules affect to each decision. In this section, we explain which compliance rules are related to each activity and how the model of the process can affect to the decision problem. How to obtain the valid values is introduced in the next section.

As we have introduced, the scope of a compliance rule is the set of activities where this compliance rule can be evaluated. It means that the compliance rules that can be analysed in each activity (EvaluatedCR) are the compliance rules where it is involved, in a formal way as:

If $\text{Scope}(\mathbb{CR}_1)=\{A_i, \ldots, A_j\} \wedge \text{Scope}(\mathbb{CR}_n)=\{A_k, \ldots, A_l\}$
then $\text{EvaluatedCR}(A_l)=\{\mathbb{CR}_1 \cup \ldots \cup \mathbb{CR}_m\}$
where $A_l \in \text{Scope}(\mathbb{CR}_1) \vee \ldots \vee A_l \in \text{Scope}(\mathbb{CR}_n)$.

4 Data Decision-Making Support by Using Constraint Programming

As it was commented in Sect. 3, the information between activities is transmitted by dataflow, and this dataflow has to be analysed by compliance rules. Then, it can be used to obtain a reduction of the domains of the variables that, will be valid. In order to know that, we propose the use of constraint satisfaction problems.

Constraint satisfaction problems (CSPs) represent a reasoning framework consisting of variables, domains and constraints. Formally, it is defined as a triple $< X, D, C >$ where $X = \{x_1, x_2, \ldots, x_n\}$ is a finite set of variables, $D = \{d(x_1), d(x_2), \ldots, d(x_n)\}$ is a set of domains of the values of the variables, and $C = \{C_1, C_2, \ldots, C_m\}$ is a set of constraints. Each constraint C_i is defined as a relation R on a subset of variables $V = \{x_i, x_j, \ldots, x_k\}$, called the *constraint scope*. The relation R may be represented as a subset of the Cartesian product $d(x_i) \times d(x_j) \times \ldots \times d(x_k)$. A constraint $C_i = (V_i, R_i)$ specifies the possible values of the variables in V simultaneously in order to satisfy R. Let $V_k = \{x_{k_1}, x_{k_2}, \ldots, x_{k_l}\}$ be a subset of X, and an l-tuple $(x_{k_1}, x_{k_2}, \ldots, x_{k_l})$ from $d(x_{k_1}), d(x_{k_2}), \ldots, d(x_{k_l})$ can therefore be called an *instantiation* of the variables in V_k. An instantiation is a solution only if it satisfies the constraints C.

In order to solve a CSP, a combination of search and consistency techniques is commonly used (Dechter 2003). The consistency techniques remove inconsistent values from the domains of the variables during or before the search. Several local consistency and optimization techniques have been proposed as ways of improving the efficiency of search algorithms.

In this case, the CSPs will be formed by the variables of the dataflow with the restricted domains defined by the expert and/or refined by previous decisions, and the compliance rules represented by numeric constraints that can be evaluated in each moment (EvaluatedCR), but how to create the CSP depends on the topology of the business process model.

As we have commented, all the compliance rules can influence in the possible values of a data, but in which sense will be defined by the topology of the business process model and the compliance rules associated to the control flows. In this chapter, the most common control flows are analysed (AND, OR and XOR). For a sequence of activities or an AND split operator appear, all the constraints related to each of them will be included in the CSP with the AND operator (Fig. 37.3a, b). In the case of XOR control flow, where only one branch can be executed, the constraints of each branch will be combined with the constraint of the control flow and its negation, respectively (Fig. 37.3c). And for OR control flow, each constraint associated to each branch is combined with the constraints of the activities of the branch (Fig. 37.3d).

For the example of Fig. 37.2, the CSP built to analyse the possible valid values is:

```
Income[0..+∞] Integer
totalCost[0..+∞] Integer
...
InviteInterSpeaker[0..4000] Integer
totalCost = numParticipat ∗ CostPerParticipant + fixCost +
inviteSpeaker + 0.10 ∗ numParticipant*proceedingPrice ∧
numParticipant = numParticipantEarly + numParticipantLate ∧
...
Income*0.80 ≤ TotalCost ≤ Income
(Income-totalCost < 4000) ∨
((Income-totalCost ≥ 4000) ∧
  (partnerPart ≥ 5000 ∧ (500 ≤ inviteLocalSpeaker ≤ 2000)) ∧
  (partnerPart ≥ 5000) ∧ (1000 ≤ inviteInterSpeaker ≤ 4000)))
```

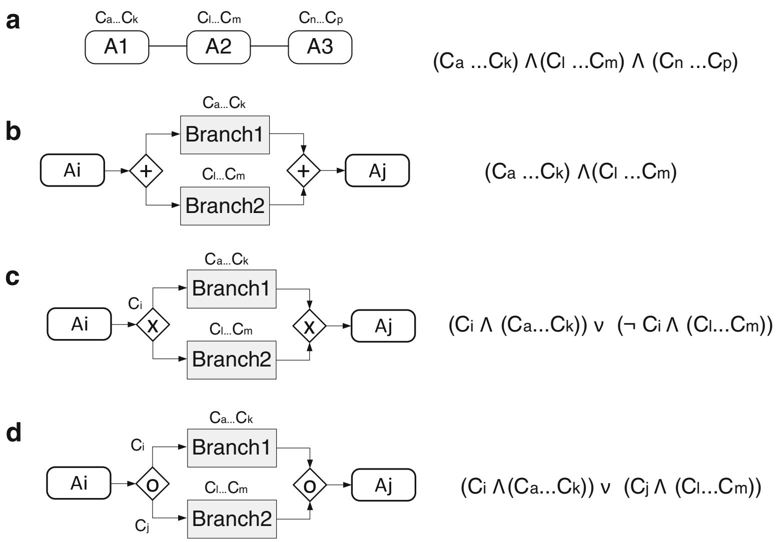


Fig. 37.3 Constraints of the CSPs depending on the business process model

Once the CSP has been solved, the minimum and maximum values for each variable will be obtained. The obtained information can be used to the corresponding business process activity.

5 Conclusions and Future Work

In this chapter the use of constraints to represent compliance rules has been proposed. This representation has been also used to define a decision-making support that provides the valid values of variables when some decisions have to be taken before all the information is known. In order to know the valid values, CSPs have to be created and solved dynamically according to the business model.

These ideas have been developed only taking one instance into account, but the parametrization of compliance rules can be improved learning from previous instantiations. It can also be interesting to provide a way to detect which compliance rules are not satisfiable because their definitions are not correct. Also we propose as future work to extend the type of constraints to handle enumerated domains.

Acknowledgements This work has been partially funded by Junta de Andalucía by means la Consejería de Innovación, Ciencia y Empresas (P08-TIC-04095) and by the Ministry of Science and Technology of Spain (TIN2009-13714) and the European Regional Development Fund (ERDF/FEDER).

References

Awad A, Decker G, Weske M (2008) Efficient compliance checking using bpmn-q and temporal logic. In: BPM, Milan, pp 326–341

Awad A, Weidlich M, Weske M (2011) Visually specifying compliance rules and explaining their violations for business processes. J Vis Lang Comput 22(1):30–55

Beeri C, Eyal A, Milo T, Pilberg A (2007) Monitoring business processes with queries. In: VLDB'07: proceedings of the 33rd international conference on very large data bases, VLDB endowment, Vienna, pp 603–614

Chesani F, Mello P, Montali M, Riguzzi F, Sebastianis M, Storari S (2008) Checking compliance of execution traces to business rules. In: Business process management workshops, Milan, pp 134–145

Dechter R (2003) Constraint processing. The Morgan Kaufmann series in artificial intelligence. Morgan Kaufmann, San Francisco

Delcambre LML, Kop C, Mayr HC, Mylopoulos J, Pastor O (eds) (2005) Conceptual modeling – ER 2005, 24th international conference on conceptual modeling, Klagenfurt. Lecture notes in computer science, vol 3716. Springer, Berlin

Governatori G, Hoffmann J, Sadiq SW, Weber I (2008) Detecting regulatory compliance for business process models through semantic annotations. In: Business process management workshops. Lecture notes in business information processing, vol 17. Springer, Berlin, pp 5–17

Knuplesch D, Ly LT, Rinderle-Ma S, Pfeifer H, Dadam P (2010) On enabling data-aware compliance checking of business process models. In: ER, Vancouver, pp 332–346

Liu Y, Müller S, Xu K (2007) A static compliance-checking framework for business process models. IBM Syst J 46(2):335–362

Ly LT, Rinderle S, Dadam P (2008) Integration and verification of semantic constraints in adaptive process management systems. Data Knowl Eng 64(1):3–23

Ly LT, Rinderle-Ma S, Dadam P (2010) Design and verification of instantiable compliance rule graphs in process-aware information systems. In: CAiSE, Hammamet, pp 9–23

McDermid DC (2003) Integrated business process management: using state-based business rules to communicate between disparate stakeholders. In: Business Process Management, Eindhoven, pp 58–71

Meng J (2002) Achieving dynamic inter-organizational workflow management by integrating business processes, e-services, events, and rules. Ph.D. thesis, Gainesville. Chair-Su, Stanley Y. and Chair-Helal, Abdelsalam

OMG (2011) Business process model and notation (bpmn) 2. 0 reference manual

Rajbhandari S, Rana OF, Wootten I (2008) A fuzzy model for calculating workflow trust using provenance data. In: Proceedings of the 15th ACM Mardi Gras conference: from lightweight mash-ups to lambda grids: understanding the spectrum of distributed computing requirements, applications, tools, infrastructures, interoperability, and the incremental adoption of key capabilities, MG '08. ACM, New York, pp 10:1–10:8

Sadiq SW, Orlowska ME, Sadiq W (2005) Specification and validation of process constraints for flexible workflows. Inf Syst 30(5):349–378

Teresa Gómez-López M, Gasca RM (2008) Run-time monitoring and auditing for business processes data using constraints. In: International workshop on business process intelligence, BPI 2010. Springer, Berlin, pp 15–25

van der Aalst WMP, ter Hofstede AHM, Weske M (2003) Business process management: a survey. In: Business process management, Eindhoven, pp 1–12

Walzer K, Breddin T, Groch M (2008) Relative temporal constraints in the rete algorithm for complex event detection. In: DEBS '08: proceedings of the second international conference on distributed event-based systems. ACM, New York, pp 147–155

Weber I, Hoffmann J, Mendling J (2008) Semantic business process validation. In: SBPM'08: 3rd international workshop on semantic business process management at ESWC'08, Tenerife

Weber B, Sadiq SW, Reichert M (2009) Beyond rigidity – dynamic process lifecycle support. Comput Sci R&D 23(2):47–65

Weske M (2007) Business process management: concepts, languages, architectures. Springer, New York/Secaucus

zur Muehlen M, Indulska M (2010) Modeling languages for business processes and business rules: a representational analysis. Inf Syst 35(4):379–390

Chapter 38
Modelling Business Transactions Across Service Supply Chain Networks

Noel Carroll, Rafiqul Haque, Eoin Whelan, and Ita Richardson

1 Introduction

In today's service-dominant business environment (Normann 2001; Spohrer et al. 2007), harnessing innovative applications of technology is considered one of the critical factors towards organisational sustainability. In addition, with the emergence of 'Cloud Computing', understanding the application and indeed '*value*' of technology to support infrastructure as a service (IaaS) is becoming increasingly more important. Consequently, the application of technology to support services has altered our traditional understanding of the 'organisation', making it more difficult to conceptualise the paradigm of services. Thus, the emergence of 'service science' as a discipline has underscored the importance of understanding the complexities of service and their intertwining properties (Normann 2001; Chesbrough and Spohrer 2006; Spohrer et al. 2007; Spohrer and Maglio 2009). In doing so, the discipline of service science has received much interest, although little attention has been placed thus far, on exploring the contribution of information technology (IT) within a service network and the impact of IT on relational structures which support service networks. To align with this recent paradigm, we propose the need to expand on modelling techniques to accommodate for the 'modern' and digitised service environment by introducing a business transaction language (BTL). The development of BTL attempts to gain a more fundamental insight of the business transaction model which specifies the elements of a business transaction from an application perspective. As this chapter

The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 under grant agreement 215483 (S-Cube). It was supported, in part, by Science Foundation Ireland grant 03/CE2/I303_1 to Lero – the Irish Software Engineering Research Centre (www.lero.ie).

N. Carroll (✉) • R. Haque • E. Whelan • I. Richardson
Department of Computer Science and Information Systems, Lero – The Irish Software Engineering Research Centre, University of Limerick, Limerick, Ireland
e-mail: Noel.Carroll@lero.ie; Rafiqul.Haque@lero.ie; Eoin.Whelan@ul.ie; Ita.Richardson@lero.ie

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_38,

will demonstrate, a business transaction is a series of collaborative activities distributed across the multiple partners in service networks that explicitly enforce the achievement of an agreed-upon business objective in end-to-end processes (S-Cube 2010). Understanding how transactions are influenced by the service architecture and vice versa is of critical importance in the service environment, as we discuss throughout this chapter.

2 Modelling Business Transactions

Modelling business transaction denotes specifying the structural and behavioural aspects of transactions that underpin the applications such as Service-Based Application (SBA). Structural aspect deals with the construction of transaction such as nested structure. In this research, we do not concentrate on structural aspect because a list of transaction models such as distributed transaction as suggested by Gray and Reuter (1993) and open nested transaction model (Traiger 1983) are structurally rich enough to be used for business transactions in service network environment. Our focus is rather on behavioural aspect which deals with the operations including process level operations (e.g. refund) and system level operations (e.g. commit). In this section, we discuss different issues that are related to business transaction behaviour.

Behavioural aspect of business transaction in service network involves business perspective that entails real-world business elements. Within service network environment, businesses are collaborative representations among partners which are achieved through a mutual understanding on obligations and guarantees in terms of qualities, actions, etc. This coalition (mutual relationship) is formally known as *agreement* which is an element of business perspective. The collaborative partners have to comply with the agreement while carrying out the transactions. Thus, agreement is considered as one of the primary determinants of transaction behaviours in service network environments. In particular, the obligations in terms of quality of service contained in agreements may promote the failure atomic behaviour of transactions. For example, a buyer may cancel an order if the seller fails to satisfy the agreed delivery lead time; this results the failure of list of transactions that have already committed successfully. This behaviour is similar to the notion of atomicity. The example shows how agreement influences the failure atomic behaviour of business transaction. Additionally, obligations in terms of actions (service operations) are a set of rules that impose constrains on service transaction. These rules must be satisfied while performing transactions independently by collaborating partners.

Furthermore, vitality is an attribute of business process activities (note that activity specifically a composite activity is also called *service*) which influences the behaviour of business transaction in service networks. Some participating services in an SBA may be vital that must commit successfully because the successful commit of a business transaction depends on the commit of each of these vital services. For example, the successful commit of a business transaction depends on the commit of *order processing*, *payment*, and *delivery* services. Failure of any of these activities fails the transaction entirely. In addition, if the transaction of a vital service fails, transactions of all the services that are *dependent* on the vital one also fail. This indicates that dependency is an important issue for business transaction behaviour.

Transaction behaviour is also determined by a list of properties. The widely known transaction properties are atomicity, consistency, isolation, and durability, shortly called ACID (Harder and Reuter 1983). The database application relies heavily on these properties, but they (specifically atomicity and isolation) are not suitable for complex applications (Gray and Reuter 1993; Chrysanthis and Ramamritham 1990) such as SBAs. Thus, the transaction models (e.g. Nested, Multilevel, and Flat) that are built on these properties lack the efficiency when used for new applications (Chrysanthis and Ramamritham 1990; Garcia-Molina and Salem 1987) such as SBAs. The strictness of these properties is the primary reason for inefficiency. More specifically, atomicity does not allow any failure in transaction, and isolation uses two-phase locking protocol, which degrades the performance of business transaction (Gray and Reuter 1993). Nonetheless, flexibility is an important metric to measure efficiency in business transaction. In a word, flexible behaviour is highly desired in business transaction for service network.

Decidedly, a business transaction model should (1) incorporate business elements, (2) facilitate specifying the attributes of business process activities, and (3) ensure flexible behaviour. How to define such business transaction model? Traditional approach in particular hardwiring business elements and attributes in application logic is not feasible for transactions in service network environment due to many reasons predominantly managing the runtime changes. In service network environment, services may need to be integrated with running application *on the fly* because of the demand; this is called Just-in-Time (JIT) integration. JIT integration is advantageous for SBAs that support end-to-end business processes in service network because it is hard to perceive the service demand in advance. However, the integration of new service may require changes in transactional attributes. For instance, if a shipper is replaced by a new shipper, the transactional attributes (e.g. delivery time) may need to be changed. In addition, traditional transactions are defined by the application programmer, but the participation of business personnel (e.g. business analysts) is enormously important especially for business transactions in service network environment. Taking these issues into consideration, we proposed a design-time business transaction language (BTL) (S-Cube 2010) in the earlier phase of this research. In this chapter, we present the recent development of BTL.

Before going into the details about BTL, we discuss our contribution to ensure flexible transaction behaviour. For flexible transaction behaviour, we refine the classical transaction properties in particular, atomicity and isolation. We do not consider consistency and durability within the scope. Table 38.1 shows these properties with a brief description of each.

BTL allows specifying the above properties. BTL is a declarative language that is used to model business transactions in service networks. It is declarative because it describes *what* (transactional elements) should be defined in a business

Table 38.1 List of business transaction properties

Transactional properties		Description
Behavioural	Flex-atomicity	Atomicity cannot be fully compromised due to the fact that complete non-atomicity may increase the transaction failure rate exponentially. But flexibility guarantees that a failure of a transaction can be repaired using contingent transaction. We also call this flex- atomicity as *eventual failure atomicity* which denotes a transaction is failed entirely only when it is beyond to be repaired
	Relaxed-isolation	Relaxed-isolation determines the visibility of outcomes. The classical isolation property does not allow externalising the outcome of a child transaction until the parent transaction commits. Relaxed-isolation ensures the externalisation of every change that happens after operation. For instance, the transition of the state of transactional activity is notified to all of its collaborating partners at once. Relaxed-isolation is prone to inconsistency. In fact, it is the trade-off between consistency and performance. In service network environment, performance is a primary factor to gain customer satisfaction. Thus, relaxed-isolation may be more suitable than the classical isolation by tolerating inconsistency at lower extent

transaction model but not *how* they should be implemented. The implementation of BTL is not within the scope of this chapter. However, BTL comprises three main perspectives embodying *business perspective, functional perspective,* and *protocol perspective*.

BTL facilitates capturing business-, functional-, and protocol-related elements through these perspectives. The code snippet in Listing 38.1 shows the fundamental structure of BTL. Business transaction is the central entity that is lined with all these perspectives using different relational semantics. We claim BTL a novel approach owing to its ability to capture the business elements that are critical for business transactions in service network.

SLA is the primary element of business perspective, which influences the behaviour of business transactions in service network. It is formal agreement between two and many partners. SLA contains a set of obligations and service guarantees. *Policy* and *quality of service (QoS)* are the constructs used to capture the obligations and service guarantees, respectively. The participants are agreed on such obligations and guarantees that must be satisfied while performing business transactions. The functional perspective of BTL entails business functions (which are in fact composite services or processes). Business functions are composed of events and activities. An activity is an abstraction of a piece of work that has to be performed within a process, whereas an event denotes an occurrence or something has happened in a process (Leymann and Roller 2000). Since flex-atomicity is a desired property of business transaction in service network, BTL allows specifying contingency activity and compensating activity. The key purpose of these activities is to prevent the total failure of business transactions. Listing 38.2 shows the code snippet of business and functional elements.

```
BTL>
       <BusinessPerpsective>
             <! -- Business perspective contains the
             business related elements -->
             <element> name = "" </element>
       </BusinessPerpsective>
       <FunctionalPerpsective>
             <! -- Business perspective contains
             function related elements such as
             activities -->
             <element> name = "" </element>
       </FunctionalPerpsective>
       <ProtocolPerspective >
             <! -- protocol description contains list
             of Protocols including business and
             technical -->
             <element> name = "" </element>
       </ProtocolPerspective>
  </BTL>
```

Listing 38.1 The code snippet shows thee aspects of BTL

In a service network, business transactions happen through executing business functions in an ordered manner. For instance, a purchase order business transaction may follow the execution order of business functions in following sequence: order registration, payment, and delivery. In contrast, the delivery transaction may happen before payment, or payment can be split into before and after delivery. The execution order is important to the business transactions in service network since they ensure correctness of the business transactions. This order is maintained using business protocols. Business protocol defines the ordering in which a particular partner sends messages to and expects messages from its partners based on an actual business context and captures all behavioural aspects that have cross-enterprise business significance (Papazoglou 2003). BTL includes protocol perspective that allows specifying the business protocol, in other words, specifying the order of business functions, which determines the order of business transactions. See the code snippet in Listing 38.2.

In this section, we have discussed the issues that need to be considered, suggested a list of properties that are important for modelling business transactions in service networks, and presented the core XML syntax of BTL. We provide a detailed description of transaction behaviour of business transactions in Sect. 3 using a scenario.

```
<BTL>
      <BusinessPerpsective>
            <KPI indicators = ""/>
            <Agreement>
                  <participant name =" Participant Name
                  " type ="xs:string" minOccurs ="2"
                  maxOccurs="n"/>
                  </participants>
                  <Policy type = type = "xs:string">
                  <expression>.....</expression>
                  <Policy>
                  <QoS>
                        <QoSAttribute name ="" type =
                        "xs:string"/>
                        <QoSMetric name =""/>
                         <QoSTargetValue value = ""
                         valueType = "xs:string"/>
                  </QoS>
            </Agreement>
      </BusinessPerpsective>
      <FunctionalPerpsective>
            <BusinessFunction>
                  <event name = "" type = "xs:string" />
                   <activity name = "" type = "xs:string"
                   vital = "xs:boolean"/>
            </BusinessFunction>
      </FunctionalPerpsective>
      <ProtocolPerspective>
            <BusinessProtocol>
                  <sequence>
                  ................................
                   </sequence>
            </BusinessProtocol >
      </ ProtocolPerspective >
</BTL>
```

Listing 38.2 The code snippet shows the elements that are used to model business transactions in SNs

2.1 Social Network Analysis and BTL

Since we have established that a service comprises complex exchanges of resources and competencies, it is important to understand how service interactions deliver these. SNA presents the technique to investigate network on analysis (e.g. structure of service relationships) and explore other network characteristics (Carroll et al. 2010).

Table 38.2 General principles of a network (Lewis 2009)

Characteristic	Description
Structure	Not a random collection of nodes and links and have a distinct format or topology which suggests that function follows form
Emergence	Network properties are emergent as a consequence of a dynamic network achieving stability
Dynamism	Dynamic behaviour is often the result of emergence or a series of small evolutionary steps leading to a fixed-point final state of the system
Autonomy	A network forms by the autonomous and spontaneous action of interdependent nodes that 'volunteer' to come together (link) rather than central control or central planning
Bottom-up evolution	Networks grow for the bottom or local level up to the top or global level. They are not designed and implemented from the top down
Topology	The architecture or topology of a network is a property that emerges over time as a consequence of distributed and often subtle forces or autonomous behaviours of its nodes
Power	The power of a node is proportional to its degree (number of link connecting to the network), influence (link values), and betweenness or closeness; the power of a network is proportional to the number and strengths of its nodes and links
Stability	A dynamic network is stable if the rate of change in the state of its nodes/links or its topology either diminishes as time passes or is bounded by dampened oscillations within finite limits

The term 'relational structure' may be used to describe a 'bundle of intuitive natural language ideas and concepts about the patterning' (Freeman et al. 1992; p. 12) in service relations among organisations to deliver a service. For example, Freeman et al. (1992; p. 12) describe that a 'social network' is 'a collection of more or less precise analytic and methodologically concepts and procedures that facilitate the collection of data in a systematic study of such patterning'. Thus, we can develop a greater understanding of a service network by describing its structure (nodes and links) and its behaviour (what the network does as a result of interactions among the nodes and links). This also supports the execution of a business transaction. This has an important application to the management, engineering, and design of service networks, that is, service science.

Network analysis emphasises the relations which connect the node positions within a system and 'offers a powerful brush for painting a systematic picture' of a service infrastructure and their interaction (Knoke and Kuklinski 1991; p. 173). Simply put, Salancik (1995) explains that 'a network theory about organisations should also be able to say how network properties themselves generate the properties of organisations' (p. 349). The major characteristics of attribute analysis are that the unit of analysis is the individual actor and the variable describes the behaviour of the network actor. We can apply Lewis (2009; pp. 20–21) list of the key characteristics of network science of the modern service era (Table 38.2):

Applying these principles to a service network, SNA may be simply described as an x-ray of the organisational service structure which highlights the importance of relational structures to support service performance (Carroll et al. 2010). According to Tichy et al. (1979), network analysis is concerned with the structure and pattern of these relationships and seeks to identify both their causes and consequences (p.507). This may be mapped onto the business transaction to allow the transactions 'tell the story' of service interactions. Tichy et al. (1979) provide an overview of network concepts and network properties as listed in Table 38.2 which are considered fundamental to the business transaction and service network performance. As identified in Table 38.3, we can borrow network analysis concepts to extend BTL development. Within a service context, these modifications are often initiated by a customer and influenced by business transaction properties, for example, an SLA which impacts the relational content. SNA provides us with the opportunity to visualise the dynamic infrastructure of the transactional network, and SNA properties (Table 38.3) can provide greater semantics to BTL and the nature of the service infrastructure. Therefore, the orchestration of structural relations (emergent property of the connection, the exchange process) or attributes (intrinsic characteristics, e.g. value of an exchange) becomes a central concept to analyse a network structural properties.

To describe our approach to date, we discuss a simple scenario which examines how service transactions consist of complex relations which allow us to apply BTL to understand how business transaction applies to a service model. In addition, we demonstrate how SNA provides a technique to model transactional relational flows to further extract service metrics within the network.

3 Scenario: Business Transactional Analysis

Service networks are not independent, isolated entities, but act in a concerted manner to survive in an ever-increasing dynamic business environment. Therefore, interacting organisations build networks to serve their customers in a dynamic manner (S-Cube 2009). In this section, we analyse business transactions using a supply chain scenario of Auto Inc. that is located in Southeast Asia (S-Cube 2008). Auto Inc. is a local branch of a large enterprise in the automobile industry in Europe. The Auto Inc. supply network comprises a regional headquarters in Singapore, a manufacturing factory in Vietnam, local and international suppliers, distribution centre, and several warehouses located in different countries in Southeast Asia. This demonstrates the distributiveness of the Auto Inc. network which has been spanned across wide territories. We develop a service network scenario including bank and insurance as service partners with the existing business partners. Figure 38.1 depicts the service network scenario of Auto Inc.

Auto Inc. supply chain network embodies suppliers from different regions and continents in particular, from Europe. Auto Inc. does not have logistics partner because the suppliers provide logistics support. In addition, the distribution centre is responsible for the transportation of tangible service (product) to the retailers. This setting manifests that the inter-organisational service network structure is

Table 38.3 Organisational network analysis concepts and network properties

Property	Explanation
Transactional content	Four types of exchanges: 1. Expression of effect (e.g. initiate a transaction) 2. Influence attempt (e.g. negotiating a SLA) 3. Exchange of information (e.g. terms and conditions) 4. Exchange of goods and services (e.g. payment)
Nature of links	
1. Intensity	The strength of the relations between individuals (i.e. intensity of service interactions)
2. Reciprocity	The degree to which a relation is commonly perceived and agreed on by all parties to the relation (i.e. the degree of symmetry)
3. Clarity of expression	The degree to which every pair of individuals has clearly defined expectations about each other's behaviour in the relation, i.e., they agree about appropriate behaviour between one another (i.e. SLA)
4. Multiplexity	The degree to which pairs of individuals are linked by multiple relations. Multiple roles of each member (e.g. consumer, supplier, negotiator) and identifies how individuals are linked by multiple roles (the more roles, the stronger the link)
Structural	
1. Size	The number of individuals participating in the network (i.e. service ecosystem)
2. Density (correctedness)	The number of actual links in the network as a ratio of the number of possible links (can determine service opportunity)
3. Clustering	The number of dense regions in the network (i.e. network positioning, structural holes which may identify service threats)
4. Openness	The number of actual external links of a social unit as a ratio of the number of possible external links (e.g. level of outsourcing)
5. Stability	The degree to which a network pattern changes over time (i.e. level of innovation and service evolution)
6. Reachability	The average number of links between any two individuals in the network (e.g. service brokerage)
7. Centrality	The degree to which relations are guided by the formal hierarchy (may determine power within a service network)
8. Star	The service with the highest number of nominations (can influence the level of success a service enjoys)
9. Liaison	A service which is not a member of a cluster but links two or more clusters
10. Bridge	A service which is a member of multiple clusters in the network (linking competencies and industry-specific services)
11. Gatekeeper	A star who also links the social unit with external domains (i.e. knowledge diffusion and service network analyst)
12. Isolate	A service which has uncoupled from the network (service termination)

highly complex since it connects many disparate and spatial service nodes that fire up numerous interactions among them.

The SBA that supports the end-to-end processes of the Auto Inc. service network comprises order processing, delivery, and payment services. These are composite services encapsulating operations that perform the requests triggered by the

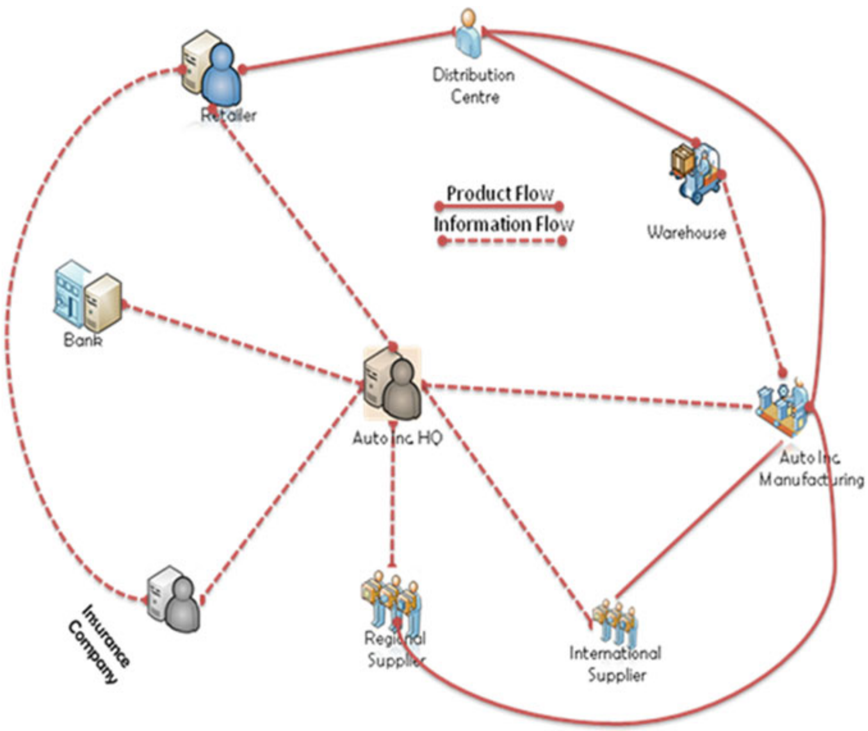


Fig. 38.1 Auto Inc. service network scenario comprises of partners entailing buyers, sellers, banks, insurance company, warehouse, distribution centre, and suppliers

partners across the network. The transactional properties are blended with these services, for which they are treated as transactional services. For this scenario, we consider that BTL has been used to define the transactional properties of these services at design time. These properties determine the behaviour of business transactions in Auto Inc. service networks. We discuss the three most important properties including business behavioural consistency, flex-atomicity, and reflexivity (we defined in Sect. 2) in this section. We discuss these properties to analyse the behaviours of transactions at runtime.

Business transactions in service networks (Fig. 38.1) are said to be consistent if and only if the business transactions satisfy the obligations and guarantees such as response time, processing lead time, delivery lead time, and data privacy. We name this consistency as *business behavioural consistency*. The obligations and guarantees that determine business behavioural consistency are stipulated in SLA, an element of the business perspective of BTL. A transaction monitor is employed to monitor SLA at runtime. The monitor flags the transactions as inconsistent if they fail to satisfy SLA during as well as after the execution. The business behavioural inconsistency may result failure of a transaction. For instance, if a delivery from international supplier is delayed, it is the behavioural inconsistency of the supplier.

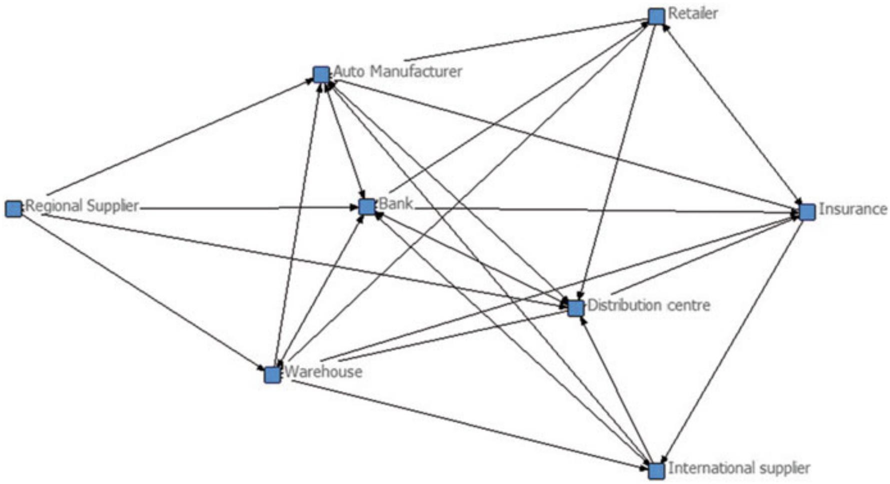


Fig. 38.2 Auto Inc. through SNA

It affects the transaction in several ways such as Auto Inc. may cancel the order which fails the transaction between Auto Inc. and the supplier. The effect of this failure may propagate throughout the network if Auto Inc. fails to deliver the cars ordered by retailers. The retailer cancels the order, which fails the transaction entirely for this particular instance (purchase order).

Furthermore, if a regional supplier fails to deliver the raw materials (e.g. CD player), the transaction between regional supplier and Auto Inc. fails. According to the notion of classical atomicity, the transactions that are linked to this failed transaction, for instance, the transaction between retailer and Auto Inc., should be aborted immediately. This atomicity has been relaxed in business transaction in Auto Inc. service network by invoking the compensation activity as well as contingent activity that are supplied by BTL (note that the transaction manager integrates BTL). We call it flex-atomicity. The compensation activity undoes the results of the transaction between regional supplier and Auto Inc., and contingent activity initiates an alternative transaction which repairs the failed transaction. This resists the failure of business transaction entirely.

Business in the given scenario (Fig. 38.2) is collaborative. Thus, it is important that any change in end-to-end process in Auto Inc. service network should be externalised immediately to the other partners that are part of the collaboration. This is *relaxed-isolation* property of business transaction. Note that *relaxed-isolation* allows exceptions. In particular, the failure of a transaction may not always be broadcasted across the partners in the service networks to resist the total failure of transactions. We suggest masking these failures using some techniques. The current version of BTL does not facilitate defining *relaxed-isolation* property. This work is in progress which we will add in the next version of BTL.

From the above discussion, it is clear that transactional properties play the pivotal role to controlling the runtime service behaviours. Thus, we analyse the behaviour (level of interaction) of transactions in the Auto Inc. service network through its interactions as we employ SNA. As Fig. 38.2 illustrates, SNA allows us to visualise the service network through the level of interaction (for more information on the application of SNA to service networks, refer to Carroll et al. 2010). Figure 38.1 illustrates the service network from a process perspective while Fig. 38.2 highlights the importance of actors such as the bank to support the service operations. The bank is a key actor within the service network since it finances many of the service operations and consequently is positioned within the centre of the service network SNA map. This is true from a business perspective and a customer perspective, as it relies on finance to, for example, avail of car loans. In this simple example, there are 32 ties within the network and a density (proportion of ties relative to the number of possible ties) of 0.5714 which suggests that there is opportunity to improve the service network cohesion. In addition, there are several key metrics which were outlined in Table 38.3 which support the development of service analysis through BTL.

For each pair of nodes within the service network, SNA calculates the number of edges in the shortest path between them. The average distance (among reachable pairs) is 1.347 while the distance-based cohesion (‘compactness’) within the network is 0.723. In general, the range is 0–1 where larger values indicate greater cohesiveness. The distance-weighted fragmentation (‘breadth’) of the network is 0.277. As suggested above, the density of the network allows us to identify the number of actual links in the network as a ratio of the number of possible links. This can also suggest possible opportunities. These density measures, for example, are summarised in Table 38.4. These measures assist us to gain insight as to the structure of the network upon which BTL is deployed to model service behaviour as discussed above.

4 Conclusion

This chapter is the result of the first phase of our research work on business transaction language and service network modelling analysis. BTL as described in this chapter is a specification containing expected details of transaction and as part of our future work; we are currently working on mapping the transaction properties (defined using BTL) to runtime language which will realise these properties. We believe that there is a list of works to be carried to successfully complete the BTL. In this chapter, we emphasise the need for BTL to enhance performance reporting mechanisms. From our perspective, the requirement of a language must be well-justified before the development of the language; otherwise, it can be obsolete like many other languages. In addition, we identify the suitability

Table 38.4 Service supply chain density measures

Density measures	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	Size	Ties	Pairs	Densit	AvgDis	Diamet	nWeakC	pWeakC	2StepR	ReachE	Broker	nBroke	EgoBet	nEgoBe
1. Regional supplier	4.00	10.00	12.00	83.33	1.17	2.00	1.00	25.00	100.00	24.14	1.00	0.17	0.00	0.00
2. International supplier	5.00	15.00	20.00	75.00	1.25	2.00	1.00	20.00	100.00	20.00	2.50	0.25	0.67	3.33
3. Auto manufacturer	7.00	23.00	42.00	54.76			1.00	14.29	100.00	17.07	9.50	0.45	0.58	1.39
4. Warehouse	7.00	24.00	42.00	57.14			1.00	14.29	100.00	16.67	9.00	0.43	1.17	2.78
5. Distribution centre	7.00	23.00	42.00	54.76			1.00	14.29	100.00	16.67	9.50	0.45	2.50	5.95
6. Retailer	5.00	15.00	20.00	75.00	1.25	2.00	1.00	20.00	100.00	20.00	2.50	0.25	0.33	1.67
7. Bank	7.00	19.00	42.00	45.24			1.00	14.29	100.00	16.67	11.50	0.55	10.75	25.60
8. Insurance	6.00	20.00	30.00	66.67	1.33	2.00	1.00	16.67	100.00	17.95	5.00	0.33	1.17	3.89

1. Size. Size of ego network
2. Ties. Number of directed ties
3. Pairs. Number of ordered pairs
4. Density. Ties divided by Pairs
5. AvgDist. Average geodesic distance
6. Diameter. Longest distance in egonet
7. nWeakComp. Number of weak components
8. pWeakComp. NWeakComp divided by size
9. 2StepReach. # of nodes within 2 links of ego
10. ReachEffic. 2StepReach divided Size
11. Broker. # of pairs not directly connected
12. Normalized Broker. Broker divided by number of pairs
13. Ego Betweenness. Betweenness of ego in own network
14. Normalized Ego Betweenness. Betweenness of ego in own network

of SNA to support our quest to examine behavioural and structural aspects of service networks and its influence on service performance. As part of our future work, we are continuing to develop BTL and apply this within a real-world case study. In addition, we plan to merge SNA analysis metrics to enhance service network modelling analysis techniques which are sought in service management and service computing fields.

References

Carroll N, Whelan E, Richardson I (2010) Applying social network analysis to discover service innovation within agile service networks. Serv Sci 2(4):225–244

Chesbrough H, Spohrer J (2006) A research manifesto for services science. Commun ACM 49(7):35–40

Chrysanthis KP, Ramamritham K (1990) ACTA: a framework for specifying and reasoning about transaction structure and behavior. In: Proceedings of the ACM SIGMOD symposium on the management of data. ACM, New York

Freeman LC, White DR, Romney AK (1992) Research methods in social network analysis. Transaction Publishers, New Brunswick

Garcia-Molina HA, Salem K (1987) Sagas. In: Proceedings of ACM SIGMOD international conference on management of data. ACM, New York

Gray J, Reuter A (1993) Transaction processing: concepts and techniques. s.l. Morgan Kaufmann, San Francisco

Harder T, Reuter A (1983) Principle of transaction oriented database recovery. ACM Comput Surv 15(4):287–317

Knoke D, Kuklinski JH (1991) Network analysis: basis concepts, chapter 14. In: Thompson G, Frances J, Levačić R, Mitchell J (eds) Markets, hierarchies and networks: the coordination of social life. Sage Publications, London

Lewis TG (2009) Network science—theory and application. Wiley, Hoboken

Leymann F, Roller D (2000) Production workflow: concepts and techniques. Prentice Hall PTR, Upper Saddle River

Normann R (2001) Reframing business: when the map changes the landscape. Wiley, Chichester

Papazoglou MP (2003) Web services and business transactions. World Wide Web: Internet Web Inf Syst 6(1):49–91

Salancik GR (1995) Wanted: a good network theory of organization. Adm Sci Q 40:345–349

S-Cube (2008) S-Cube project website. http://www.s-cube-network.eu/

S-Cube (2009) Initial models and mechanisms for quantitative analysis of correlations between KPIs, SLAs and underlying business processes. Available Online: http://www.s-cube-network.eu/results/deliverables/wp-jra-2.1

S-Cube (2010) Business transaction language. Available Online: http://www.s-cube-network.eu/results/deliverables/wp-jra-2.1

Spohrer J, Maglio PP (2009) Service science: toward a smarter planet. In: Karwowski W, Salvendy G (eds) Service engineering. Wiley, New York

Spohrer J, Maglio PP, Bailey J, Gruhl D (2007) Steps toward a science of service systems. IEEE Comput 40(1):71–77

Tichy NM, Tushman ML, Frombrun C (1979) Social network analysis for organizations. Acad Manage Rev 4:507–519

Traiger LI (1983) Trends in system aspects of database Management. In: Proceeding of the 2nd international conference on databases (UCOD-2)

Chapter 39
Quantitatively Evaluating the Effects of Price Promotions Using Data Mining

Min Gan and Honghua Dai

1 Introduction

Price promotions (also called discount promotions), i.e. short-term temporary price reductions for selected items (Hermann 1989), are frequently used in sales promotions. The main objective of price promotions is to boost sales and increase profits. Quantitative evaluation of the effects of price promotions (QEEPP) is essential and important for sales managers to analyse historical price promotions and informative for devising more effective promotional strategies in the future. However, most previous studies only provide insights into the effects of discount promotions from some specific prospectives, and no approaches have been proposed for comprehensive evaluation of the effects of discount promotions. For example, Hinkle [1965] discovered that price promotions in the off-season are more favourable, and the effects of price promotions are stronger for new products. Peckham [1973] found that price promotions have no impact on long-term trend. Blattberg et al. [1978] identified that different segments respond to price promotions in different ways. Rockney [1991] discovered three basic types of effects: effects on discounted items, effects on substitutes and effects on complementary items.

Since the POS (point of sale) system has been used in retail, a large number of transactions have been collected and inputted into databases. Association rules introduced by Agrawal et al. [1993] and Anand et al. [1998] are well-suited for modelling cross-selling relationships embedded in market basket data (Anand et al. 1997, Hruschka et al. 1989). For instance, association rule *margarine* $\rightarrow$ *sugar* $[conf = 0.18]$ indicates that among the customers who purchase margarine, 18 % of people also purchase sugar in the same shopping trip. Therefore, association rule mining has been widely applied to market basket analysis (Agrawal

M. Gan (✉) • H. Dai
Deakin University, 221 Burwood Highway, Burwood, VIC 3125, Australia
e-mail: mg@deakin.edu.au; honghua.dai@deakin.edu.au

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_39,

et al. 1993), cross-selling strategies (Anand et al. 1998), product assortment (Brijs et al. 1999) and maximum-profit product selection (Brijs et al. 2000, Wang and Su 2002, Wong et al. 2003) and has also been widely used in other areas such as analysis of academic results (Robertas 2008). However, to the best of our knowledge, QEEPP has never been addressed in the literature in association with data mining.

Naively, the difference between sales/profit level of a discounted item in a promotion period and its sales/profit level before the promotion time seems to reflect the effect of a promotion. However, this difference may not comprehensively and precisely reflect the promotional effect when (1) the promotion not only affects the sales of the discounted item itself but also influences the sales of other related items, or (2) the sales of the item is not only affected by its promotion but also influenced by the promotions on other items. This chapter aims to propose a framework for quantitatively evaluating the effects of price promotions with considerations of three types of promotional effects. To achieve a comprehensive and precise quantitative evaluation, we first represent three types of promotional effects by appropriate patterns such as association rules. We then discover the promotional effects based on association rule mining, and finally, we quantify the effects on sales and profits by analysing the changes in sales and profits of each related item. The final results are displayed as weighted impact graphs.

The rest of this chapter is organised as follows. Section 39.2 addresses motivating scenarios. Section 39.3 proposes the framework, and a simulation is presented in Sect. 39.4. Section 39.5 concludes this chapter.

2 Motivating Scenarios

In this section, we explore scenarios of discount promotions and sales in a retail store. In these scenarios, we address (1) what original data is available and needed, (2) why naive analysis of changes in sales and profits is not enough, (3) how discount promotions can affect the sales and (4) what should be evaluated in our framework.

2.1 Data and Assumptions

In a retail store, assume itemset I contains all items on sale. The information of every transaction is collected by a POS machine and stored in a transaction database DB. As shown in Table 39.1, each row in DB represents the information of one transaction, which includes TID (transaction ID), time (transaction time), item-information (item, unit price, quantity, subtotal) and total.

We assume in this store that sales managers select a certain number of items to discount at certain discount rates every period of time (e.g. 2 weeks). The discount

Table 39.1 Transaction database *DB*

TID	Time	Item-information (item, unit price, quantity, subtotal)	Total ($)
10000001	13:28, 12/03/2009	Apple, 1.62, 2.4 kg, $3.96 Sugar, 2.3, 1.5 kg, $3.45	7.41
10000002	...		
.			
.			
.			
19999999	...		

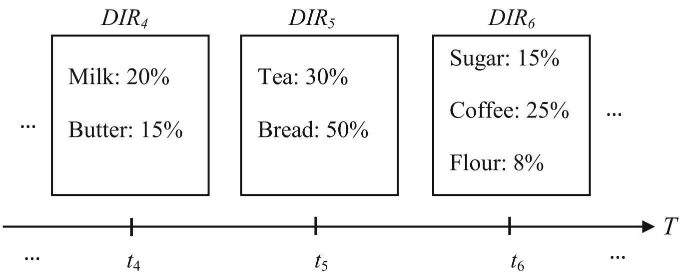


Fig. 39.1 A fragment of discount records

records are ordered in a sequence as shown in Fig. 39.1. At time t_4, milk and butter are selected to be discounted by 20 and 15 %, respectively, until time t_5. The discount promotions during $[t_j, t_{j+1})$ are represented by a set of items with discount rates, denoted as DIR_j, e.g. in Fig. 39.1, $DIR_4 = \{(Milk, 20\ \%), (Butter, 15\ \%)\}$. We assume that different items are selected to discount every time, i.e. $DI_j \cap DI_{j+1} = \emptyset$. We divide discount rates into six levels: 1 (1–10 %), 2 (11–20 %), 3 (21–30 %), 4 (31–40 %), 5 (41–50 %) and 6 ($\geq$ 51 %). We use DIL_j to represent the set of items with discount levels in $[t_j, t_{j+1})$, e.g. $DIL_5 = \{(Tea, 3), (Bread, 5)\}$.

Example 1 Assume that *DB* in Table 39.1 is a transaction database of a retail store and Fig. 39.1 shows the discount promotions performed in this store in a past time period.

2.2 Possible Impacts

It has been revealed in the marketing community that discount promotions have three types of promotional effects: positive effects on the sales of discounted items, positive effects on the sales of complements and negative effects on the sales of substitutes (Rockney 1991). In Example 1, the effects can be represented by impact graphs in Fig. 39.2. Each arrow indicates a promotional effect. For example, in

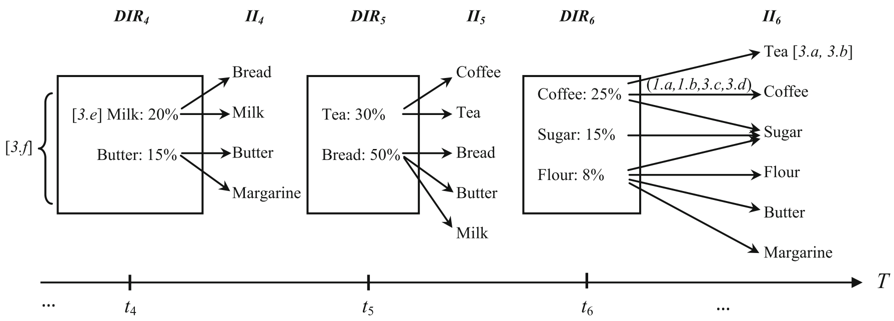


Fig. 39.2 (Weighted) impact graphs

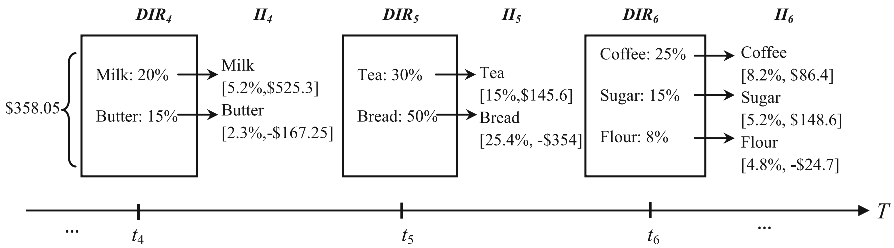


Fig. 39.3 A naive measurement

$[t_4, t_5)$, the discount on milk, denoted as $Dis(milk)$, has a positive effect on the sales of milk itself (Type 1), and has a positive effect on the sales of its complement, bread (Type 2). The discount on butter has a negative effects on the sales of its substitute, margarine (Type 3). The set of influenced items (IIs) in $[t_4, t_5)$ is denoted as II_4. In Example 1, $II_4 = \{bread, milk, butter, margarine\}$.

2.3 *Naive Measurement Versus Quantitative Evaluation*

A naive way to measure the promotional effects is to examine the changes in the sales/profits of discounted items. Figure 39.3 shows the changes due to the price promotions in three time periods. In Fig. 39.3, milk [5.2 %, $\$525.\ 3$] means that the discount promotion on milk in $[t_4, t_5)$ increases its average daily sales volume in $[t_4, t_5)$, denoted as $dV\ (milk,\ [t_4,\ t_5))$, by 5.2 % and obtains average daily profit gain (G_{dP}) of \$525.3. Thus, G_{dP} from DIR_4 is $\$525.\ 3 - \$167.\ 25 = \$358.\ 05$.

An essential deficiency of the naive measurement is the ignorance of two other types of promotional effects (on complements and substitutes). Consequently, it cannot comprehensively and precisely evaluate the promotional effects. To begin with, the measurement is incomplete without considerations of the other two types of effects. Secondly, the effects could be underestimated or overestimated.

For example, the effect of *Dis*(*milk*) in [t_4, t_5) may be underestimated since the positive effect of *Dis*(*milk*) on the sales of bread is ignored. This means the profit contribution of *Dis*(*milk*) may be greater than $525.3. Similarly, the effect of *Dis* (*butter*) may be overestimated as the negative effect of *Dis*(*butter*) on the sales of margarine is ignored. Thirdly, the naive way fails to reveal how the sales of an item is influenced. For example, Fig. 39.3 indicates that sugar gains G_{dP} $148.6 from its discount sales. Actually, as indicated in Fig. 39.2, three items increase the sales of sugar. Fourthly, the difference in the sales/profit level of an item between two adjacent time periods does not necessarily reflect the change due to the particular promotions. For example, the increase in *dV* of coffee 8.2 % = [*dV* (*coffee*, [t_6, t_7)) − *dV* (*coffee*, [t_5, t_6))] / *dV* (*coffee*, [t_5, t_6)) may not be the real increase due to the promotion of coffee, as the sales level of coffee in [t_5, t_6) has been influenced by the promotion on tea in [t_5, t_6). This indicates that the real increase may be greater than 8.2 %.

To achieve a comprehensive and precise evaluation, we quantitatively evaluate the effects by taking into account the three types of promotional effects. In this evaluation, we need to tackle three key issues: representation, discovery and evaluation of promotional effects. The promotional relationships are discovered and seven metrics are evaluated in our framework. As shown in Fig. 39.2, the metrics are represented as notations of the format class.subclass like 1.a, 1.b etc. The metrics are listed as follows:

1. Representation
   - 1.a —Effect types (1, 2, 3)
   - 1.b —The strength of each effect
2. Discovery—Discovery of promotional effects (arrows in Fig. 39.2).
3. Evaluation
   - 3.a —G_{dV}(y, [t_j, t_{j+1}): The growth of *dV* of item y due to the price promotions in [t_j, t_{j+1}).
   - 3.b —G_{dP}(y, [t_j, t_{j+1}): The growth of *dP* of item *y* due to the price promotions in [t_j, t_{j+1})
   - 3.c —$RG_{dV}(x \xrightarrow{P} y, t_j)$: The growth of *dV* of item *y* due to the effect of *Dis*(x)
   - 3.d —$RG_{dP}(x \xrightarrow{P} y, t_j)$: The growth of *dP* of *y* due to the effect of *Dis*(x)
   - 3.e —G_{dP}(·, *Dis*(x), [t_j, t_{j+1})): The overall growth of *dP* gained from the promotion on x in [t_j, t_{j+1})
   - 3.f —G_{dP}(·, *Dis*(DIR_j), [t_j, t_{j+1})): The overall growth of *dP* gained from the promotions on DIR_j in [t_j, t_{j+1})

3 The Framework

3.1 Representation of Promotional Effects

Firstly, association rules (Agrawal et al. 1993) are briefly reviewed. Given itemset I, an association rule is an implication of the form $X \rightarrow Y$, where $X, Y \subset I$ and $X \cap Y = \emptyset$. The support of itemset X in the transaction database DB, denoted as $sup(X)$, is defined as the number of transactions in DB that contain X. The support of association rule $X \rightarrow Y$, $sup(X \rightarrow Y)$ is defined as $sup(XY)$. The confidence of $X \rightarrow Y$, $conf(X \rightarrow Y)$ is defined as $sup(XY)\,/\,sup(X)$. Given minimum thresholds *min_sup* and *min_conf*, association rule mining is to discover association rules with support and confidence no less than *min_sup* and *min_conf*. We use $AR(DB$, *min_sup*, *min_conf*) to denote the set of association rules in DB that satisfy the thresholds of *min_sup* and *min_conf*. We use $Dis(x)$ to represent a discount promotion on item x.

The three types of effects (1.a) can be formally represented as follows.

Definition 1 (Representation of Promotional Effects)

1. Effect on itself—$Dis(x)$ has a positive effect on the sales of itself, denoted as $x \rightharpoonup x$.
2. Effect on complements—$Dis(x)$ has a positive effect on the sales of its complement y ($y \neq x$), denoted as $x \rightharpoonup y$, if $x \rightarrow y \in AR(DB, min\_sup, min\_conf)$.
3. Effect on substitutes—$Dis(x)$ has a negative effect on the sales of a substitute y of x ($y \in Subs(x)$), denoted as $x \rightharpoondown y$.

As defined above, each type of promotional effect is represented as a rule, which is called a discount promotional rule (DPR) in this chapter. We use $x \xrightarrow{P} y$ to denote a general DPR of any type.

For each DPR, the strength of its promotional effect (1.b) is defined as follows. We set the strength of $x \rightharpoonup x$ as 1 because normally a discount promotion on a product has a definite effect on the sales of itself. The strength of $x \rightharpoondown y$ is defined as the minus value of the market share of y among the whole market of x and the substitutes of x. For example, assume that a has two substitutes b and c, and V(DB, a):V(DB, b):V(DB, c) = 2:1:2 (V(DB, a) denotes sales volume of a in DB), then the strength of $a \rightharpoondown b$ is − 1/5. The strength of complementary effects can be measured by metrics based on the support. Promotional strengths (PSs) of three types of DPRs are formally defined as follows.

Definition 2 (Promotional Strength) The promotional strengths of three types of promotional rules in Definition 1 are respectively defined as:

1. $PS(x \rightharpoonup x) = 1$
2. $PS(x \rightharpoonup y) = \frac{sup(xy)}{sup(x)} - \frac{sup(y)}{|DB|}$
3. $PS(x \rightharpoondown y) = -\frac{V(DB,y)}{\sum_{z \in \{x\} \cup Subs(x)} V(DB,z)}$

Given a minimal threshold of promotional strength *min_PS*, the DPRs with absolute values of strengths no less than *min_PS* are called strong DPRs. In examining promotional effects, we only consider strong DPRs. Given *DB*, *min_sup*, *min_conf* and *min_PS*, the set of strong DPRs is denoted as *SDPR*(*DB*), where the other parameters are omitted. The rules in *SDPR*(*DB*) represent promotional relationships among different items in DB.

3.2 Discovery of Promotional Effects

According to Definition 1, the first type SDPR '$x \rightharpoonup x'$ can be obtained immediately. The second type of SDPRs can be discovered via association rule mining. For the third type '$x \rightharpoonup y$'' we assume that Subs(x) is provided by the sales manager.

There have been many approaches available for mining association rules, such as Apriori (Agrawal and Srikant 1994) and FP-growth (Han et al. 2000). In this chapter, we choose one of the most efficient approaches, FP-growth (Han et al. 2000) to discover association rules. In the process of finding association rules, PS is calculated and SDPRs are generated. If the local confidence of some rules formed by items of particular discount levels deviates substantially from the global confidence of the same rules without discounts information, we treat them as different rules (Chen et al. 2008).

3.3 Evaluation of Promotional Effects

The first key issue in the evaluation is to detect $G_{dV}(y, [t_j, t_{j+1}))$ (the real growth of dV of an influenced item y in $[t_j, t_{j+1})$). As discussed in Sect. 39.2.3, the naive difference does not necessarily reflect the real growth. Therefore, we define $G_{dV}(y, [t_j, t_{j+1}))$ (3.a) as the difference between $dV(y, [t_j, t_{j+1}))$ (average daily sales volume of y in $[t_j, t_{j+1})$) and the latest normal average daily sales volume (LNdV) of y with respect to t_j.

Definition 3 (The Latest Normal Sales Time Period, LNdV) For any $y \in II_j$, an interval $[t_{h-1}, t_h)$ $(1 < h \leq j)$ is called a normal sales time period (NSTP) of y if the sales of y is not influenced by any discount promotion during $[t_{h-1}, t_h)$, i.e. $y \notin II_{h-1}$. Furthermore, $[t_{h-1}, t_h)$ is called the latest normal sales time period (LNSTP) of y with respect to t_j, denoted as $LNP(y, t_j)$, if there exists no h' $(h < h' \leq j)$ such that $[t_{h'-1}, t_{h'})$ is an NSTP of y. The latest normal sales volume of y with respect to t_j is defined as the average daily sales volume of y during the LNSTP of y with respect to t_j, i.e. $dV(y, LNP(y, t_j))$.

For instance, in Fig. 39.2, $[t_4, t_5)$ and $[t_5, t_6)$ are two NSTPs of flour and $LNP(flour, t_6) = [t_5, t_6)$. $[t_5, t_6)$ is not a NSTP of coffee, and $LNP(coffee, t_6) = [t_4, t_5)$.
Thus, $G_{dV}(y, [t_j, t_{j+1}))$ can be defined as $dV(y, [t_j, t_{j+1})) - dV(y, LNP(y, t_j))$.
$G_{dP}(y, [t_j, t_{j+1}))$ (3.b) gained from $G_{dV}(y, [t_j, t_{j+1}))$ can be defined as

$$\begin{cases} \widehat{UP}(y) \times dV(y,[t_j,t_{j+1})) - UP(y) \times dV(y,LNP(y,t_j)) & \text{if } y \in DI_j \\ UP(y) \times G_{dV}(y,[t_j,t_{j+1})) & \text{otherwise} \end{cases} \tag{39.1}$$

where, $\widehat{UP}(y)$ is the unit profit of y when it is sold at a discount price in $[t_j, t_{j+1})$, and $UP(y)$ is the unit profit of y when it is sold at the normal price.

Let $SR(DIL_j \xrightarrow{P} y)$ be the set of SDPRs whose left-hand side belongs to DIL_j and the right-hand side is y. Thus, $G_{dV}(y, [t_j, t_{j+1}))$ can be seen as the overall contribution of all the rules in $SR(DIL_j \xrightarrow{P} y)$. We now separately evaluate the contribution of any rule $r : x \xrightarrow{P} y$ to $G_{dV}(y, [t_j, t_{j+1}))$. The contribution of r depends on two factors: $G_{dV}(x, [t_j, t_{j+1}))$ and $PS(x \xrightarrow{P} y)$, and can be weighted by

$$w(x \xrightarrow{P} y) = G_{dV}(x,[t_j,t_{j+1})) \times PS(x \xrightarrow{P} y) \tag{39.2}$$

Then, the total weight of the overall contribution of the discount promotions on DIL_j to $G_{dV}(y, [t_j, t_{j+1}))$, denoted as $W(\cdot \xrightarrow{P} y)$, can be defined as

$$W(\cdot \xrightarrow{P} y) = \sum_{r \in SR(DIL_j \xrightarrow{P} y)} w(r) \tag{39.3}$$

The contribution of $x \xrightarrow{P} y$ to $G_{dV}(y, [t_j, t_{j+1}))$, $RG_{dV}(x \xrightarrow{P} y, [t_j, t_{j+1}))$ (3.c) is called rule based daily volume gain. It is defined as

$$RG_{dV}(x \xrightarrow{P} y,[t_j,t_{j+1})) = \frac{w(x \xrightarrow{P} y)}{W(\cdot \xrightarrow{P} y)} \times G_{dV}(y,[t_j,t_{j+1})) \tag{39.4}$$

For the convenience of comparison, we further quantify $RG_{dV}(x \xrightarrow{P} y, [t_j, t_{j+1}))$ as a percentage, which is defined as

$$\frac{RG_{dV}(x \xrightarrow{P} y,[t_j,t_{j+1}))}{dV(y,LNP(y,t_j))} \times 100\% \tag{39.5}$$

Similar to Eq. (39.1), $RG_{dP}(x \xrightarrow{P} y, [t_j, t_{j+1}))$ (3.d) can be defined as

$$\begin{cases} \widehat{UP}(y) \times dV_+(y, [t_j, t_{j+1})) - UP(y) \times dV(y, LNP(y, t_j)) & \text{if } y \in DI_j \\ UP(y) \times RG_{dV}(x \xrightarrow{P} y, [t_j, t_{j+1})) & \text{otherwise} \end{cases} \tag{39.6}$$

where, $dV_+(y, [t_j, t_{j+1})) = dV(y, [t_j, t_{j+1})) + RG_{dV}(x \xrightarrow{P} y, [t_j, t_{j+1}))$.

Based on (3.d), $G_{dP}(\cdot, Dis(x), [t_j, t_{j+1}))$ (3.e) can be defined as

$$G_{dP}(\cdot, Dis(x), [t_j, t_{j+1})) = \sum_{all\ y,\ s.t.\ x \xrightarrow{P} y \in SDPR(DB)} RG_{dP}(x \xrightarrow{P} y, [t_j, t_{j+1})) \tag{39.7}$$

Thus, 3.f can be defined as

$$G_{dP}(\cdot, Dis(DIR_j), [t_j, t_{j+1})) = \sum_{all\ x \in DIR_j,\ s.t.\ x \xrightarrow{P} y \in SDPR(DB)} G_{dP}(\cdot, Dis(x), [t_j, t_{j+1})) \tag{39.8}$$

4 Simulation

Since we have not been able to obtain real retail data with discount information, we generated simulated data to test our method.

4.1 Data Generation

In data generation, we simulate the transactions of a middle-sized retail store. The scenario assumes the store offers 5,000 different items.[1] Promotional relationships are embedded in historical transactions in the past 6 months, and managers select a percentage of items to discount every 2 weeks. Sales volumes are influenced by discount promotions according to the promotional relationships and promotional strengths evident in the historical transactions.

Synthetic data was generated as follows. To begin with, we used the IBM synthetic data generator (Agrawal and Srikant 1994) to generate an original transaction database, and set appropriate parameters to simulate the characteristics of the store. The number of items N is set as 5,000, the number of transactions $|D| = 200{,}000$, and the average size of the transactions $|T| = 4$. Secondly, we

[1] In this chapter, we do not consider the update of items such as the introduction of new items and deletion of some items.

adjusted the supports (frequencies) and qualities of items in some transactions in the original database so that average daily sales volume and local support of every item was kept almost the same every 2 weeks (intended for simulating stable levels of sales when no discount promotions are performed). We then discovered association rules and SDRP(DB) from the database.

Thirdly, we randomly generated the unit selling price and unit purchase price for each item. In the generation of prices, we used settings similar to that in Wang and Su [2002]. The prices were generated such that 13 % of items had low-unit profits between \$0.1 and \$1.0, 75 % of items had medium-unit profits between \$1.0 and \$10.0, 10 % of items had high-unit profits between \$10.0 and \$50.0 and 2 % of items had extremely high-unit profits between \$50.0 and \$500.0.

Fourthly, around 10 % items are selected for discount every 2 weeks. In the selection of items and the setting of discount rates, we tried to make the combinations as diverse as possible.

The final process involved changing local supports of items that were influenced by the discounted items according to the promotional relationships represented by the rules in DPRS(DB). The transaction database after the change processing was the simulation of the real transactions with the influence of discount promotions.

4.2 Results

We used the simulated data sets to test our method. The parameters in the experiments were set as $k = 100$, $min\_sup = 0.5\ \%$, $min\_conf = 0.1$ and $min\_PS = 0.2$. Figure 39.4 shows a fragment of weighted impact graphs obtained during a promotion period. In this graph, the structure depicts the promotion relationships among the items with the effects quantified by six values. The graph can be viewed from three different angles: discounted items, promotional effects and influenced items. In Fig. 39.4, if we look at discounted item i_{23}, we know that $Dis(i_{23})$ has influenced the sales of four items, and it has had an overall profit contribution of $\$76.31$. Viewing the effect of $Dis(i_{23})$ on the sales of i_{51}, one knows that $Dis(i_{23})$ has a negative influence on the sales of i_{51} with a strength of 0.31, a sales decrease of 0.36 % and a profit increase of \$6.53. Viewing this from an angle of the influenced item i_{148}, one knows that two discounted items (i_{148} and another) have influenced its sales, and the average daily sales volume of i_{148} has increased by 13.41 % with its average daily profit gain dP increasing by $\$283.75$. In the overall profit gain of $\$283.75$, the main part ($\$248.82$) is from the promotion on itself, and the other part ($\$34.93$) is from the promotion on another item.

In summary, weighted impact graphs obtained by our framework represent and evaluate the effects of price promotions comprehensively and quantitatively from different angles.

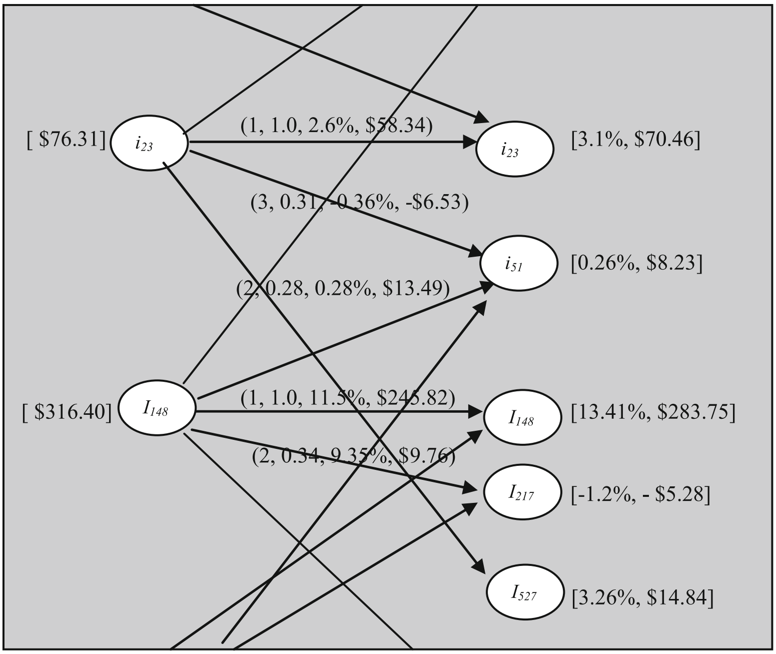


Fig. 39.4 A fragment of results

5 Conclusion and Future Work

In this chapter we investigated an important topic of price promotions: quantitative evaluation of the effects of price promotions on sales and profits. We proposed a framework for representing, discovering and evaluating the effects of price promotions based on data mining techniques such as association rule mining. The proposed framework is able to discover and quantitatively evaluate complex promotional effects hidden in historical transaction databases. The results are visualised as weighted impact graphs.

This work is significant for the marketing and data mining communities. The proposed framework enables sales managers to gain a deeper and detailed quantitative insight into the effects and helps them explore the inner mechanism of price promotions. Results obtained from the framework are not only useful for quantitatively examining the effects of historical price promotions but also informative for creating more effective discounting strategies.

In the data mining community, market basket analysis based on association rule mining (Agrawal et al. 1993) has been extensively studied, and a number of methods have been developed for real market basket analysis tasks, such as product assortment (Brijs et al. 1999) and maximum-profit product selection (Brijs et al. 2000, Wang and Su 2002, Wong et al. 2003). However, these methods ignore complex effects of price promotions. Therefore, when applying these methods to real market basket analysis tasks, bias may be introduced due to the ignorance of the effects of price promotions. Our framework could be embedded to improve existing methods.

It should be noted that the proposed framework is based on a few assumptions, however, some of these assumptions may not conform to reality. To begin with, we assume that promotions are conducted every 2 weeks. In a real store, items may be selected and discounted every day. Secondly, it is assumed that price promotions have no long-term influence on sales. Thirdly, it is assumed that items offered by a store are fixed without introduction of new items, or deletion of obsolete items. In consideration of these issues, we aim to improve this framework in our future research.

References

Agrawal R, Imielinski T, Swami AN (1993) Mining association rules between sets of items in large databases. In: Proceedings of SIGMOD international conference on management of data'93, Washington, DC, pp 207–216

Agrawal R, Srikant R (1994) Fast algorithms for mining association rules. In: Proceedings of the 20th international conference on very large data bases, Santiago, pp 487–499

Anand S, Hughes J, Bell D, Patrick A (1997) Tackling the cross-sales problem using data mining. In: Proceedings of the 1st Pacific-Asia conference on knowledge discovery and data mining, Singapore, pp 331–343

Anand S, Patrick A, Hughes JG, Bell DA (1998) A data mining methodology for cross-sales. Knowl Based Syst 10:449–461

Blattberg R, Buesing T, Peacock P, Sen S (1978) Identifying the deal prone segment. J Mark Res 15:369–377

Brijs T, Swinnen G, Vanhoof K, Wets G (1999) Using association rules for product assortment decisions: a case study. In: Proceedings of the 5th ACM SIGKDD international conference on knowledge discovery and data mining, San Diego, pp 254–260

Brijs T, Goethals B, Swinnen G, Vanhoof K, Wets G (2000) A data mining framework for optimal product selection in retail supermarket data: the generalized PROFSET model. In: Proceedings of the tth ACM SIGKDD international conference on knowledge discovery and data mining, Boston, pp 300–304

Chen Y, Huang TC, Chang S (2008) A novel approach for discovering retail knowledge with price information from transaction databases. Expert Syst Appl 34:2350–2359

Han J, Pei J, Yin Y (2000) Mining frequent patterns without candidate generation. In: Proceedings of SIGMOD international conference on management of data, Dallas, pp 1–12

Hinkle C (1965) The strategy of price deals. Harv Bus Rev 43:75–85

Hermann S (1989) Price management. Elsevier Science, Amsterdam

Hruschka H, Lukanowicz M, Buchta C (1989) Cross-category sales promotion effects. J. Retailing and co Elsevier Science Publishers B.V.

Peckham JO (1973) The wheel of marketing. Nielsen, Northbrook

Robertas D (2008) Analysis of academic results for informatics course improvement using association rule mining. In: Proceedings of ISD'08, Paphos, pp 357–363

Rockney GW (1991) Assessing the impact of retail price promotions on product substitution, complementary purchase, and interstore sales displacement. J Mark 55(2):17–28

Wang K, Su M (2002) Item selection by "Hub-Authority" profit ranking. In: Proceedings of the 8th ACM SIGKDD international conference on knowledge discovery and data mining, Edmonton, pp 652–657

Wong RC, Fu AW, Wang K (2003) MPIS: maximal-profit item selection with cross-selling considerations. In: Proceedings of the 3rd IEEE international conference on data mining, Melbourne, pp 371–378

Chapter 40
Incorporating Users into AmI System Design: From Requirements Toward Automation

Estefanía Serral, Luca Sabatucci, Chiara Leonardi, Pedro Valderas, Angelo Susi, Massimo Zancanaro, and Vicente Pelechano

1 Introduction

The term ambient intelligence (AmI) is still a vision of the future of consumer electronics in which the computational power is embedded in everyday appliances and physical objects to turn environments into sensitive places able to understand users' needs and to automate their daily tasks (Weiser 1995). In the context of AmI, task automation is central and raises many challenges since the system must adapt to each individual's specific needs. These challenges become even more critical when the domain is characterized by the presence of many actors, every one owning different institutional roles, responsibilities, skills, and motivations (Cook et al. 2003). In addition, since users' preferences may change in time, it is also important that the developed system provides evolution facilities for adapting to new requirements; otherwise, the system may become useless, obsolete, or perceived as intrusive by final users. It is therefore of paramount importance to use requirements engineering techniques for the analysis of users' needs and for involving users to participate in design and development choices (Rolland and Salinesi 2009, Van Lamsweerde 2003).

We present the results of a cooperative work for developing a user-intensive AmI system that involves a multidisciplinary team composed by, among others, software engineers, sociologists, and HCI designers. The challenges are summarized in (1) eliciting a set of requirements that encapsulate the real users'

E. Serral (✉) • P. Valderas • V. Pelechano
Centro de investigación en Métodos de Producción de Software,
Universidad Politécnica de Valencia, Valencia, Spain
e-mail: eserral@pros.upv.es; pvalderas@pros.upv.es; pele@pros.upv.es

L. Sabatucci • C. Leonardi • A. Susi • M. Zancanaro
Fondazione Bruno Kessler IRST, Trento, Italy
e-mail: sabatucci@fbk.eu; cleonardi@fbk.eu; susi@fbk.eu; zancana@fbk.eu

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_40,

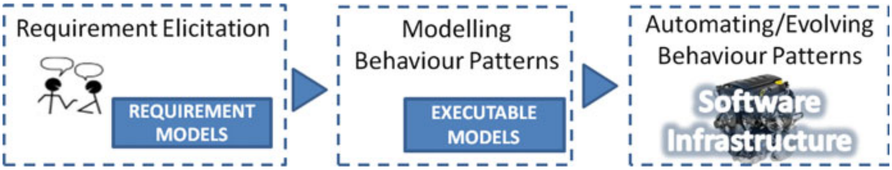


Fig. 40.1 Overview of the methodology

needs and (2) providing software infrastructures that guarantee the fulfillment of requirements and enable their customization and evolution at runtime.

The receipt we present in this chapter contains three main ingredients: a goal-oriented requirement engineering (GORE) process (Bresciani et al. 2004), a user-centered design process (Leonardi et al. 2010a) and a software infrastructure based on executable models (Serral et al. 2010a). Opportunely mixing these ingredients, we obtain:

- Iterative involvement of users within the design methodology, from early requirements identification to prototype evaluation
- Requirements engineering techniques able to integrate users' needs, preferences, and activities since the beginning of the design process
- Software infrastructures that guarantee the fulfillment of these requirements and also simplify their evolution over time without the need to redeploy the system

The resulting methodology (Fig. 40.1) consists of the following steps:

Requirement elicitation: This step stems from the consideration that modelling user requirements requires a deep understanding of the organizational and social context in which any system will operate (Nuseibeh and Easterbrook 2000). Section 40.3 provides details about the RE process, which is obtained by coupling two methods (Leonardi et al. 2010a,b): the User-Centered Design (UCD) (Cooper et al. 2007, Sharp et al. 2007) and Tropos (Bresciani et al. 2004) (a GORE process).

Identifying and modeling behavior patterns: A behavior pattern is a set of tasks that are habitually performed when similar contexts arise (Neal and Wood 2007). In this step, the behavior patterns that users want to be automated are identified and specified in models of a high level of abstraction, which are also prepared for being directly executed. Section 40.4 introduces the language to specify these executable models. Section 40.5 explains the process to obtain the executable models (behavior patterns) from the requirement elicitation models.

Automating and evolving user behavior patterns: We have developed a software infrastructure that directly uses the executable models at runtime to automate the behavior patterns as specified. Section 40.6 introduces the infrastructure and discusses automation and evolution of behavior patterns.

The feasibility of this methodology and the automation infrastructure has been evaluated through the development of an automated user-intensive AmI system within the ACube Project.[1] It aims at providing support to medical and assistance staff in elderly. This is also the running example used overall this chapter.

Finally, Sect. 40.7 concludes this chapter providing discussions about the benefits and drawbacks of the presented work.

2 Related Work

AmI is a young research area that has been mainly focused on the development of implementation technologies, but, recently, methodological approaches are emerging. So far, the integration of requirement elicitation techniques with software infrastructures that guarantee the fulfillment of the captured requirements is still a challenge. Here, we analyze the state of art about RE and automation approaches for AmI systems.

Kolos-Mazuryk et al. present a survey of GORE techniques used for AmI system development. In this study, they identify the modeling of context and interactions is still an open challenge. Our research is actually moving in this direction. Sutcliffe et al. (1998) propose a framework for relating scenarios to use cases. Also, works such as Dardenne et al. [1993], Rolland and Salinesi [2009], Uchitel et al. [2004], and Cockburn [2001]) propose combinations of goal techniques with scenarios. Casas et al. (2008) describe how they use *personas* in a project focused on the design of AmI systems. Their work suggests the usefulness of personas as a starting point for user modeling and discusses implications for using this information in the design of the architecture of the system. In addition, our approach tries a step further for providing an integrated effort, covering as well as requirement elicitation, all the other stages of the development.

Concerning the automation topic, two interesting machine-learning approaches are used in the MavHome (Cook et al. 2003) and iDorm (Hagras et al. 2004) projects. These are respectively based on probabilistic and fuzzy approaches, techniques that suffer of the typical training problems that are often not practicable in real contexts. Also, these approaches act on the basis of what happened, according to what they see happening and believe is going to happen, without considering users' desires or knowing users' goals.

3 Requirement Elicitation

The requirement elicitation process is based on the systematic collaboration of Tropos and UCD, with the aim of taking advantages of the synergy of the two approaches. UCD methods (specifically contextual inquiries, scenarios, and

---

[1] The ACube project was founded by the local government of the Autonomous Province of Trento in Italy;http://acube.fbk.eu.

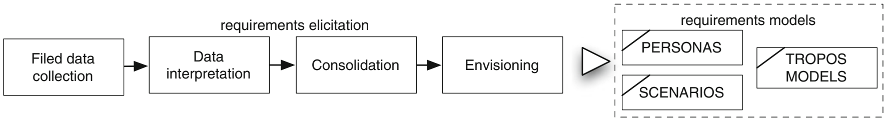


Fig. 40.2 Requirement elicitation process

personas) (Cooper et al. 2007, Dey 2001, Rolland and Salinesi 2009) are used to gain a rich understanding of the domain and to efficiently communicate and negotiate design ideas with stakeholders. On the other side, Tropos models are used to gain an abstract model of the observed domain, and to reason on strategical details for introducing the system.

The requirement elicitation (as shown in Fig. 40.2) is composed of the following four phases, in which techniques from Tropos and UCD are combined:

Field Data Collection – Contextual Inquiry. The initial field data collection activity is performed for acquiring the richest possible understanding of the domain. The first step for a successful design of a product is actually to consider the wide range of stakeholders (Sharp et al. 2007). UCD approach recommends to early define which are the users of the system and their characteristics and to define how to involve them in the design process. Several UCD methods exist in order to get rich insights about the domain. Recently, contextual inquiry (Sharp et al. 2007) demonstrated the capacity to satisfy the needs for a deep, but at the same time rapid, understanding of complex domain. Contextual inquiry mainly consists in interviewing people in their context, preferably when performing their tasks. Resulting data are raw, since it is preferable at this stage to keep the richness of the data and avoid abstraction.

Data Interpretation. The *Tropos Early Requirement* is used for modeling the initial set of domain entities when the system is not yet existing. It includes a bird's-eye view over the domain, in which actors and roles are specified together with their responsibilities and delegations. This view provides an intuition of which interactions occur in the environment. Subsequently each actor is exploited in a goal model, in order to provide details about human behavior, highlighting the rationale by relating each activity to institutional motivations. An example of strategic-view data is the delegation of responsibility [to ensure security in the institute] that is assigned to caregivers. This institutional goal is declined in two caregiver goals: [to avoid guest's aggressive behavior] and [to avoid guest's escape]. Details of these two goals are provided in caregiver's goal model, in which it is defined that the plans to achieve both of them are [direct observation of critical guests] and [observation of critical spaces].

Data Consolidation. The consolidation activity concludes the analysis of the domain by generating two artifacts: personas and scenarios. Their conjunct use increases the ability to identify problems and exceptional cases (Sutcliffe et al. 1998) and to envisage the system (Rolland and Salinesi 2009). *Personas* are descriptive models of system-to-be users based on behavioral data, derived from patterns observed during

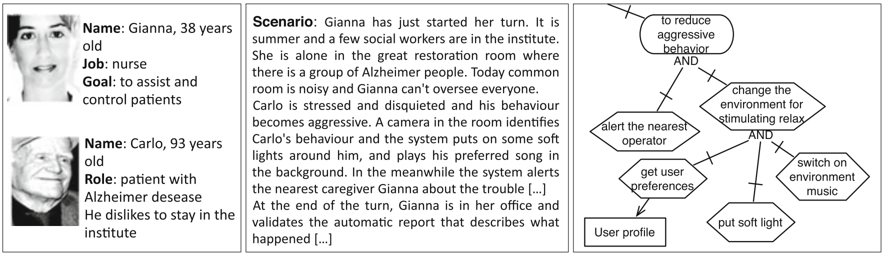


Fig. 40.3 Three produced artifacts: a couple of relevant personas, the scenario of *aggressive behavior* in which they are involved, and the slice of correspondent goal model

interviews, with the aim of representing the diversity of observed motivations, behaviors, and mental models (Cooper et al. 2007). Examples can be found in Fig. 40.3. Personas are provided in the form of rich descriptions of archetype users, meant to focus attention on users goals and motivations both for envisioning and validation purposes. Personas are meant to trigger on emphatic response from the designers and users, in order to support them in taking design decisions at both the cognitive and emotional level.

Envisioning. The envisioning phase aims at specifying the system-to-be and its impact to the domain. Many techniques can be profitable for envisioning system functionalities and services; examples are internal meetings, brainstorming, focus groups, and scenarios. The outcomes of this phase are the list of requirements (modeled in the Tropos language) and a set of technological scenarios.

Tropos Late Requirements. Whereas the early requirement focuses on the domain as it is, this phase introduces the system as a new actor of the domain. In particular, some responsibilities (currently assigned to users) will be delegated to the *system*, while other new dependencies are discovered among actors (both human and software) of the domain. Specific emphasis is given to modeling why the system is necessary and how it will modify current human practices. This enables the analysis of the system impact to the social organization and to the actors. Finally, the system actor is deeply analyzed in order to specify how it will accomplish the goals that are delegated by the humans. For instance, the [direct observation of critical guests] (identified in the early requirements phase) is delegated to the system, and it is decomposed into [to reduce dangerous situations] and [to reduce aggressive behavior]. In particular, designers defined that the system plans for achieving the goal [to reduce aggressive behavior] are (see Fig. 40.3) the following: [alert the nearest caregiver] and [change environment parameters] (e.g., put soft light and play music).

Technological Scenarios. Scenarios are short narrative stories that represent personas in their context, supported by the envisaged technology (see Fig. 40.3). Scenarios concretely describe the behavior of services as experienced by specific, though fictional, users. Stories help the design teams in negotiating a shared representation of the domain and hence a more effective collaborative elicitation of requirements.

4 Identifying and Modeling Users' Behavior Patterns

The behavior patterns to be automated must be identified and modeled from the obtained requirements. To do this, the requirement models are transformed into two executable models: a context model (which specifies the context on which the behavior patterns to be automated depend), and a task model (which describes the tasks that must be carried out for each behavior pattern according to the context described in the context model). Before discussing the transformation, we briefly describe these target models. More details about them can be found in Serral et al. [2010a].

ContextModel. The context model semantically describes the context for properly automating behavior patterns. This model is based on a context ontology proposed in Serral et al. [2010b]. The ontology provides classes such as *User*, which captures information about the users; *Policy*, which captures the system permission for each user; *Location*, which captures information about the locations of the environment where the system is deployed; *EnviromentProperty*, which captures the values of the properties sensed in the environment; and *TemporalProperty*, which captures temporal information. Some of the most important classes of this ontology are also shown on the left of Fig. 40.4. The specific context of the system is represented as instances of the ontology classes. For instance, the *restoration room* should be defined as an instance of the *Location* class, the *noise level* should be defined as an instance of the *EnviromentProperty* class, etc. On the right of Fig. 40.4, an example of context model in a tree form is shown.

Context-Adaptive Task Model. The context-adaptive task model allows the behaviour patterns to be specified by splitting them up into simpler tasks whose execution adapt to the context. We have selected a task model mainly for two reasons: (1) tasks center the requirements modeling process around the user's own experiences (Lauesen 2003), and (2) it allows great expressivity (Johnson 1999), which is needed for accurately specifying user behavior patterns in such a way that they can be automated from their specification.

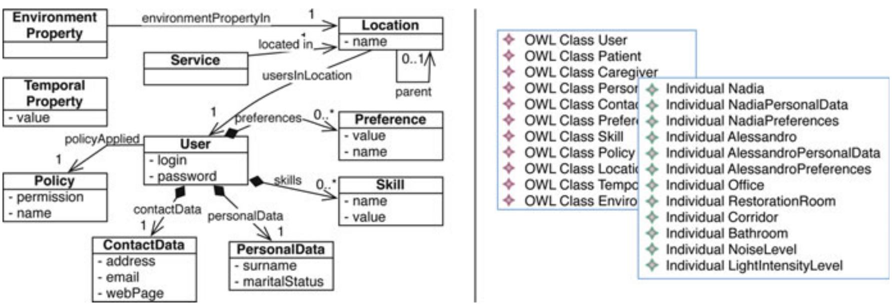


Fig. 40.4 Example of context model

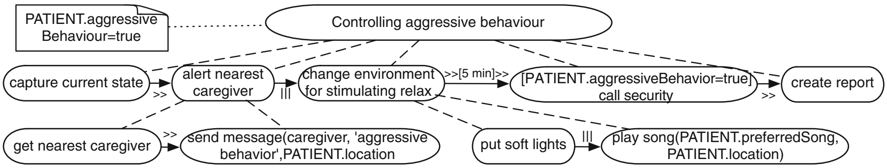


Fig. 40.5 Examples of behavior pattern modelling

The proposed context-adaptive task model is based on the *hierarchical task analysis (HTA)* technique (Shepherd 2001), which breaks down tasks hierarchically into other tasks. We propose to define a task hierarchy for each behavior pattern. Figure 40.5 shows, for instance, the *controlling aggressive behavior* pattern. The root task represents the behavior pattern and has an associated context situation, which defines the context conditions whose fulfilment enables the execution of the behavior pattern. It also has a priority (high, medium, and low) to establish the execution priority of the pattern in case several patterns are enabled at the same time. This root task can be broken down into composite tasks (which are intermediate tasks) and/or system tasks (which are leaf tasks). The composite tasks are used for grouping subtasks that share a common behavior or goal. The system tasks represent atomic tasks that have to be performed by the system (e.g., play a song). Hence, each system task has to be related to a pervasive service that can carry it out. The relation is established by means of the name of the service and the name of its corresponding operation (e.g., multimedia service and playMusic operation). Also, system tasks can have output and input parameters, which can be context parameters.

Both composite and system tasks can have a context precondition that are represented between square brackets. If the precondition is not fulfilled, the task will not be executed. In addition, every task has a name (which explains the task in a user-comprehensible way) and an internal ID (which is a unique identifier).

A behavior pattern or a composite task is broken down into simpler tasks that are related between them using temporal operators that determine the task execution order. This is known in HTA as a plan (Shepherd 2001). We base the temporal operator definition on (Paternó 2002) which provides one of the richest sets of temporal operators. For instance, the $|=|$ relationship means that the related tasks can be performed in any order, the $\gg$ relationship means that the target task will be executed when the first finishes, and the $t \gg$ relationship means that the target task is enabled after t minutes. These temporal relationships can be also used between behavior patterns to allow the composition of patterns.

Note that we make the specified tasks automatically adapt to context by using context parameters. To refer a context parameter, the name of the context property and the name of the individual to which this property belongs have to be specified (e.g., the *aggressive behavior* context property of the *Carlo* individual). However, to deal with the automation of behavior patterns for a group of users, instead of specifying the context property of a user individual, it can be specified as a context property of the ontology class that groups together the corresponding user individuals. For instance, the *controlling aggressive behavior* pattern has to be executed for every patient in which aggressive behavior is detected; therefore, instead of specifying the same behavior pattern for each patient, we specify the behavior pattern once and use in its context situation the *aggressive behavior* context property of the *PATIENT* class, indicating by using capital letters that it is an ontology class, and therefore, the context condition has to be checked for every *patient* individual.

5 From Requirement Models to Executable Models

The methodology we propose is completed by a set of guidelines which allow context and task models to be created from personas, scenarios, and Tropos models. The activity cannot be automated because it cannot leave aside interpretation and reasoning. It is also possible that moving from requirements to executable models, the designer discovers lacks and incongruences; this is solved by starting a new contextual inquiry iteration, by going back to final users with new questions.

Step 1: Detect the behavior patterns to be automated. The step consists in identifying the behavior patterns that can be automated by the system. To identify them, the Tropos goal model is used. A one-to-one relationship is identified between goals delegated to the system and behavior patterns (e.g., the goal [to reduce aggressive behavior] could be transformed into a behavior pattern named *controlling aggressive behavior*).

Step 2: Model the task hierarchy of each behavior pattern. Each behavior pattern is specified using a task hierarchy, from more general to more specific tasks. This hierarchy is obtained from the task decomposition of the corresponding goal in the Tropos goal model. This is completed with the information provided by the technological scenarios: the action verbs whose subject is the system represent tasks to be automated (e.g., *the system plays his (Carlo) preferred song*). An example of task hierarchy obtained following this guideline is shown in Fig. 40.6.

Step 3: Specify users. The users involved in the tasks to be automated are identified. The Tropos actor model and personas provide useful information for creating a hierarchy of users, which has to be specified in the ontology as subclasses of the *User* class. For instance, the actor model identifies the roles *caregiver* and *patient*, while the personas instrument identifies more specific type of users: Carlo, who is a *patient with Alzheimer disease*, and Gianna, who is a *nurse* which is a type of caregiver. Real users will be specified in the hierarchy as individuals of the class that better represent their characteristics.

Step 4: Specify context. Tasks to be automated usually depend on context information. This context information appear in the scenarios as adjectives (e.g., *noisy*), locations (e.g., *restoration room*), temporal aspects (e.g., *season*), etc. Also, the motivations of the goals specified in the Tropos goal model can be used for detected

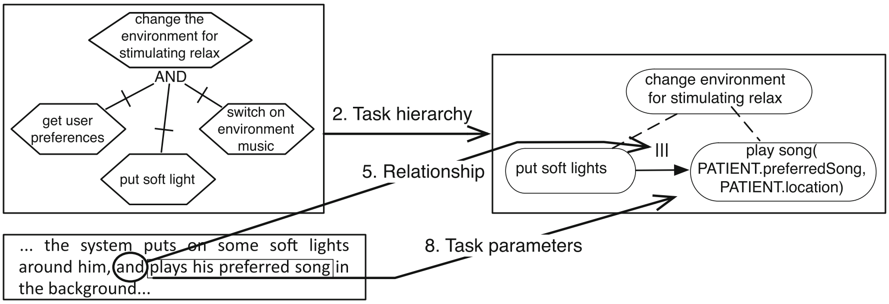


Fig. 40.6 Transformation performed following the provided guidelines

needed context information (e.g., *aggressive behavior*). The identified context properties must be specified in the context model as individuals of the corresponding ontology classes (e.g., noisy should be an instance of the EnvironmentProperty class).

Step 5: Specify temporal relationships. If a behavior pattern, or a composite task, has been refined by temporal refinements, its subtasks have to be related between them by using temporal relationships that rigorously specify the execution order of these subtasks. Scenarios can help to define these relationships. For instance, as shown in Fig. 40.6, in the scenario it is explained that the systems put soft lights and play Carlo's preferred song; meanwhile, the system alerts the caregivers. From this information, we can deduce that the order of execution of these tasks is not important; then, the $|=|$ relationship must be used.

Step 6: Specify the context situation. Each behavior pattern has to be related with a context situation whose fulfillment activates the execution of the pattern. The meaning of the goal to be achieved as well as the technical scenarios can help to define these context situations. For instance, to achieve the goal [to reduce aggressive behavior] , the identified controlling aggressive behavior pattern must be activated when an aggressive behavior is detected, as also is explained in the scenario shown in Fig. 40.3 (*...his behavior becomes aggressive. A camera in the room identifies it and the system ...*).

Step 7: Specify context dependencies. The specified tasks may have to be executed only when some context conditions are satisfied. Thus, these conditions are specified as task preconditions by using the context properties identified in Step 3. For instance, the *call security* task will be executed if the user continues behaving aggressively after executing the previous tasks; then, the context precondition $aggressiveBehavior = true$ must be added to this task.

Step 8: Specify task parameters. If a system task needs parameters to be performed. To detect these parameters, resources in goal models and technological scenarios are used. An example from the scenarios is shown in Fig. 40.6: the text *the system plays his (Carlo) preferred song*, is used for detecting the 'PATIENT. preferredSong' parameter of the task *play song*.

6 Automating and Evolving Behavior Patterns

In this section, we introduce a software infrastructure that allows the specified user behavior patterns to be automated. The infrastructure interprets the task model and the context model at runtime and executes the behavior patterns in the opportune context. This infrastructure (shown in Fig. 40.7 and presented in detail in Serral et al. 2010a) is defined by the following elements, which are implemented by using Java and OSGi technology:

Pervasive Services. A smart environment provides users with pervasive services that control the devices deployed in the environment (e.g., switching lights on, playing music, etc.) and sense context (e.g., detection of presence, measurement of temperature, etc.). We have used a model-driven development (MDD) method (Serral et al. 2010b) to automatically generate the Java/OSGi code of these services from a set of models. The implementation of these services is out of the scope of this work.

Model Management Mechanisms. Provide high-level constructors for managing the models at runtime. Using these mechanisms, any element of the task model and any individuals of the context model can be managed. These mechanisms can be downloaded from http://www.pros.upv.es/art/.

Context Monitor. Context changes are physically detected by sensors, which are controlled by the pervasive services. The context monitor is continuously monitoring these services to capture the context changes. When a change is detected, the monitor processes it and updates the context model accordingly using the model management mechanisms. For instance, if aggressive behavior in a user is detected, the *aggressivenbehavior* context property of the corresponding *User* individual is updated by the context monitor, which also informs the automation engine (next explained) about this update.

Automation Engine. Is in charge of automating the specified behavior patterns in the opportune context. When the engine is informed about a context update, it analyzes the task model to check if some behavior pattern must be executed in the new context. If so, the engine performs the corresponding behavior patterns respecting their priorities. To perform each behavior pattern, the engine executes its system tasks according to

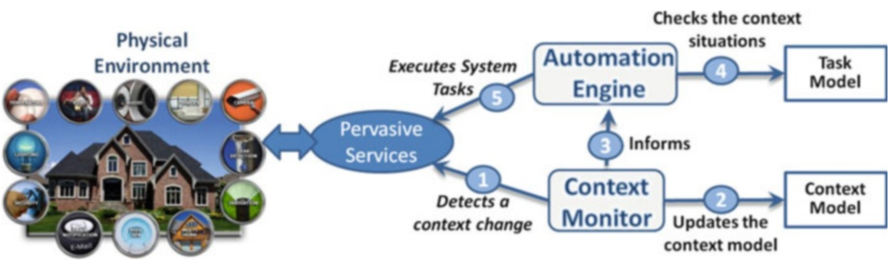


Fig. 40.7 Software infrastructure for automating user behaviour patterns

the temporal relationships specified among its tasks and the current context information (stored in the context model) on which tasks and relationships depend. To execute each system task, the engine searches for the service associated to that task and executes it with the corresponding parameters. For instance, if the context situation of the *controlling aggressive behavior* pattern is fulfilled (see Fig. 40.5), the system captures the current context state. Then, the system searches for the caregivers nearest to the patient location and then sends them a message to warn that aggressive behavior has been detected in the corresponding location. Next, the system puts soft lights and plays the preferred song of the patient that is behaving aggressively. Five minutes later, if the patient is still behaving aggressively, the system warns the security officers. Finally, the system creates a report about the incident.

It is worth noting that rather than translating the models into code, the automation engine directly interprets them at runtime to automate the behavior patterns as specified. This considerably facilitates the evolution of the patterns at runtime: as soon as the models are changed, the changes are applied in the system. To carry out this evolution, the model management mechanisms can be easily used since they provide concepts of a high level of abstraction (e.g., preference, task, behavior pattern, etc.).

7 Discussion

In this work, we have presented a novel methodology for developing user-intensive AmI systems capable of automating the behavior patterns of their users. This methodology supports all the stages of the development process. To achieve this, the methodology integrates a requirement elicitation process that combines UCD and GORE techniques with a software infrastructure that automates the behavior patterns identified in the automation requirements. The feasibility of the approach has been successfully evaluated by developing the ACube case study. In this case study, four scenarios, five personas, and a Tropos model were specified, thus producing 73 requirements among functional and nonfunctional ones. From these requirements, four behavior patterns were obtained and specified using the context and task models. Using these models, the behavior patterns were correctly automated by the provided software infrastructure. Detailed information about this evaluation can be found in [report].

Benefits and Drawbacks. The proposed methodology has been conceived for AmI settings in which the presence of humans is relevant and the difficulty is to build a believable knowledge of the domain and to precisely identify users' needs. The proposed approach is grounded over social science techniques that reduce the gap between analysts and the observed domain: the knowledge is directly extracted with users' participation to discover their real needs. In particular, the methodology is suitable for a quick prototyping approach, in which the development is supported by frequent deployment of mock-ups and prototypes to submit to users' validation.

An additional strength of the approach is the high level of customization of the infrastructure that allows for design-time adapting the system to different setting, for instance, nursing homes with different services. In addition, the model interpretation strategy considerably facilitates the runtime evolution of the automated behaviour to adapt it to changes in users' needs. Mechanisms and tools inspired by end-user techniques have been developed to allow this evolution (?); however, it has to be still performed by users. To achieve a more automatic evolution, we plan to extend the provided infrastructure with machine-learning algorithms that automatically detect the adaptations that must be performed in the system and change the models accordingly if users so desire.

Finally, although the approach has been evaluated by applying it into a real case study, more experimentation is needed. Thus, we also plan to apply the approach in several case studies of different domains.

References

Bresciani P, Perini A, Giorgini P, Giunchiglia F, Mylopoulos J (2004) Tropos: an agent-oriented software development methodology. In: Proceedngs of the AAMAS. IEEE Computer Society, Los Alamitos, pp 203–236

Casas R, Blasco Marín R, Robinet A, Delgado A, Yarza A, McGinn J, Picking R, Grout V (2008) User modelling in ambient intelligence for elderly and disabled people. In: Proceedngs of the computers helping people with special needs. Springer, Berlin/New York, pp 114–122

Cockburn A (2001) Writing effective use cases, vol 1. Addison-Wesley, Boston

Cook DJ, Youngblood M, Heierman IEO, Gopalratnam K, Rao S, Litvin A, Khawaja F (2003) Mavhome: an agent-based smart home. In: Proceedings of the PerCom. IEEE Computer Society, Los Alamitos, pp 521–524

Cooper A, Reimann R, Cronin D (2007) About face 3: the essentials of interaction design. Wiley, Indianapolis

Dardenne A, Lamsweerde A, Fickas S (1993) Goal-directed requirements acquisition. Sci Comput Program 20(1–2):3–50

Dey AK (2001) Understanding and using context. PUC

Hagras H, Callaghan V, Colley M, Clarke G, Pounds-Cornish A, Duman H (2004) Creating an ambient-intelligence environment using embedded agents. IEEE Intel Syst 19(6):12–20

Johnson P (1999) Tasks and situations: considerations for models and design principles in human computer interaction. In: Proceedings of the HCI international. Lawrence Erlbaum, Mahwah/London, pp 1199–1204

Kolos-Mazuryk L, Eck P, Wieringa R A survey of requirements engineering methods for pervasive services. In: Proceedings of the workshop on building software for pervasive computing, OOPSLA'05

Lauesen S (2003) Task description as functional requirements. IEEE Softw 20:58–65

Leonardi C, Sabatucci L, Susi A, Zancanaro M (2010a) Ahab's leg: mediating semi-formal requirement to final users. In: Proceedings of the CAiSE'10, Hammamet

Leonardi C, Sabatucci L, Susi A, Zancanaro M (2010b) Exploring the boundaries: when method fragmentation is not convenient. In: Proceedings of the IEEE FIPA workshop on design process documentation and fragmentation, Lyon

Neal DT, Wood W (2007) Automaticity in situ: the nature of habit in daily life. In: Psychology of action: mechanisms of human action, vol 2

Nuseibeh B, Easterbrook S (2000) Requirements engineering: a roadmap. In: Proceedings of the conference on the future of software engineering. ACM, New York, pp 35–46
Paternó F (2002) ConcurTaskTrees: an engineered approach to model-based design of interactive systems. Lawrence Erlbaum Associates
Rolland C, Salinesi C (2009) Supporting Requirements Elicitation through goal/scenario coupling. In: Conceptual modeling: foundations and applications. Springer, Berlin, p 416
Serral E, Valderas P, Pelechano V (2010a) Improving the cold-start problem in user task automation by using models at runtime. In: Proceedings of the ISD'10. Springer, pp 648–659
Serral E, Valderas P, Pelechano V (2010b) Towards the model driven development of context-aware pervasive systems. PMC 6(2):254–280
Sharp H, Rogers Y, Preece J (2007) Interaction design: beyond human computer interaction. Wiley, Chichester/Hoboken
Shepherd A (2001) Hierarchical task analysis. Taylor & Francis, London
Sutcliffe A, Maiden N, Minocha S, Manuel D (1998) Supporting scenario-based requirements engineering. IEEE Trans Softw Eng 24:1072–1088
Uchitel S, Chatley R, Kramer J, Magee J (2004) System architecture: the context for scenario-based model synthesis. In: Proceedings of the 12th symposium on FSE. ACM, New York, pp 33–42
Van Lamsweerde A (2003) From system goals to software architecture. In: Bernardo M, Inverardi P (eds) Formal methods for software architectures. Springer, Berlin/New York, pp 25–43
Weiser M (1995) The computer for the 21st century. Sci Am 78–89

Chapter 41
Adaptive Model-Driven Information Systems Development for Object Databases

Tilmann Zäschke, Christop Zimmerli, Stefania Leone, Minh K. Nguyen, and Moira C. Norrie

1 Introduction

Modern information system development faces the problem that applications may have to run on a large number of platforms with vastly different capabilities and resources, often ranging from mobile devices to desktop computers and further. This is particularly true of information systems based on object database technologies which nowadays are used at both ends of the spectrum with embedded systems as well as massive scientific databases among their target domains.[1] The semantic complexity of the data managed will also vary widely with some applications requiring only the persistence of a handful of classes, while others must manage large, interrelated collections of evolving data objects. Consequently, there is a tension between trying to enrich an object database to support more semantic concepts and greater functionality to ease the development of complex information systems and keeping it as simple as possible to avoid unnecessary overheads at runtime.

In this chapter, we propose a model-driven approach to the development of information systems that is adaptive in the sense that it generates code for a specific variant of an object database framework according to the features of the application model. The variants offer different levels of semantic support on top of the object database ranging from the native system that essentially offers only persistence to a framework that supports rich classification structures, role modelling and associations together with constraints as proposed in Norrie et al. [2008].

[1] http://www.odbms.org/download/Panel.Renaissance.ODBMS.pdf

T. Zäschke (✉) • C. Zimmerli • S. Leone •
M.K. Nguyen • M.C. Norrie
Institute for Information Systems, ETH Zurich, CH-8092 Zurich, Switzerland
e-mail: zaeschke@inf.ethz.ch; zimmerli@inf.ethz.ch; leone@inf.ethz.ch; norrie@inf.ethz.ch

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_41,

To investigate the approach and provide a proof of concept, we have developed a tool that supports the model-driven development of applications for the object database db4o.[2]

We will first present the background to this work in Sect. 41.2 before describing our approach in Sect. 41.3. Details of the development process and code generation are then given in Sect. 41.4 with a description of the implementation and tools on which it is based given in Sect. 41.5. Concluding remarks are given in Sect. 41.6.

2 Background

It has been recognised that the one-size-fits-all approach to database management systems (DBMS) cannot hope to meet the wide range of requirements and settings of modern information systems (Stonebraker and Cetintemel 2005). One alternative is to develop a DBMS that can adapt to requirements, sometimes referred to as *tailor-made data management* (Apel et al. 2008). Work on component database systems (Dittrich and Geppert 2001) provided adaptability at the architectural level through a modular DBMS architecture. Adaptation of database systems has also been investigated in the area of embedded systems where resources may be very restricted. For example, COMET (Nyström et al. 2003) is a component-based real-time database for automotive systems that represents a typical example of a DBMS that can be statically configured for a given application or target device. This configuration process is driven by requirements for each control unit in a car that will access the database. A second example is FAME-DBMS (Rosenmüller et al. 2008) which uses the idea of software product lines for static system composition and uses application code analysis to determine which components of the DBMS need to be included for a particular application. For example, if the join operator is never used, the configured DBMS will not provide that operator and associated query execution methods.

None of these research efforts has fully addressed the issue of adapting the DBMS according to the *semantic features* of the application model. This is understandable in the case of relational databases where the underlying model is based on a single construct. However, in the case of object databases, there have been many proposals to extend the semantic expressiveness of the underlying object model by integrating support for features such as rich forms of classification structures (Norrie 1995), role modelling (Gottlob et al. 1996, Kappel and Retschitzegger 1998, Pernici 1990) and associations (Albano et al. 1995, Norrie 1993). The argument made by researchers is that increasing the semantic expressiveness of the object model greatly simplifies the development effort of complex information systems. In some cases, this has been demonstrated by building semantic layers on top of commercial object databases where data management tends to be based on the object model of the associated

[2] http://www.db4o.com/

object-oriented programming language. For example, in Kuno et al. [1995], a technique called *object slicing* was proposed as a means of supporting object role modelling and implemented in MultiView as a semantic layer on top of GemStone.[3] Being aware of the possible performance penalty that comes with adding a layer to the DBMS, they also conducted experiments assessing the additional cost of implementing object slicing on top of GemStone. Object slicing has also been used to support multiple instantiation and role modelling in more recent work as part of a semantic data management layer for the object database db4o (Norrie et al. 2008).

These more expressive object data models have never been adopted within the commercial sector, and modern object databases such as db4o still base data management on the object model of programming languages such as Java. Thus, in the case of Java, developers tend to view the object database as providing persistent storage for instances of Java classes together with query facilities to support the retrieval of these instances. Commercial object databases provide little or no support for explicitly modelling and managing the semantics of application data, leaving it to the developers to encode most of this implicitly within the application programme.

Our aim is to investigate how ideas from model-driven development and adaptive DBMS can be combined to provide tools to facilitate the development and maintenance of object databases without introducing unnecessary complexity and performance overheads. In this way, the benefits of explicitly modelling and managing data semantics, and clearly distinguishing the conceptual and physical design of databases, attained in information systems based on relational technologies could be carried over to those based on object databases. At the same time, we recognise that there is a trade-off between semantic expressiveness and performance and not all applications require rich modelling features. Our approach is therefore to provide a modelling tool that supports the conceptual design of the application database and generates code that corresponds to the physical database design based on an analysis of the semantic features of the application model. In this way, we effectively offer a variety of semantic layers on top of the object database according to the needs of the application. In the following sections, we describe how this has been achieved for the db4o object database.

3 Approach

We support the process of designing information systems using object database technologies by generating persistent database classes from a conceptual model of the application data. The conceptual model is based on the OM data model (Norrie 1993) which combines features of the enhanced entity-relationship

[3] http://www.gemstone.com/products/gemstone

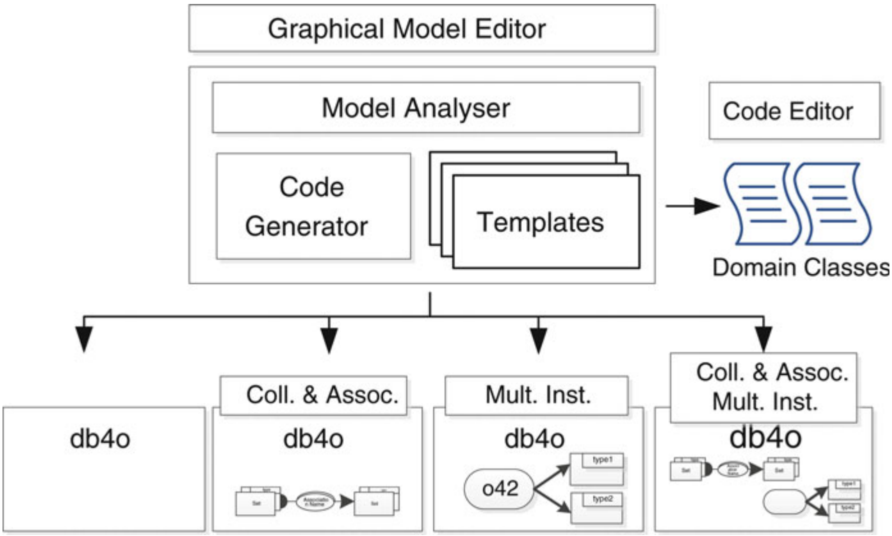


Fig. 41.1 Architectural overview

(EER) model and object models with support for role modelling. The application model may therefore define rich classification structures through constraints over collections of objects, associations with cardinality constraints and constraints over object evolution; none of which are supported natively in object-oriented languages such as Java and hence db4o as used by Java developers. Support for multiple instantiation together with operations to allow objects to gain and lose types forms the basis for flexible role modelling. For example, an object in a publications database might at one point in time be of type `draft` and later lose the type `draft` and gain the type `published`. The object may in parallel be of type `article` or type `book` to indicate the category of publication that it represents.

The application model is constructed using a graphical editor as indicated at the top of Fig. 41.1 which shows an overview of the architecture.

The conceptual model of the application is analysed to determine which semantic features of the OM model have been used, and this in turn determines the target system for code generation. The set of possible target systems effectively provide different semantic layers on top of the object database, which, in the case of the tool that we have developed, is db4o.

Currently, we offer four target systems for db4o as shown at the bottom of Fig. 41.1. The leftmost version is the native db4o system with no semantic features that go beyond the capabilities of the Java object model. With this version of the database, application development is therefore based on the Java object model, and Java classes can be made persistent in order to manage and store instances of these classes in the database.

The other three target systems all extend the semantic capability of db4o by offering a framework that implements all or some of the features of the OM model,

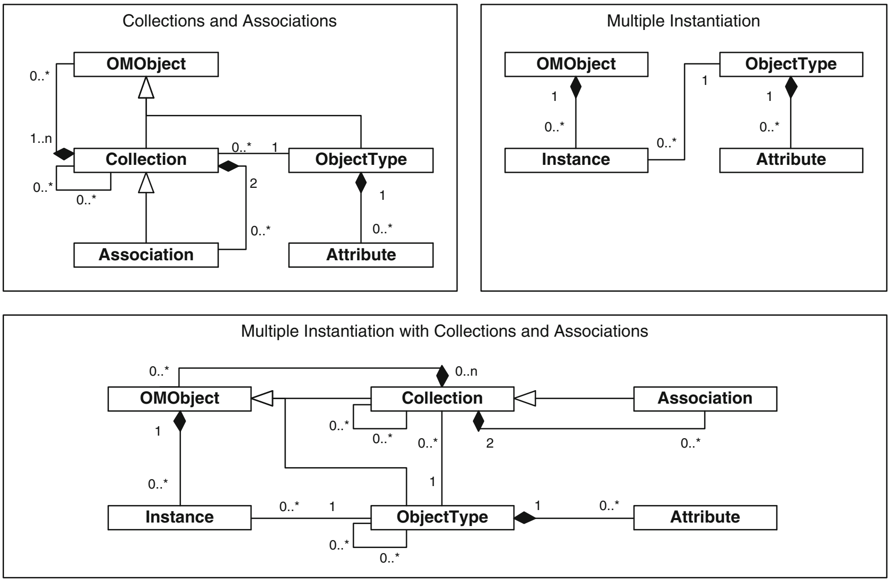


Fig. 41.2 UML class diagrams of the three different semantic layers

thereby enabling the application developer to work at a higher level of abstraction that corresponds to the conceptual model of the application domain.

Second from the left in Fig. 41.1, we show a target system where the Java object model has been extended with support for collections and associations as indicated by the semantic layer on top of db4o and the enhanced data model inside db4o. These constructs allow objects of a specific type to be semantically grouped into collections as well as objects of one collection to be associated with objects of another. Thus, this target system effectively provides a framework on top of db4o that implements the main features of the EER model for an object-oriented environment. The UML diagram in the top left of Fig. 41.2 shows how the corresponding semantic layer is implemented. The diagram depicts the type hierarchy of `OMObject`, `ObjectType`, `Collection` and `Association`. An `OMObject` requires a `Collection` to serve as its type extent. An `Association` is itself a `Collection` since it is a collection of pairs, and since it can associate any two collections or associations, it is possible to model complex relationships through nested associations.

Third from the left in Fig. 41.1, we show a target system that extends the Java object model with support for multiple instantiation and hence support for object role modelling. Multiple instantiation is not a feature of object-oriented languages such as Java but can be supported by building a framework that allows application objects to be dynamically constructed from one or more Java instances as described in Norrie et al. [2008]. As depicted in the top right of Fig. 41.2, this is implemented by the object type pattern (Martin et al. 1997) where an object can be of multiple types through the concept of instances.

The fourth target system depicted on the right of Fig. 41.1 supports multiple instantiation as well as collections and associations. It effectively provides the application developer with an implementation of the full OM model on top of db4o. The UML diagram of this semantic layer is shown in the bottom of Fig. 41.2.

The code generator, which is depicted left of the centre of Fig. 41.1, uses a set of templates to generate the persistent classes for the application as well as the database initialisation code that creates the application model in db4o. The generated classes can then be further extended by the developer with additional application logic using the code editor shown in the same diagram.

4 Model-Driven Code Generation

We illustrate the process of model-driven code generation based on an example application for managing information about publications and authors. Figure 41.3 gives an overview of the development process.The developer uses a graphical editor to construct a model of the application domain based on the OM model which is a two-level model comprising a type model and a classification model as shown in the top left of Fig. 41.3. For our example application, the type model shown in the left part of the application model of Fig. 41.4 defines two types `author` and `publication`. The right part of Fig. 41.4 shows the classification model where the rectangular shapes denote collections of objects with the collection name in the centre and the member type in the top right. Oval shapes denote associations. The classification model defines two collections `Authors` and `Publications` that contain instances of the respective types, two subcollections `Drafts` and `Published` that specify roles for `publication` objects and an `authored_by` association between `Authors` and `Publications`.

After designing the application model, the developer uses the *code generation wizard*, depicted in the top right corner of Fig. 41.3, to invoke the generation process. First, the application model is checked for the use of semantic features not supported by db4o such as collections and associations or multiple instantiation. In our example, the model analysis would recognise the use of collections and associations.

Note that it is not always possible from an application model to determine whether multiple instantiation will occur and therefore is required. However, it is possible to detect models where it is prohibited by the constraints and therefore is not required and can be excluded. For example, if we know that collection C_1 contains all instances of type t_1 and collection C_2 contains all instances of type t_2 and there is a constraint that C_1 and C_2 are disjoint, then it is not possible to create

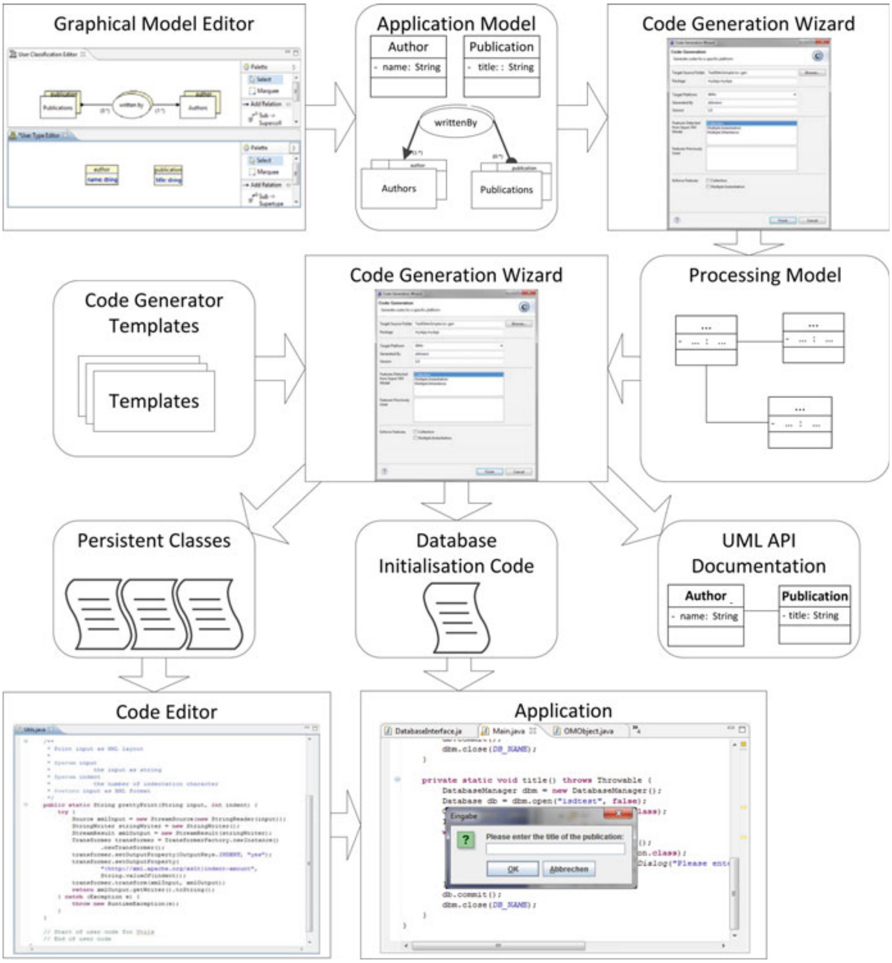


Fig. 41.3 Modules of the code generation plug-in

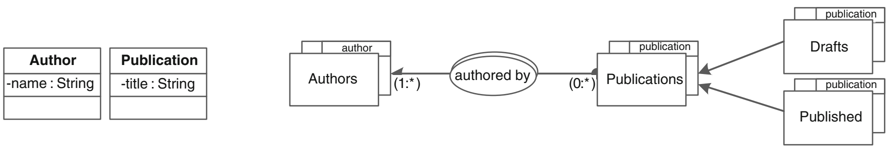


Fig. 41.4 Example of OM typing and classification diagrams

an object that has both type t_1 and t_2. In future work, we plan to extend our approach to include object life cycle diagrams which can be used to assist, not only the design of classification structures, but also determining the role modelling requirements of applications and whether or not multiple instantiation is required.

After the analysis of the application model, the developer is asked to either accept the set of required features or manually enforce or suppress particular features. Enforcing a feature can be useful if the developer knows that a specific feature is not used in the current version of the application model but is likely to be introduced in future enhancements. Although our approach does allow additional semantic features to be added later as part of the support for system evolution, this does come at a cost, and therefore, it is beneficial to give developers the option to use their knowledge to specify that certain features should be supported at this stage even if not required. Similarly, as stated above, it is not always possible to determine whether or not multiple instantiation will actually occur in practice which means that this feature may be provided in cases where it is not actually required: We therefore provide developers with the option of suppressing this feature manually if they know that it will not arise, thereby resulting in a simpler semantic layer.

Once the semantic features are chosen, the developer invokes the model conversion and code generation process. The model converter converts the applications model into a processing model that can be read by and acts as input to the code generator.

The code generator relies on *template files* for code generation. The support for different semantic layers is implemented by using different template files depending on the chosen semantic features. When the code generator is triggered from the Wizard GUI, it chooses the required set of template files and generates the domain classes and, if required, a semantic layer that exposes the required semantic features. In our example, the domain classes provide methods to create, access, update and delete authors and publications in the database. The classes can later be extended by the developer with additional application logic. For the semantic layer, the generator creates API classes for each collection and association. In addition to the domain classes and the semantic layer, the generator creates database initialisation code to create and initialise the new database. Depending on the chosen semantic layer, this includes, for example, instantiating collections and associations.

Finally, the generator also creates UML output that documents the API of the persistent classes as shown in Fig. 41.5. For the example application, the UML diagram shows on the left two classes `Author` and `Publication` which are associated. In the middle is the API of the generated collections and on the right is the API of the association.As shown in the bottom of Fig. 41.3, after the domain classes have been generated, the developer can use the *code editor* to extend them with application logic. While we have shown a sequential application design, we also support an iterative design process. Handling application model updates is quite straightforward. The new domain classes are generated from the new version of the application model and then merged with the existing domain classes. Classes that exist in both versions are analysed, and any user-specific application logic code is copied from the old to the new classes.

However, when the developer chooses different semantic features in the new version, these changes affect the semantic layer and thus the data model of the

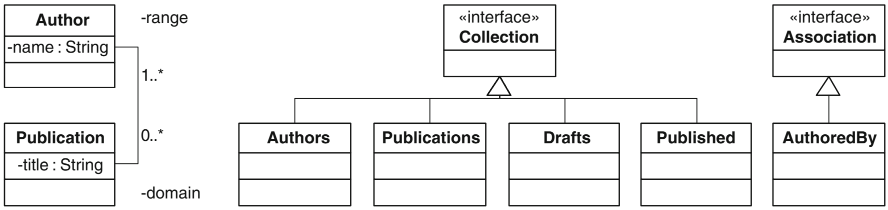


Fig. 41.5 Example of an generated UML diagram

database. Consequently, the resulting domain classes may look quite different. In such a case, user-defined code from old versions is stored in a save location and can later be accessed by the developer, but it is not migrated to the newly generated domain classes.

Changes to the semantic layer may also invalidate queries since they often rely on the specific knowledge of which attribute is stored in which class. To avoid this issue, we provide a query wrapper for db4o. The API of the wrapper is static even when the semantic layer changes, and internally, the wrapper ensures that queries are always valid and working, regardless of the chosen semantic layer.

5 Implementation

To implement our system, we used the Eclipse[4] platform which provides good extensibility through its plug-in capability. The system consists of six components as shown in Fig. 41.6. The left three components form a single Eclipse plug-in. The Wizard GUI controls the code generation process with the model analyser and converter transforming the application model into the processing model for code generation and the code generator generating the code. The model editor on the right is a graphical interface for designing classification and type models. The code editor is part of the standard Eclipse Java development tools plug-in[5] for developing Java code. For code generation, we rely on the Acceleo Code Generator[6] which is also part of the Eclipse platform.

The Wizard GUI is a single-windowed graphical user interface built onthe standard Eclipse environment. The dependency on the model editorcomponent is required for integrating the GUI into the model editor.Dependencies on the model analyser, model converter and code generator arerequired to trigger the respective components. Internally, the Wizard GUI isimplemented as a standard Eclipse `Wizard` with SWT components and multiple`Pages`.

[4] http://www.eclipse.org/

[5] http://www.eclipse.org/jdt/

[6] http://www.eclipse.org/acceleo/

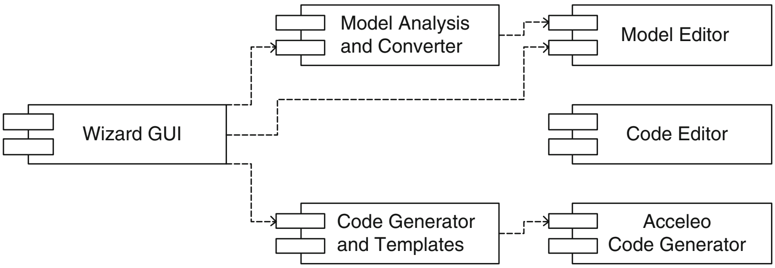


Fig. 41.6 Components and dependencies of the code generation plug-in

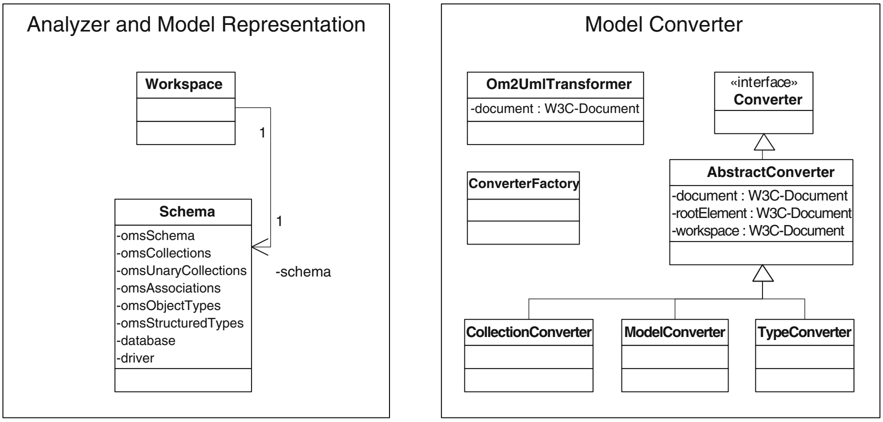


Fig. 41.7 UML class diagram of the analysis and converter component

The model analyser and converter are co-located in a single component because of their overlapping functionality given that they both need to parse and analyse application models. Figure 41.7 shows the classes used in the analyser and converter. In the left box are the classes that provide an internal representation and thus access to a given application model. The `Workspace` class provides access to the application model and implements functionality for model analysis such as `hasOMMultipleInheritance()`, `hasOMMultipleInstantiation()` and `hasOMCollection()` methods. Internally, the `Workspace` uses an instance of `Schema` to access the application model and its elements.

The classes in the right box contain the converter functionality; the entry point is the `Om2UmlTransformer` class. The converter is structured into components for analysing different aspects of an application model represented by the `CollectionConverter`, `TypeConverter` and `ModelConverter` classes. Each of these converter classes implements a single method `convert()` that can be called to convert the application model to a UML-style processing model.

The generated processing model uses UML annotations to describe properties of application models that are not available in UML as illustrated in Listing 1.

It shows the declaration of a collection using UML annotation. The name of the collection is `Authors` as given in the first line, the `details` tags define the behavioural type of the collection as `set` and its member type as `author`.

Listing 1 Excerpt from the generated processing model XML.

```
<packagedElement xmi:type="uml:Class" xmi:id="Authors" name="Authors">
 <eAnnotations xmi:id="_E0ITnM_Ug" source="OMProperty">
  <details xmi:id="_E0IjnM_Ug" key="bulkType" value="set"/>
  <details xmi:id="_E0IznM_Ug" key="memberType" value="author"/>
 </eAnnotations>
</packagedElement>
```

We chose the Eclipse Acceleo framework as the code generator for its flexibility and because it implements the OMG MOF Model to Text Language (MTL) standard.[7] Listing 2 shows an example of the template language that generates getter methods for domain classes in db4o. The statements in square brackets are template commands. Command clauses are terminated by statements that contain a '`/`'. '`p`' is a variable that contains class fields. Some of the functions such `asisCollection()` are user defined, while others such as `toUpperFirst()` are partof the language standard. Listing 3 shows the resulting Java code for a class field`title`.

Listing 2 Simplified template snippet that generates getter methods fordb4o.

```
public [if (isCollection(p))]
          [getCollectionType()/]<[castToJavaType(p)/]>
        [else]
          [castToJavaType(p)/]
    [/if]
     get[p.name.toUpperFirst()/]() {
   [if(hasCollection = 'false']
    activate(ActivationPurpose.READ);
   [/if]
   return this.p_[p.name/];
}
```

Listing 3 A generated getter method featuring db4o activation.

```
public String getTitle() {
   activate(ActivationPurpose.READ);
   return this.p_title;
}
```

[7] http://www.omg.org/spec/MOFM2T/1.0/

The template files are organised in a hierarchy. The root of the hierarchy contains template code that is independent of the semantic layer for which the code is generated. The leaves of the hierarchy contain template code that is specific to the required type of semantic layer. Template functions that are referenced but not present in a leaf are looked up higher in the hierarchy when called. When code generation is started, the generator only has to choose the leaf template for the selected semantic layer and can generate code from it.

6 Conclusion

We have re-examined the issue of extending the semantic features of object databases by building semantic data management layers on top of commercial object databases to ease the development of complex information systems. There is always a trade-off between expressive data models and performance, and this becomes even more critical when considering the wide spectrum of applications and platforms targeted by modern object databases. The approach that we propose is to adopt ideas from adaptive DBMS to tailor the semantic layer to the application model based on an analysis of its semantic features.

We have shown how the approach can be part of model-driven development by implementing an adaptive code generation framework for the object database db4o. The framework enables developers to design the conceptual model of their application abstracted from implementation details and map this onto a physical design by generating code for a semantic layer tailored to the features of their model. At the same time, developers are given fine-grained and guided control over the code generation process to take advantage of their domain knowledge and possible future system requirements. Our next steps include expanding adaptivity to other features including support for additional databases and conducting user studies to verify our approach.

References

Albano A, Ghelli G, Orsini R (1995) Fibonacci: a programming language for object databases. VLDB J 4:403–444

Apel S, Rosenmüller M, Saake G, Spinczyk O (eds) (2008) Software engineering for tailor-made data management. University of Magdeburg

Dittrich KR, Geppert A (eds) (2001) Component database systems. Morgan Kaufmann, San Francisco

Gottlob G, Schrefl M, Rock B (1996) Extending object-oriented systems with roles. ACM Trans. Inf. Syst. 14(3):268–296

Kappel G, Retschitzegger W (1998) A comparison of role mechanisms in object-oriented modeling. In: Proceedings of the GI-workshop modellierung'98, Münster

Kuno HA, Ra Y-G, Rudensteiner EA (1995) The object-slicing technique: a flexible object representation and its evaluation. Technical report CSE-TR-241-95, University of Michigan

Martin RC, Riehle D, Buschmann F (eds) Pattern languages of program design, vol 3. Addison-Wesley, Reading

Norrie MC (1993) An extended entity-relationship approach to data management in object-oriented systems. In: Proceedings of the international conference on the entity-relationship approach (ER 1993), Arlington

Norrie MC (1995) Distinguishing typing and classification in object data models. In: Kangassalo H, Jaakkola H, Ohsuga S, Wangler B (eds) Information modelling and knowledge bases VI. IOS, Amsterdam

Norrie MC, Grossniklaus M, Decurtins C, de Spindler A, Vancea A, Leone S (2008) Semantic data management for db4o. In: Proceedings of the international conference on object databases (ICOODB 2008), Berlin

Nyström D, Nolin M, Norström C, Hansson J (2003) COMET: a component-based real-time database for automotive systems. In: Proceedings of the workshop on software engineering for automotive systems, Magdeburg

Pernici B (1990) Objects with roles. ACM SIGOIS Bull. 11(2–3):205–215

Rosenmüller M, Siegmund N, Schirmeier H, Sincero J, Apel S, Leich T, Spinczyk O, Saake G (2008) FAME-DBMS: tailor-made data management solutions for embedded systems. In: Proceedings of the EDBT workshop on software engineering for tailor-made data management (SETMDM 2008), Nantes

Stonebraker M, Cetintemel U (2005) One size fits all: an idea whose time has come and gone. In: Proceedings of the international conference on Data Engineering (ICDE 2005), Tokyo

Chapter 42
Implementing a Service-Oriented Architecture: A Technochange Approach

Eli Hustad and Lene Staverløkk

1 Introduction

The history of information technology (IT) is relatively short; however, since the early 1960s, many organizations have developed IT portfolios, which have resulted in a complex mix of different legacy systems, enterprise systems, platforms, and applications inside an organization. To obtain required integration between systems, the numbers of point-to-point integration and middleware solutions (e.g., enterprise application integration (EAI)) have been increasing (Lam 2005). Consequently, the resources required for development and maintenance of these compound solutions have become extensive.

Recently, there has been increased attention toward *service-oriented architecture (SOA)* as a platform to approach challenges related to development and maintenance of heterogeneous IT portfolios (Baskerville et al. 2010; BEA Systems 2005; Legner and Heutschi 2007). SOA makes promises for increased flexibility and reusability because of its module-based architecture and services offered (Hirschheim et al. 2010). The focus on services makes SOA unique since it provides transparency across multiple (legacy) applications and data sources that are black boxed. Because of open standards that define the services, a common pool of IT resources becomes available despite different IT systems, functionalities, language codes, and platforms.

The implementation of SOA, however, requires long-term projects that involve comprehensive organizational changes (Baskerville et al. 2010; Hirschheim et al. 2010). According to Hirschheim et al. (2010), SOA should be utilized as a business

E. Hustad (✉)
Department of Information Systems, University of Agder, Kristiansand, Norway
e-mail: Eli.hustad@uia.no

L. Staverløkk
IT and Service Department, Norwegian Tax Administration, Oslo, Norway
e-mail: Lene.Staverlokk@Skatteetaten.no

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_42,

transformation tool for solving larger business needs, rather than being an IT architectural initiative only. So far, SOA research has mainly been technology oriented, and there is a need for studying the strategic, organizational, and managerial issues associated with SOA (e.g., Viering et al. 2009). Some researchers have proposed a research model for studying the relationship between business process standardization and SOA (Beimborn et al. 2009). Moreover, models to SOA governance have been focused in some research studies (Niemann et al. 2008; Schepers et al. 2008). However, with an exception of a few *empirical* studies conducted on early SOA adopters (Legner and Heutschi 2007), factors influencing the strategic alignment of SOA development with business objectives (Antikainen and Pekkola 2009; Baskerville et al. 2010), interorganizational collaboration, and SOA (Ordanini and Pasini 2008), there are few studies focusing on the organizational issues and system development (SD) challenges related to SOA implementation. This research seeks to bridge this gap, and we conducted a field study in the IT and Service Department (ITSD) of the Norwegian Tax Administration (NTA). The organization started a SOA program in 2008. The following research questions guided our research: *How are organizations implementing a service-oriented architecture? What are their motivations? What are the critical issues?* To explore these issues, we investigated the SOA implementation process to gain knowledge about how an organization accomplishes this process in practice.

This chapter is organized as follows. Section 2 conceptualizes SOA and presents related research. Section 3 provides a framework for implementing SOA, and Sect. 4 introduces the research site and method. Section 5 reports the results of the empirical study, while Sect. 6 provides a discussion of the results and the implications of this research. Finally, in Sect. 7, we make some concluding remarks.

2 The Concept of SOA and Related Research

The focus on IT architecture has become an important issue within organizations. Over the last decades, technologies have gone through remarkable changes, and organizations have tried to adapt to these changes and have invested in and developed system and technology portfolios, which over time have become complex and heterogeneous. Thus, it has become important for companies to define an enterprise strategy that includes business strategy, IT strategy, organization, and IT architecture (Venkatraman et al. 1993). Ross (2003) has studied EAs, and she defines IT architecture for an enterprise as follows: *An IT architecture is the organizing logic for applications, data and infrastructure technologies, as captured in a set of policies and technical choices, intended to enable the firm's business strategy* (ibid., p. 5). She describes four stages or maturity levels to explain how IT architecture develops over time. These stages are application silo, standardized technology, rationalized data, and modular. The modular architecture is an enterprise-wide architecture based upon standards and loosely coupled applications, data, and technology components. Advantage with this kind of architecture is the possibility to make fast adaptations in the solutions and the possibility to reuse code in SD. According to Ross (2003), the largest risks with a modular architecture are firstly, that enterprises introduce modules before they have established a rationalized data architecture and secondly the lack of governance mechanisms. However, in a modular architecture, the resources can be utilized to improve the infrastructure instead of doing local customizations in the applications.

SOA typically represents a modular architecture. There are several conceptualizations of SOA in the literature. IBM defines SOA as *...an architectural framework and approach that takes everyday business applications and breaks them down into individual business functions called services. The primary structuring element for SOA application is a service as opposed to sub-systems, systems, or components* (IBM Corporation by Chen 2006).

Conceptualization of SOA does also combine business, technology, and IT management perspectives (Ordanini and Pasini 2008). Firstly, from a business perspective, SOA represents a set of services for improving the capability of organizations to do business with its customers. Secondly, from a technological view, it is characterized by modularity, service reuse, new programming methods based upon standards, and tools involving web services. Thirdly, from an IT management perspective, SOA provides a new method for designing IT application portfolio. In addition, implementation of SOA brings about new challenges related to organizing, IT governance, and change management. A critical aspect of SOA implementation is the focus on the alignment between technology and business, since without this alignment, the potential of SOA will not be optimized (Antikainen and Pekkola 2009). Baskerville et al. (2010) conducted a comparative case study of implementation of SOA in the banking sector. They found that SOA adoption may provide greater organizational agility and competitiveness. However, SOA was rather a means to drive organizational strategy, which focused on the alignment of business and technology for agility. SOA is thus becoming more than an IT architecture with services defined by the IT department. SOA is rather a driver for a business architecture to support a service-oriented enterprise (Hirschheim et al. 2010).

According to the above-mentioned issues, implementation of SOA is a complex organizational change process, which requires strategic, business, and technological considerations.

3 A Framework for Implementing SOA

In the IT implementation literature, there are several empirical studies and frameworks focusing on enterprise-wide implementation of different kinds of enterprise information systems (Lam 2005; Markus 2004; Robey et al. 2002). According to Markus (2004), a technochange (technology-driven organizational change) project is a project which requires a socio-technical focus by combining the

IT implementation processes with the organizational change processes to ensure optimal utilization of the IT solution. We view a SOA implementation project as a typical technochange project.

In the SOA literature, there are roadmaps for implementing SOA successfully and suggestions for SOA governance models (e.g., Niemann et al. 2008). BEA Systems (2005) presents a domain model for implementing SOA and emphasizes the importance of not looking at a SOA project as a technical project only. According to BEA, the implementation should occur incrementally over several years. The BEA framework emphasizes that SOA implementation will encompass organizational transformation and changes in business processes and management models. It is therefore a strong need for alignment between business strategy and IT strategy. Moreover, the framework focuses on the importance of establishing an enterprise architecture (EA) and governance mechanisms to administer the EA. The framework points to the significance of developing reusable components and the need for taking care of and further develops the competency of the organization. The implementation costs of SOA might be high; however, the organizational value of SOA increases the more the architecture and its services are used. We combine the BEA domain model with relevant research studies to substantiate the statements of this model. The framework (Table 42.1) was utilized as a point of departure for guiding the SOA implementation in this study.

4 Research Site and Method

The research site of this study is the IT and Service Department (ITSD) unit of the Norwegian Tax Administration (NTA). NTA reports to the Ministry of Finance of the Norwegian Government and needs to follow institutional regulations and governmental decisions. Totally, the NTA organization has approximately 7,000 employees working at different units. ITSD, which is one of these units, has approximately 830 employees and is responsible for developing, operating, and maintaining NTA's IT portfolio.

In 2008, NTA went through an organizational development (OD) project, which included a review of existing business processes and work routines. This resulted in comprehensive changes in business processes, establishment of a new organizational structure, changes in collaboration models, and several employees got new work tasks. The change process caused instability and resistance in the organization; however, after a stabilization period, new processes were accepted and settled. This reorganizing project was a way to prepare the organization for implementation of SOA. NTA introduced the concept of services and analyzed the value chain to identify the most important business tasks. Along with these changes, ITSD developed a new IT strategy. The new strategy emphasizes the importance of an integrated system architecture based upon shared components, reusability, and service orientation. Newly, a similar OD project has started at ITSD.

Table 42.1 A framework for implementing SOA

Domain 1. Business strategy and process
Defining enterprise strategy, aligning business objectives and IT capabilities (e.g., Lam 2005; Ross 2003; Venkatraman et al. 1993), defining a SOA program and related challenges (BEA Systems 2005; Ross 2003)
Domain 2. Architecture
Defining strategic goals for EA, technical architecture, maturity level, and services (Ross 2003). Do IT portfolio analysis to determine how new IT integration mechanisms make influence on existing architecture and applications (Lam 2005). Build knowledge and competencies on EA, internalize the knowledge, and establish a mindset for architecture principles (Markus 2004). Ensure collaboration between business and IT professionals and embed cross-functional learning into the culture.
Make an image (reference architecture) of the strategic target of the EA (BEA Systems 2005; Lam 2005; Ross 2003). Be aware of the risk of skipping one maturity level (Ross 2003)
Domain 3. Building blocks (BB)
SOA is (1) software BB (code, data models, processes, services, applications) (BEA Systems 2005; Hirschheim et al. 2010; Ordanini and Pasini 2008; Ross 2003) and (2) organizational BB (best practices, standards, templates, routines, guidelines for design, development, operation, and governance of services). Reusable software components reduce costs and provide opportunities for developing new services (BEA Systems 2005; Ross 2003). Organizational BB should become reusable components constituting the infrastructure of the enterprise (BEA Systems 2005, 2008)
Domain 4. Project and applications
Identify and prioritize SOA projects of highest organizational value and analyze the information systems (IS) portfolio (BEA Systems 2005). Consider available competence and make realistic project plans. Include change management activities and communication plans, user involvement, collaboration, and training (Lam 2005; Markus 2004). Prototype design and migration approaches; concurrently, prototype a technical solution and the related organizational change (Markus 2004)
Domain 5. Organization and governance
Establish new governance models and be aware of changes in power balance (from silo to modular); decision-making processes will involve people outside the silo (Ross 2003). Many technochange projects fail (Markus 2004)and enable organizational and cultural adaptation, support from management, adapt the solution to organizational context, business goals, and IT strategy (BEA Systems 2005; Lam 2005; Markus 2004). Reasons for failures: organizational issues, misfit between the new IT solution, and the organizational culture. Establish efficient collaboration structures, discuss architectural issues, and develop functionalities for end users. Clarify forthcoming changes in roles and responsibilities for the IS professionals (Lam 2005; Markus 2004) and ensure technical and behavioral integration (Lee et al. 2003).
Domain 6. Costs and benefits
The investment costs of SOA are high; benefits from SOA became visible over time by using the architecture (BEA Systems 2008; Lawler and Howell-barber 2008). SOA implementation is a long-term project. Make a strong business case when deciding to implement SOA, evaluate economic, business, and strategic reasons. Benefits: reduced SD costs (reuse of services, out-phasing existing applications). Establish good metrics and make values visible. The resource allocations are decreasing when moving between maturity levels toward a modular architecture (Ross 2003)

One of the authors works at ITSD and is directly involved in the SOA program. She conducted a qualitative field study comprising participating observation, interviews, and document analysis. Thus, this study has elements of an action research design (Reason and Bradbury 2001). This design has benefits providing the opportunity to study a phenomenon closely in the organization, which would be almost impossible for an outsider. The research study started in 2008 with a pilot study of an EAI project and continued with the main study focusing on SOA implementation and the establishment of an EA (2009–until now). Firstly, the action researcher working at ITSD *diagnosed* the organization and its challenges and provided a recommendation of *planned* actions related to each domain of the BEA model. *Diagnosing* and *planning* constitute two stages of an action plan cycle as defined by Susman and Evered (1978). These recommendations were communicated to the CIO, the CEO, and the system architect manager.

Mostly, data collection consisted of participating observation in terms of taking part in meetings, by being actively involved in the Platform project (see Table 42.4, SOA bus) as a team participant and by being a project leader for development of SOA pilot services. The Platform project delivered principles and guidelines for using the integration platform, proposed governance models and related routines, and developed documentation. Proposed governance models were communicated to different management groups, the IT staff, and the system architects.

In total, 15 semi-structured, face-to-face interviews were conducted. We choose interviewees to cover different hierarchical levels: top management, line managers, project leaders, system developers, and test engineers. The intranet of NTA was the main source of the secondary material and included strategy and project documents, IT portfolio management reports, and institutional rules and documents. The process of data collection and analysis proceeded iteratively, allowing themes to emerge and then to be examined more deeply as relevant. The study has an interpretive approach and has followed the principles of Klein and Myers while conducting and evaluating the findings (Klein and Myers 1999).

5 Results

NTA has continuously developed IT solutions from 1969 until today. The organization is an innovative public institution and has been an early adopter of new technologies and development tools. This has resulted in a complex IT portfolio with several application silos. Table 42.2 depicts the various technological platforms and core applications and the year of implementation. According to Ross' maturity levels of architectures (Ross 2003), the IT portfolio at NTA consists of system solutions with different maturity (Table 42.3).

The executive and transaction systems in NTA represent application silos consisting of approximately 90 different applications. Most of those are in-house developed core applications. The remaining are standardized enterprise systems (e.g., SAP) or business support systems. The internal integrations (e.g., point-to-point and EAI) cause data duplicity among applications. Additionally, many application silos have user interfaces toward the same external partners. Most of the information exchange between partners has until now been file based and batch

Table 42.2 Standardized technology architecture for application silos

Decade	Type of server	Operative system	Database	Development tool
1969	Mainframe	OS390	DB2	COBOL
1981	Minicomputer	OS400		RPG
1985	Minicomputer	Aix		Pro-Iv
1997	Minicomputer	Aix	Oracle	Pl/sql
2008/2009	Minicomputer	Aix	Oracle	JDeveloper/Workshop
2010	PC	Linux	Oracle	JAVA

Table 42.3 Maturity levels of NTA's IT architecture (A)

Application silo	Standardized tech A	Rationalized data A	Modular A
Executive and transaction IS, in-house development	PCs with standard software applications, mobile phones, servers	Enterprise-wide systems (ERP, e-business suites, and others from different vendors)	SOA program with different projects (EA, SOA bus, pilot services, shared components)

oriented. Most NTA workers use thin clients (e.g., Citrix), while employees in ITSD mainly use laptops.

In the following, we utilize the domains of the BEA model and underlying critical issues identified in the literature, to analyze the SOA implementation status of NTA (Table 42.4).

NTA has gradually implemented their EA, and there are incremental activities going on in all six domains of the BEA model. The most important issue has been to prepare the organization and to establish the OD programs to transform toward a service-oriented enterprise. Lately, the focus has been on the domains of BB and organization and governance. In the following, we discuss the most critical issues and challenges identified in this study.

6 Discussion and Implications

Primarily, in this study we identified five issues, which were particularly critical when implementing SOA: (1) creating a new organizational structure, (2) creating a cross-silo mindset, (3) creating a common pool of human and technical resources, (4) introducing SOA governance principles that create a collective and integrated approach for SD, and finally, (5) treating SOA implementation as a technochange project. All issues are interdependent, and the latter incorporates the four other issues. In particular, these issues would require changes in the SD culture and practice of the organization. We discuss these issues below and generate a set of propositions.

To move toward a service-oriented organization, NTA needed to implement organizational and cultural changes, and they altered toward a matrix-based organization. In addition, project organizing became the primary approach to

Table 42.4 SOA implementation activities and status in NTA according to the BEA domain model

Domain 1. Business strategy and process
Business strategy and IT strategy: In 2008–2009, the IT strategy was renewed, focus on component-based development and SOA as a means to obtain integrated system architecture. A three-phase business strategy project started at the same time, focus on building a new enterprise strategy, collaboration issues, services, legitimacy, competence building, and regulations.
Establishing a SOA program: In spring 2009, ITSD and the Director of Taxes in NTA started the IT architecture project aiming to establish an overarching EA and an IT regulation plan. The regulation plan comprises the future goal of NTA's system portfolio
Domain 2. Architecture
NTA has a complex IT architecture with an IT portfolio consisting of application silos, which are at different maturity stages (Tables 42.2 and 42.3). The long-term objective is to obtain integration across technical and organizational silos. ITSD has developed a strategic image/representation of the EA goal and established a roadmap for parts of the system portfolio.
The knowledge and competency about EA and SOA vary across NTA, and they have activated competence programs to increase the competency
Domain 3. Building blocks (BB)
Reusable organizational BB. NTA has activities related to development of SOA governance principles and preparation of a handbook that defines design and development of common services. The principles concern testing, security issues, adoption of standards, and common architecture terms. NTA has a system portfolio consisting of business critical applications (e.g., systems for VAT) which also require a high level of security. NTA has dedicated resources to develop an organizational infrastructure of security to prepare for SOA.
Competencies are important organizational BB. The aim is to build a competence center which constitutes an own section in the new organizational structure. The competence center is responsible for building and maintaining important expertise for developing new services. The consultants belonging to this section are supposed to work at different projects initiated across departments, and the center provides a common expertise pool of human resources.
System developers have started to build competencies in JAVA programming to gain JAVA certification. They learn how to use the new development platform and related tools and rules for how to develop services according to best practices.
Reusable software BB: There are several projects going on (see domain 4). Because of security concerns, the development of services and connection to the SOA bus takes time. One project is going to deliver two shared components, and connection of core systems have just started. There are ongoing activities for establishing common services on the integration platform (SOA bus). In addition to develop reusable services, shared repositories of code are developed (e.g., framework for error corrections, logging, and access control)
Domain 4. Projects and applications
In the period from 2008 until now, there have been several projects focusing on IT architecture and renewing of the IT function in ITSD. Examples of deliveries from these projects are a road map for the future IT architecture in NTA, a SOA governance regime (new roles, processes), and an integration platform (the SOA bus) with a set of services at the bus. Other deliveries are a SD platform including a set of "best practices" for SD on a JAVA platform, in addition to training material for project managers, system architects, system developers, and test engineers using this platform. There are also several SD projects that develop shared services and frameworks (e.g., replacing existing COBOL applications)
Domain 5. Organization and governance
An incremental implementation of a new organizational model (matrix structure) is going on. This includes activation of a change management program. Focus is also on developing shared

(continued)

Table 42.4 (continued)

components and breaking down organizational silos. This requires changes in processes, collaboration structures, coordination, and establishment of SOA governance.
NTA gradually reduces silo organizing and moves toward cross-silo and project-based organizing. Projects are organized across application and organizational silos to obtain multidisciplinary teams. Experts from a competence center will attend projects initiated from different silos.
ITSD defines SOA governance which demands changes in IT culture; e.g., development of common and reusable services should replace ad hoc integration practices
Domain 6. Costs and benefits
NTA has a strong business case and good reasons for organizational transformation based upon SOA implementation. The aim is to break down the technical and organizational silo structures. Balance Scorecard is in use. NTA is working on establishing metrics and values

accomplish SD tasks. In addition, the objective was to set up multidisciplinary teams to ensure collaboration among participants across different silos. We propose:

P1: SOA adopters need to think differently about organizational structure and should become more project oriented and utilize cross-disciplinary teams.

The system portfolio at NTA consists of application silos. The organizing of SD activities is also silo based since groups of system developers are responsible for a particular silo. For instance, some system developers are experts on COBOL programming and mainframe, others are working with Oracle platforms, and others again have JAVA competencies. This structure has existed in the organization for years and has become a part of the SD culture. Generally, collaboration across silos was not necessary because development, operational control, and maintenance belonged to each silo. We propose that:

P2: SOA adopters need to establish a SD culture that emphasizes a cross-silo mindset, which obliterates organizational and technical silo structures.

The aim in ITSD is to provide a common pool of human resources available for different kinds of projects to encourage cross-silo work. This reorganization will cause changes in work tasks, responsibilities, and roles for the system developers. In addition, system developers need to build new competencies to be able to contribute in this new "cross-silo" structure, and a competence center for experts will involve several of the system developers. Over time, the plan is to phase out COBOL and build software components by using JAVA. ITSD has established several training programs to build JAVA competencies. System developers need to learn a new language code, which might be a steep learning curve. Some of the programmers have worked in NTA for 40 years and have no JAVA experience. However, these programmers do have a unique competence in COBOL, Oracle, or mainframe; these competencies are getting more and more seldom and are important to maintain when people retire. Phasing out mainframes and COBOL are supposed to take several years. We propose that:

P3a: SOA adopters should create a common pool of technical and human competency resources to provide services based upon reusability and flexibility.

P3b: SOA adopters need to maintain competencies and expertise on legacy systems.

SOA governance principles require changes in the SD culture and practice. NTA has an overarching framework of common SD guidelines. However, traditionally in NTA, these guidelines are interpreted and used differently in practice across application silos. Often, SD activities have lacked coordination and documentation and have been on an ad hoc basis. This has resulted in certain path dependencies for each silo and inflexible solutions. Additionally, integration of applications and changes become difficult and costly. System developers in NTA need to accept a different SD culture and practice when implementing SOA. The SD culture based upon SOA governance principles requires a cross-silo and community mindset, which would lead to increased control and less freedom for the system developers inside each silo. The intention is to implement development practices that are best for the SOA community as a whole and provide a common pool of both human and technical resources through services. Thus, we propose:

P4: SOA adopters need to build governance principles that encourage a collective approach rather than a silo approach for the system developers.

According to the BEA domain model and previous research reporting on enterprise-wide IS implementations, we characterize SOA implementation as a technochange project (Markus 2004). Firstly, implementation of SOA in this study forces changes in organizational structure and governance principles for SD, and changes in SD culture and practices become inevitable. Secondly, a SOA project cannot be treated as either an OD project or as an IS implementation project only. In this study, NTA used its IT strategy (the SOA program) to drive an array of organizational and social changes. These changes comprised creating a new organizational structure, which should ideally break down the silos, and new SD practices and governance principles to support the obliterating of silos. According to Markus (2004), effective technochange management is a program of change initiatives of which an IT project is one; others may include organizational and business process restructuring, job redesign, and training programs (ibid., p. 5). In this study, NTA established a SOA program that comprised a combination of organizational (e.g., matrix structure) and technical (e.g., platform project) projects. The SOA program in NTA combines and prioritizes OD programs and technical SOA projects. Moreover, NTA has established training programs to build required competencies and skills. Finally, we propose:

P5: A Technochange approach is a suitable approach to tackle the complexity of a SOA implementation project.

7 Conclusion

This study has highlighted several critical issues for SOA adopters. Findings demonstrate that organizations with heterogeneous IT portfolios may achieve great advantages by implementing SOA. However, SOA implementation is also a complex and risky enterprise-wide initiative that requires a technochange management approach. Thus, SOA implementation projects are long-term projects without "quick wins." Benefits became visible when the organization has accomplished a combination of organizational and cultural change programs in addition to technical SOA projects to establish a service-oriented enterprise. In this study, the SOA program is an ongoing initiative. Because of business critical silo applications, the company cannot make radical changes, but needs to follow an incremental and long-term approach. In addition, to build new competencies on JAVA, the organization also see the challenge of keeping and maintaining scarce competencies such as COBOL programming skills and knowledge about mainframes. Thus, to break down silos by creating organizational and technical BB and creating a cross-silo mindset among system developers are important. Findings indicate that SOA adopters need to develop new governance principles that avoid ad hoc and silo-based SD practices and encourage a collective approach beneficially for the organization as a whole.

The study has implications for practice. Lessons learned from this study could be transferred to other companies and support them in a challenging SOA implementation process. Our research is exploratory and as such has limitations. It will, however, serve as input for subsequent qualitative studies of SOA implementation in organizations. It would also be interesting to see if findings are generalizable to other organizations.

References

Antikainen J, Pekkola S (2009) Factors influencing the alignment of SOA development with business objectives. In: Newell S, et al (eds) Proceedings of the 17th European conference on information systems (ECIS 2009), Verona, Italy, pp 2579–2590

Baskerville RL, Cavallari M, Hjort-Madsen K, Pries-Heje J, Sorrentino M, Virili F (2010) The strategic value of SOA: a comparative case study in the banking sector. Int J Inf Technol Manage 9(1):30–53

BEA Systems (2005) BEA SOA domain model. Ebiz – the insider's guide to business and IT agility. Available from: http://www.ebizq.net/white_papers/6196.html. Cited 30 Jan 2011

BEA Systems (2008) BEA's SOA reference architecture: a foundation for business agility. Available from: http://www.zdnetasia.com/whitepaper/bea-s-soa-reference-architecture-a-foundation-for-business-agility_wp-361122.htm. Cited 20 Mar 2011

Beimborn D, Joachim N, Muenstermann B (2009) Impact of service-oriented architectures (SOA) on business process standardization – proposing a research model. In: Newell S, et al (eds) Proceedings of the 17th European conference on information systems (ECIS 2009), Verona, Italy, pp 2675–2686

Hirschheim R, Welke R, Schwarz A (2010) Service – oriented architecture: myths, realities, and a maturity model. MIS Q Exec 9(1):37–48

IBM Corporation by Chen, Y (2006) Service oriented architecture. Available from: http://sei.pku.edu.cn/~huanggang/ibmcourse/2006/SOA.pdf. Cited 20 Apr 2009

Klein HK, Myers MD (1999) A Set of principles for conducting and evaluating interpretive field studies in information systems. MIS Q 23(1):67–93

Lam W (2005) Investigating success factors in enterprise application integration: a case-driven analysis. Eur J Inf Syst 14(2):175–187

Lawler JP, Howell-barber H (2008) Service-oriented architecture: SOA strategy, methodology, and technology. Auerbach Publications, Boca Raton

Lee J, Siau K, Hong S (2003) Enterprise integration with ERP and EAI. Commun ACM 46 (2):54–60

Legner C, Heutschi R (2007) SOA adoption in practice – findings from early SOA imp. In: Österle H, Schelp J, Winter R (eds) Proceedings of the fifteenth European conference on information systems (ECIS 2007), St. Gallen, pp 1643–1654

Markus ML (2004) Technochange management: using IT to drive organizational change. J Inf Technol 19(1):4–20

Niemann M, Eckert J, Repp N, Steinmetz R (2008) Towards a generic Governance model for service-oriented architectures. In: Proceedings of the fourteenth Americas conference of information systems, Toronto, Canada

Ordanini A, Pasini P (2008) Service co-production and value co-creation: the case for a service-oriented architecture (SOA). Eur Manage J 26(5):289–297

Reason P, Bradbury H (2001) Handbook of action research: participative inquiry and practice. Sage, London, XLII, 468 p

Robey D, Ross JW, Boudreau M-C (2002) Learning to implement enterprise systems: an exploratory study of the dialectics of change. J Manage Inf Syst 19(1):17–46

Ross JW (2003) Creating a strategic architecture competency: learning in stages. MIS Q Exec 2 (1):31–43

Schepers TGJ, Iacob ME, Eck PATV (2008) A lifecycle approach to SOA governance. In: Proceedings of the 2008 ACM symposium on applied computing, Fortaleza, Ceara, Brazil. ACM, New York, pp 1055–1061

Susman GI, Evered RD (1978) An assessment of the scientific merits of action research. Adm Sci Q 23(4):582–603

Venkatraman N, Henderson JC, Oldach S (1993) Continuous strategic alignment: exploiting information technology capabilities for competitive success. Eur Manage J 11(2):139–149

Viering G, Legner C, Ahlemann F (2009) The (lacking) business perspective on SOA–critical themes in SOA research. In: Wirtschaftinformatik proceedings. Paper 5

Chapter 43
Knowledge-Based Information Systems: A Wind Farm Case Study

Bindi Chen, William Wei Song, Yanhui Feng, Yingning Qiu, and Peter Tavner

1 Introduction

Information system development (ISD) is used to enhance business processes through information technologies (IT). According to (Xia and Lee 2005), ISD usually involves the analysis, design and implementation of IT systems and applications to support business functions. These steps are all typically knowledge-intensive and require diverse expertise. In addition, knowledge management (KM) becomes a factor for success in today's economy, where knowledge is fundamental for delivering benefit. Global organisations are developing KM projects to harvest knowledge from their processes to remain innovative and competitive (Abdullah et al. 2006). Usually, there is a great deal of information within organisations but a lack of knowledge (Grundspenkis 2006). Information and knowledge assets are intellectual capital, which can provide commercial potential for organisations if well utilised (Apshvalka and Grundspenkis 2005). Therefore, current research on ISD emphasises the integration of KM, and it can be seen as the intersection between KM and ISD, as shown in Fig. 43.1.

The concept of knowledge integration in ISD was first proposed by Grant (1996). After that, it remained a theoretical development with little attention paid to practice. In 2003 Benko and McFarlan introduced real options theory to represent an instance of successful knowledge integration. A relevant paper in 2005 was (Fichman et al. 2005) based on rich secondary analyses of multi-case data. In 2006, (Tiwana et al. 2006) was one of the earliest management information system studies using conjoint research design and one of the first empirical tests of real options theory.

In addition, the Enterprise Knowledge Development–Change Management Method (EKD–CMM) was introduced in Nurcan and Colette (2003) to provide a

B. Chen (✉) • W.W. Song • Y. Feng • Y. Qiu • P. Tavner
School of Engineering and Computing Science, Durham University, Durham DH1 3LE, UK
e-mail: bindi.chen@dur.ac.uk

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_43,

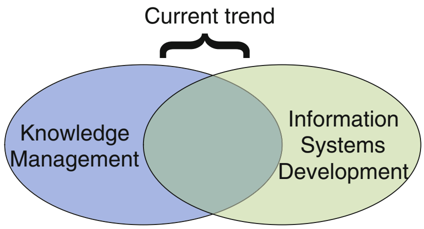


Fig. 43.1 Intersection of KM and ISD

multiple and dynamically constructed way of organising and guiding enterprise knowledge modelling and the organisational change processes. The method was built on the notion of labelled graph intentions and strategies called a map and associated guidelines. In 2006, an experimental study was conducted (Meso et al. 2006) to determine whether appropriately matching a methodology to a business application developed more effective knowledge-based processes among team members. The results suggested that ISD should include knowledge or cognitive elements.

The goal of integrating KM into ISD is to provide knowledge creation, storage, use/reuse and transfer to support an organisation's problem-solving and decision-making. This can also help to change an organisation's management strategy to be more competitive in a rapidly changing environment. There are technical difficulties involved in the development of KM and ISD integration, for example, data processing, knowledge discovery and knowledge utilisation, but these can be resolved using IT and artificial intelligence (AI) techniques.

This chapter attempts to bridge the gap between KM and ISD by a step-by-step approach to analyse information to create knowledge. This chapter then applies that approach to an industrial case study from wind industry. Firstly, a method of organisation using intelligent agents is proposed. Secondly, this chapter considers how KM can be integrated with the information system using intelligent agents and multi-agents. Finally, this chapter describes a model of a knowledge-based information system (KIS) that could be a structure to support the management of a real wind farm (WF), as well as supporting enterprise-wide knowledge, experience and decision-making.

The goal of this research is to develop a prototype of KIS model and to implement it into a working WF information system to evaluate and validate the proposed framework.

2 Organisation as Intelligent Agent

Agents are a valuable tool in ISD and KM. A widely accepted definition of an agent is defined as follows:

> *An agent is an entity that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors* (Russell and Norvig 2003).

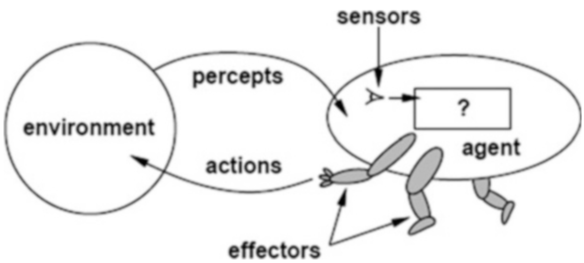


Fig. 43.2 Agents interact with environment through sensors and effectors (Russell and Norvig 2003)

Agents can be natural or artificial. For example, a human agent has eyes and ears for sensors and hands and legs for effectors; a robotic agent could have camera for a sensor and wheels for effectors; a software agent has data as its percept and actions. All agents are knowledge workers, and their decisions respond to their environment. A generic agent is shown in Fig. 43.2.

From the systems theory point of view, any organisation can be considered as a set of objects together with relationships between them (Grundspenkis and Kirikova 2005). Some organisations try to develop information systems to harvest knowledge and remain innovative and competitive. As new situations appear, knowledge is gained, new technologies are introduced and new solutions are developed. There is a trend for ISD to integrate such knowledge with the goal of making rational decisions and taking the best possible action. Thus, the interpretation of an organisation as a whole, using the intelligent agent concept, is obvious.

In computer science, an intelligent agent is an autonomous entity which observes and acts upon an environment and directs its activity towards achieving goals using learning or knowledge (Russell and Norvig 2003). An intelligent organisation can be considered as a combination of intelligent agents perceiving the current state of the environment, using its sensors for knowledge acquisition. Knowledge about the current state of an organisation and its goal state can be used to determine actions, which can be applied, through effectors, to the organisation's environment. This output is determined on the basis of percept and knowledge captured into the intelligent agents' knowledge base. An example of the interpretation of an organisation in terms of intelligent agents is shown in Fig. 43.3.

From an intelligent agent's point of view, the KIS means knowledge acquisition, processing and use for rational decision-making, choosing the best action and generating new knowledge. The main functions of a KIS can be characterised as follows:

- Detection of information and knowledge (sensors)
- Storage of detected information and knowledge (knowledge base)
- Decision-making (inference engine)
- Retrieval, visualisation of knowledge and actions (effectors)

An organisation's business processes are supported by intelligent agent activities, such as decision-making and action. Intelligent agents should provide possible solutions and alternatives, theoretically satisfactory to support the organisation's business processes.

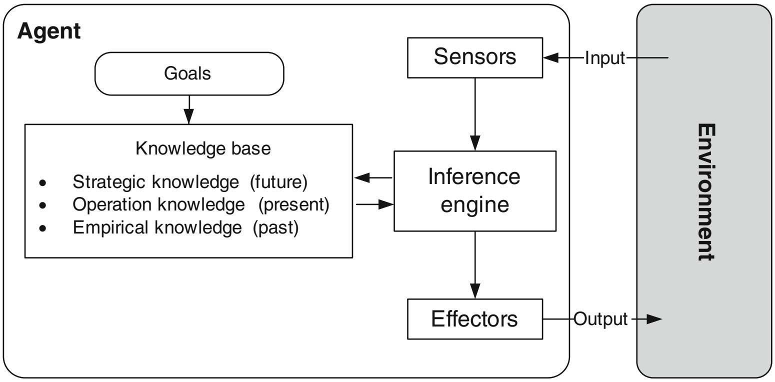


Fig. 43.3 An organisation as an intelligent agent

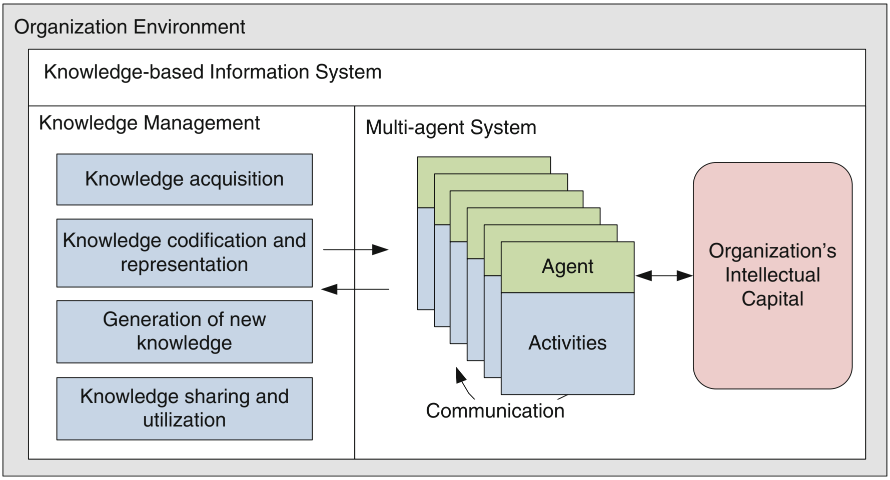


Fig. 43.4 KIS framework

3 Proposed Model Using Multi-agents

More detailed examination of the business processes inside an organisation reveals that there will be many different people and subsystems doing diverse tasks required by the organisation. Therefore, a multi-agent system (MAS), which consists of multiple interacting intelligent agents, will be needed to solve problems that are too difficult or impossible for an individual agent. A proposed KIS model which combines KM and MAS is shown in Fig. 43.4.

As shown in Fig. 43.4, the conceptual framework of a KIS consists of two main parts: a KM and a MAS for business process support. The KMS is an integrated set of technologies, hardware and software, to provide knowledge acquisition, storage, processing, retrieval and representation. The KM is usually supported by AI techniques. The purpose of adding KM into the KIS is to identify intellectual capital of the organisation and organise that knowledge to make it easily accessible and applicable. The MAS is a physical or virtual environment where intelligent agents of the organisation may communicate with each other for effective knowledge sharing and distribution to achieve the business process goals.

4 Case Study

In order to study these KM–ISD integration issues, it would be desirable to apply the proposed approach to a real-world scenario. The following section applies the approach to a large wind farm (WF) to demonstrate the concept of the proposed KIS.

4.1 Large Wind Farms

The wind industry is currently the fastest growing renewable electrical energy source around the world (The Economics of Wind Energy 2009; Tavner 2008). A large WF will contain 100–500 wind turbines (WTs). Each of them is an autonomous, robotic electric generator of 2–5 MW output connected to an electrical power system. Revenue is earned from the WF for the MWh of electrical energy generated, and current revenue for large WFs is about £60/MWh generated. The WF will have a complex signal conditioning and data acquisition (SCADA) system to control this renewable power station which could generate 1.4–35 GWh per day equivalent to £84k–2.16M revenue per day (Feng et al. 2010).

Each WT has 100–400 input/output (I/O) signals and alarms, which are monitored every 10 min providing the necessary monitoring and control environment to manage the power station. Therefore, a WF of 100–500 such WTs could produce 10,000–200,000 items of signal and alarm data every 10 min.

With the increase of wind energy development, especially the increase in the number of offshore WFs, where maintenance access is difficult, the operation and maintenance (O&M) of these huge autonomous, robotic structures is gaining financial importance, as any downtime due to faults will lead to a loss of revenue. This heightens the need for a comprehensive KIS that can use existing WT monitoring systems to provide advanced O&M management services.

4.2 Basic Concepts

This section provides a formal description of the WF.

Definition 1 (WF): A WF is defined as follows:

$$F = \{T, f_1, f_2, \ldots f_n\}$$

where F is a WF containing a collection of WT T with n different attributes f_1, $1 \leqslant i \leqslant n$, for example, location, number of WTs and maximum power output.

Definition 2 (WT): A WF can have many WTs; thus, T is defined as follows:

$$T = \{t_1, t_2, \ldots t_n\}$$

where t_i, $1 \leqslant i \leqslant n$ denotes the characteristics of individual WTs; see below.

Definition 3 (Subsystem): Each WT has one or more subsystems:

$$t_i = \{s_1, s_2, \ldots s_n\}$$

where s_i, $1 \leqslant i \leqslant n$ denotes individual WT subsystem, for example, a pitch system, gearbox or generator.

Definition 4 (Component): Each subsystem contains one or more components:

$$s_i = \{c_1, c_2, \ldots c_n\}$$

where c_i, $1 \leqslant i \leqslant n$ denotes individual components of a subsystem, for example, for the pitch system, an encoder, a blade or the rotor hub.

Definition 5 (Component attributes): Each component has some attributes:

$$c_i = \{ca_1, ca_2, \ldots ca_n\}$$

where ca_i, $1 \leqslant i \leqslant n$ denotes the component attributes, for example, the value of the blade angle.

The key objective for developing a KIS for a WF is to offer a comprehensive solution that exceeds the operational and management potential of existing conventional monitoring systems, such as SCADA or condition monitoring systems (CMS), with the aim of reducing O&M costs and raising WF generation revenue by reducing downtime. The main functions of a KIS for a WF would include:

- *Monitoring*: Use existing WT monitoring systems, SCADA and CMS, to display current WT and WF performance.
- *Problem-solving*: Assist experts to diagnose WT or WF failures.
- *Decision-making*: Using advanced AI to detect incipient WT or WF faults early.
- *Planning and prediction*: Maintenance to be scheduled based on the early detection of incipient WT or WF faults.

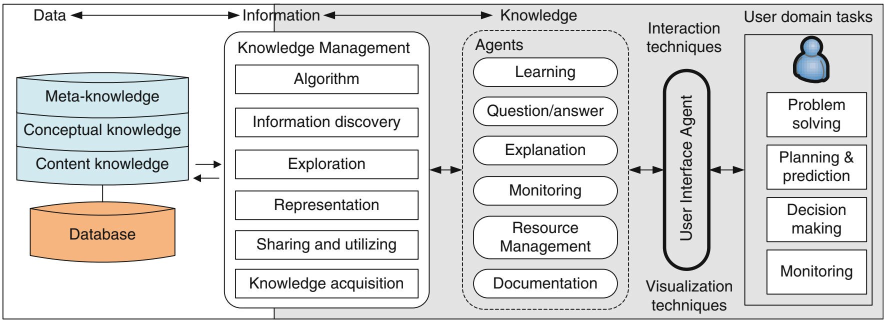


Fig. 43.5 Model for the WF KIS system architecture

4.3 Exploring the WF Case Study

To achieve the "knowledge-based" KIS and meet these objectives, a collaborative structure is proposed that connects to the WF database. A prototype model of the WF KIS system architecture is show in Fig. 43.5.

The system architecture includes four main components, from the left to right of Fig. 43.5: resources, KM, agents and a user interface. Each of these components corresponds to a logical step in KIS implementation.

- Resources consist of a primary database and knowledge libraries. In this component, all data are transformed and saved in the same format to make the data easily accessible and improve efficiency.
- KM consists of algorithm, information discovery, exploration, representation, sharing and utilisation functions adapted to extract knowledge and solve problems. These will be supported by AI techniques, such as artificial neural networks (ANN), pattern recognition and generic algorithms.
- Agents will provide activities such as learning, questions and answers, explanations, information retrieval, resource management and documentation.
- Finally, the user interface is an agent consisting of tools for WF operators and maintainers for visualising the data structure and displaying results. The use of data visualisation techniques will be emphasised because it will bring additional exploratory data analysis techniques appropriate to complex WF data. The data visualisation tools will provide for the large dimensional datasets the following: data clustering and projection for feature extraction, visualisation and interpretation.

For the WF operator and maintainer, the main goal of the KIS will be the acquisition of knowledge, for monitoring, decision-making, problem-solving, planning and prediction, with the aim of reducing O&M costs and raising WF generation revenue by reducing downtime.

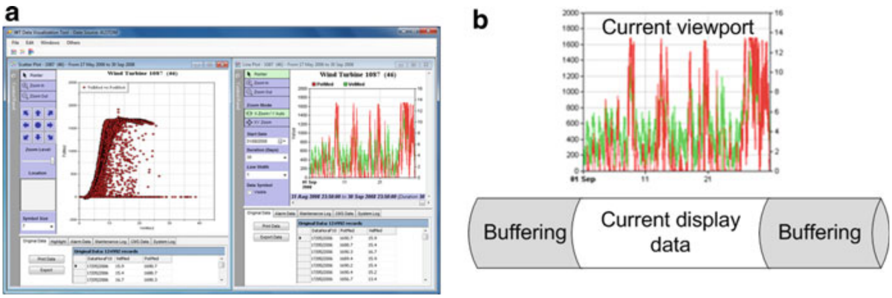


Fig. 43.6 (**a**) Screenshot of the system implementation work on WF SCADA data; (**b**) Buffering technique

5 Implementation

The research is a part of a larger activity to improve offshore WF availability and involves accessing 2 terabytes of real WT data. In order to process such large amounts of data, a dedicated server consisting of 2 processors, 48 gigabytes of memory and 8-terabyte hard drives, was constructed in School of Engineering and Computing Sciences, Durham University. The initial research has developed a client/server-based prototype application in which the server-side is used to handle data and process client requests, while the client-side is on a user's PC, allowing users to request server content or services. The initial objectives of this research focus on the following two problems:

- *Massive data processing*: To handle 2 TB real WT data and also feature with information extraction to resolve data noise and randomness.
- *Knowledge representation and support decision-making*: Use the human expertise knowledge base for problem-solving and decision-making.

5.1 Results

At the current stage of the implementation, the prototype KIS can handle large data size with fast visualisation and interaction. For a typical WF binary data set, it can display up to 300 MB of data at a time without significant performance problem. An example of it in operation is shown in Fig. 43.6a. The prototype KIS also includes "buffering", "dynamic display" and "data aggregation".

- *Buffering*: Apart from loading the data into the viewport, the client-side temporarily holds adjacent viewport data to reduce data loading time and enhance user interactions, for example, for zooming in/out or for scrolling through moving data. This is shown in Fig. 43.6b.
- *Dynamic display*: If the user viewport resolution is less than the requested data size, it is not necessary to display all data. Requested data can be down-sampled,

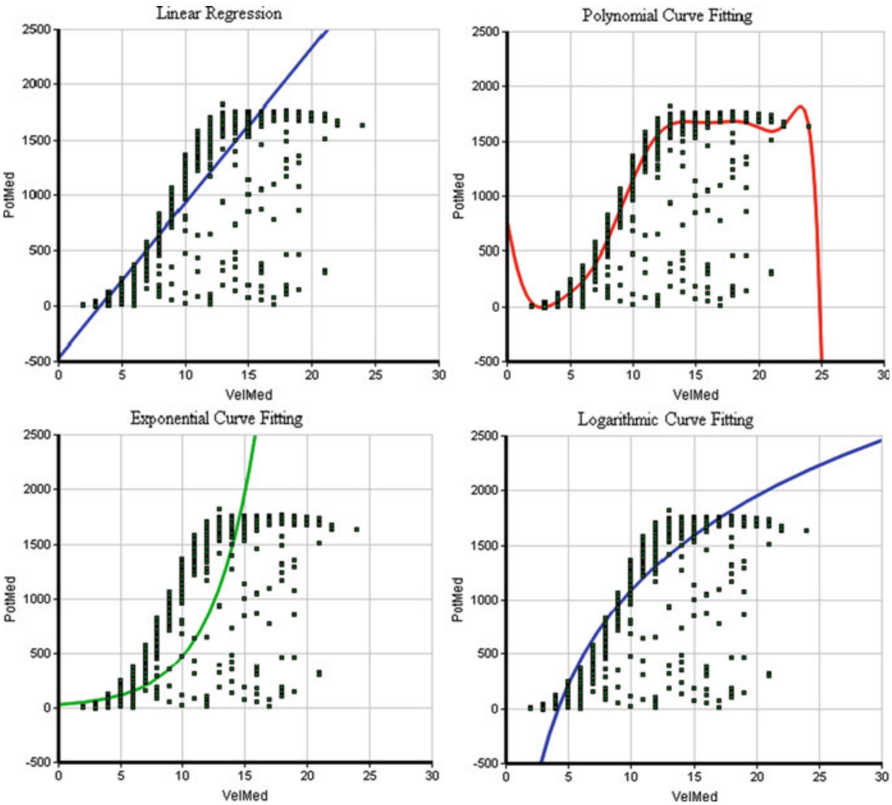


Fig. 43.7 Regression analysis for WT power curve fitting from SCADA data

using, for example, an average calculation, on server-side before being transferred to client-side.

- *Data aggregation*: The KIS provides statistical data aggregation techniques, such as average, ignore betweens, first or last of data, for the user to down-sample and enhance data visualisation.

The initial research investigated the following techniques:

- Regression analysis for data processing
- Artificial neural network (ANN) solutions for decision-making
- Self-organising feature mapping (SOM) for data clustering

Figure 43.7 shows regression analysis being applied for WT power curve fitting, which is proving useful for processing and detecting WT failures (Harman and Raftery 2003; Feng et al. 2011). Note that in Fig. 43.6a, polynomial curve fitting was used over a limited range to obtain a good WT power curve. Figure 43.8 shows an ANN being used to analyse SCADA alarms. This analysis is done on a WT with

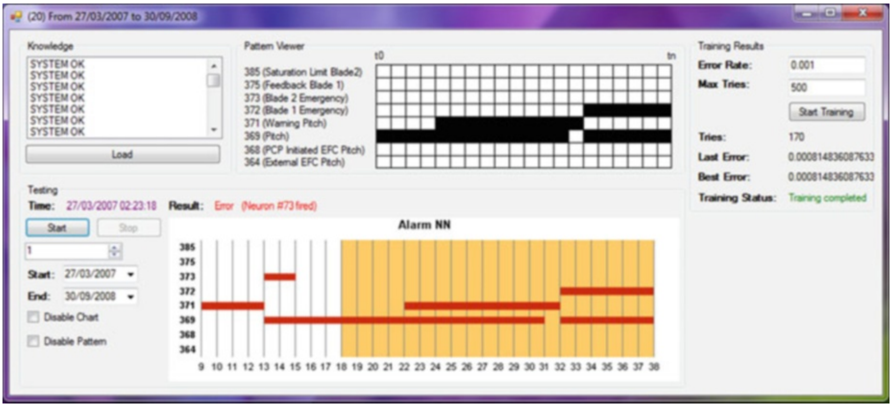


Fig. 43.8 ANN for SCADA alarm analysis for WT pitch mechanism fault detection

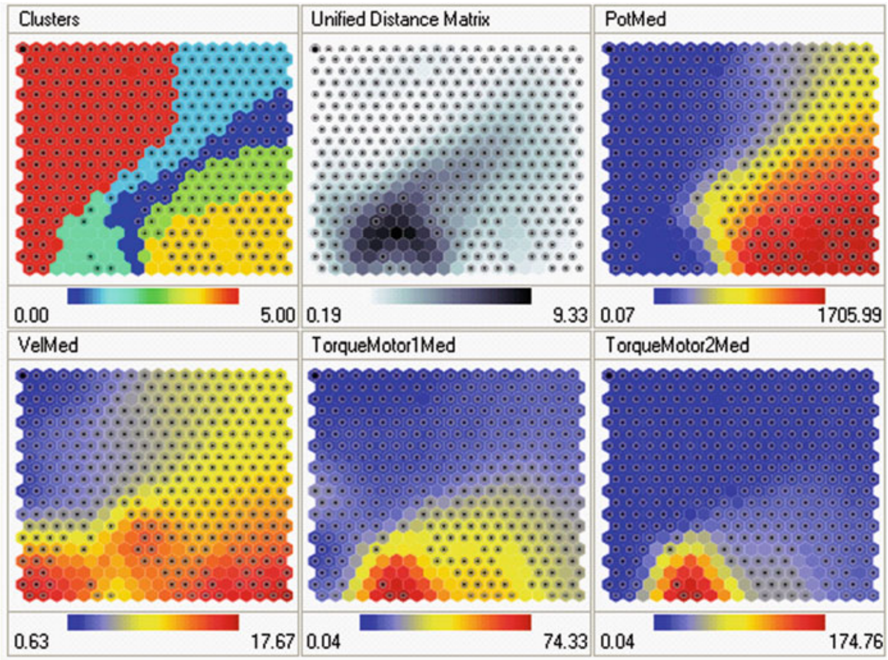


Fig. 43.9 SOM SCADA data clustering for WT pitch mechanism fault detection

known pitch failure to prove the benefit and the challenges of such an approach. Figure 43.9 shows a self-organising map (SOM) being used to cluster SCADA data from the pitch mechanism of a WT in order to detect pitch faults without expert supervision.

6 Conclusion

This chapter has identified a role for KM and MAS for the implementation of a large WF knowledge-based ISD. Several conceptual models for the configuration of the proposed KIS are discussed. Firstly, the organisation as a whole is interpreted as an intelligent agent, using its own knowledge base for knowledge capture. Secondly, how MASs may support an organisation's KM is discussed. Finally, a case study for the proposed KIS system architecture for a large WF is presented.

Considerable future works are needed to achieve results in implementing the proposed KIS for large WFs; the potential for using intelligent agents, MAS and KM to develop a more intelligent information system has been demonstrated in a number of real WT downtime scenarios. These will be presented in the full paper. We intend that the proposed KIS with MAS will serve as a platform to investigate the development of more intelligent information systems.

References

Abdullah MS, Kimble C, Benest I, Paige R (2006) Knowledge-based systems: a re-evaluation. J Knowl Manage 10(3):127

Apshvalka D, Grundspenkis J (2005) Personal knowledge management and intelligent agent perspective. In: Proceedings of the 14th international conference on information systems development pre-conference-ISD, Karlstad, Sweden, 14–17 August 2005

Benko C, McFarlan W (2003) Connecting the dots: aligning projects with objectives in unpredictable times. Harvard Business School Press, Boston

Feng Y, Tavner PJ, Long H (2010) Early experiences with UK round 1 offshore wind farms. Proc Inst Civil Eng Energy 163(4):167–181

Feng Y, Qiu Y, Crabtree C, Long H, Tavner PJ (2011) Use Of SCADA and CMS Signals for failure detection & Diagnosis of a wind turbine gearbox. In: EWEA, 2011, Brussels, Belgium

Fichman R, Keil M, Tiwana A (2005) Beyond valuation: real options thinking in IT project management. Calif Manage Rev 47(2):74–100

Grant R (1996) Prospering in dynamically-competitive environments: organizational capability as knowledge integration. Org Sci 7(4):375–387

Grundspenkis J (2006) Agent-based framework for modelling of organization and personal knowledge from knowledge management perspective. In: Knowledge enterprise: intelligent strategies in product design, manufacturing, and management, proceedings of PROLAMAT. Springer, Boston, pp 62–70

Grundspenkis J, Kirikova M (2005) Impact of the intelligent agent paradigm on knowledge management. In: Leondes CT (ed) Intelligent knowledge-based systems, vol 1, Knowledge-based systems. Kluwer, Boston

Harman KD, Raftery PG (2003) Analytical techniques for understanding the performance of operational wind farms. In: EWEC, 2003, Madrid, Spain

Meso P, Madey G, Troutt MD, Liegle J (2006) The knowledge management efficacy of matching information systems development methodologies with application characteristics—an experimental study. J Syst Softw 79(1):15–28

Nurcan S, Colette R (2003) A multi-method for defining the organizational change. Inf Softw Technol 45(2):61

Russell SJ, Norvig P (2003) Artificial intelligence: a modern approach, 2nd edn. Prentice Hall, Upper Saddle River, Chapter 2. ISBN 0-13-790395-2

Tavner PJ (2008) Wind power as a clean-energy contributor. Energy Policy 36(12):4397–4400

The Economics of Wind Energy (2009) A report by the European Wind Energy Association. Krohn S (ed). Poul-Erik Morthorst. Shimon Awerbuch

Tiwana A, Keil M, Fichman R (2006) IS project continuation in escalation situations: a real options model. Decis Sci J 37(3):357–391

Xia W, Lee G (2005) Complexity of information systems development projects: conceptualization and measurement development. J Manage Inf Syst 22(1):45–83

Chapter 44
The Power of the Crowd: Performing Usability Testing Using an On-Demand Workforce

C. Schneider and T. Cheung

1 Introduction

Web sites and online commerce sites, in particular, have become increasingly important for many business organizations. Such sites are used to attract new customers or to advertise and sell products or services. Given that a poorly designed Web site can have strong negative effects on visitors' transaction experience and perceptions of corporate image, quality Web page design is often regarded as a critical success factor for many organizations (Kim et al. 2003). An important component of a Web site's design is its usability (including factors such as its structure and navigational feedback), which can influence a person's use of a Web site and, in turn, purchasing behavior (Agarwal and Venkatesh 2002; Venkatesh and Agarwal 2006). Relatedly, Valacich et al. (2007) found that a site's functional convenience (determined by factors such as ease of use and ease of navigation) is an important consideration for utilitarian (e.g., online banking sites), hybrid (e.g., news or shopping sites), and hedonic (e.g., music, movie, or games sites) Web sites alike. Nevertheless, examples abound of poorly designed Web sites (see, e.g., www.websitesthatsuck.com, Flanders), and even Web sites of large companies suffer from usability problems.

One factor contributing to poorly designed Web sites is insufficient usability testing. Commonly, usability testing is seen as a lengthy endeavor, and even large corporations, including airlines, banks, and online shopping sites, appear to neglect usability testing. While Krug (2006) advocates that testing few users is better than testing none at all, thorough testing should involve a large number of diverse participants, so as to detect as many potential problems as possible (Spool and Schroeder 2001; Woolrych and Cockton 2001). However, increasing the number of

C. Schneider (✉) • T. Cheung
Department of Information Systems, City University of Hong Kong, Kowloon, Hong Kong SAR
e-mail: christoph.schneider@cityu.edu.hk; is.tc@cityu.edu.hk

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_44,

participants in usability studies leads to diminishing marginal returns, to the point where testing additional users is not cost-effective.

Our study attempts to address this problem and demonstrates how crowdsourcing (i.e., outsourcing of tasks to the crowd (Howe 2006)) can help to lower the costs of testing. Further, our study shows how the recording of users' mouse movements can provide additional insights into navigational problems of a Web site. In the following section, we will discuss Web site usability, followed by a discussion of the concept of crowdsourcing and how it can be used for usability studies. We will then present the methodology of our study, followed by a discussion of our results.

2 Web Site Usability

Usability has been defined in various ways; one widely used definition of usability is that proposed by ISO 9241-11, namely, "the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use" (see Karat 1997, p. 34). Krug (2006) succinctly summarized this definition of usability in the context of Web sites as "don't make me think," suggesting that a Web site should be self-evident, or at least, self-explanatory, requiring a visitor to spend as little thought as possible when interacting with the site. An important consideration, thus, is the navigation of a Web site, consisting of location, design, but also the naming of the links (see, e.g., Garrett 2011; Kalbach 2007; Wodtke and Govella 2009).

As people are trying to conserve resources (such as time), they are likely to quickly leave a Web site if they do not easily find what they are looking for (reflected in a high "bounce rate," or percentage of visitors leaving the site after visiting the first page). As these users are unlikely to come back, this can have a direct negative influence on a company's bottom line (Krug 2006). Hence, improving the site's navigation, so as to aid users in finding what they are looking for, is of paramount concern, especially for new Web sites. Established e-commerce sites such as Amazon.com (which, in 2011, drew more than two million visitors per day, according to www.stats.website.org) can afford to perform usability testing "on the fly," by presenting different versions of its Web sites to different customers and testing the ensuing effect on sales. Smaller companies, however, often do not have an established, loyal customer base, nor do they have the traffic needed to perform such tests. For such companies, it is thus important to minimize usability-related flaws before the site is launched. However, such companies also often face limited budgets and see themselves faced with difficulties in performing large-scale usability tests.

Whereas conducting usability studies with merely a few participants is far better than no usability testing at all (Krug 2006), with some arguing that as little as five participants are sufficient for usability testing (Nielsen), other research has shown that larger numbers of participants are needed to reliably catch usability problems

(Spool and Schroeder 2001; Woolrych and Cockton 2001). Especially with online shopping sites catering to diverse audiences with diverse backgrounds in experiences, education, and skills, actual users are likely to encounter various different usability problems. Hence, drawing on a limited pool of participants for usability testing is likely to lead to many problems not being discovered (Spool and Schroeder 2001), and organizations should attempt to perform larger scale test with participants reflecting their target audience. Crowdsourcing has recently become a popular means for drawing on a large number of diverse people to complete certain tasks (Kittur and Kraut 2008) and has the potential to overcome issues associated with recruiting participants for large-scale user testing.

3 Crowdsourcing

The term crowdsourcing, coined in 2006 by Jeff Howe (Howe 2006), refers to outsourcing of tasks not to other companies, but to the crowd—in other words, anyone who is willing to work on the task. Crowdsourcing thus creates a distributed labor network (Howe 2009) on which companies can draw on an as-needed basis. Zittrain (2008) compared crowdsourcing to distributed human computing; akin to flexibly obtaining computing resources from the cloud, companies can draw on a flexible, on-demand workforce, or human computing resources, through crowdsourcing.

A primary reason for the rise of crowdsourcing is the potential to harness the collective intelligence of the crowds. The notion of collective intelligence is based on the premise that people with diverse information and a diverse range of perspectives can outperform individual experts (Page 2007; Surowiecki 2004). Collective intelligence is an important mechanism that helps to create various informational products of high quality; a prime example is open-source software, which bases its quality largely on the contributions by a large, diverse set of participants (e.g., Raymond 2001). As a result, crowdsourcing has become popular among many organizations to perform a wide range of tasks, from very small to large-scale problems. For example, crowdsourcing provider InnoCentive helps companies such as pharmaceutical giant EliLilly find solutions to important problems; as anyone interested in working on these problems can contribute their solutions, InnoCentive in effect creates an on-demand R&D department. In addition to solving large problems, crowdsourcing is increasingly used for micro-tasks that can be performed by people without specific expertise, often within seconds or several minutes (Kittur and Kraut 2008).

3.1 Mechanical Turk

One popular micro-task market is Amazon's Mechanical Turk (MTurk, www.mturk.com). Using MTurk, "requesters" can post so-called "human intelligence tasks" (HITs) that "workers" (sometimes called "turkers") can complete. Typically, such HITs are tasks that are relatively easy to complete for humans, but very difficult for computers, such as tagging images, identifying whether Web sites include adult content, verifying a business' opening hours, or generating keywords for products; on a random day in April 2011, turkers could choose between more than 130,000 different HITs. For completing a HIT, a turker can typically earn US $0.01–US$1.00, depending on the time involved. Given the large number of turkers, tasks can often be completed within days, if not hours.

Recent research has shown that work completed by turkers can be of relatively high quality; for example, Kittur and Kraut (2008) have shown that turkers' ratings of the quality of articles in the online encyclopedia Wikipedia correlated strongly with Wikipedia administrators' ratings. Relatedly, researchers in psychology and other social sciences are increasingly using MTurk, particularly, as the diversity of the participant pool is larger than that of typical undergraduate college samples, and the data are as reliable as those collected using other methods, if not more (Buhrmester et al. 2011). Mason and Suri (2010) discuss a number of studies involving tasks such as public goods dilemmas and prisoners' dilemma games that found that the behavior of turkers closely resembled that of participants in traditional laboratory experiments. Further, Buhrmester and colleagues (2011) and Mason and Watts (2010) found that as long as compensation rates are realistic, data quality is not affected by the compensation rate; hence, high-quality data can be obtained rather inexpensively using micro-task markets such as MTurk. In the following section, we present the methodology of our usability study using MTurk.

4 Method

To conduct our study of assessing and improving Web site usability, we set up a mock online store and recruited participants through MTurk to search for specific products and provide feedback on the navigation of the online store. We conducted two rounds of the study, refining the navigation of the store after the first iteration. In each round, the participants were required to find three specific products belonging to three different categories and were asked to complete a brief questionnaire consisting of 11–13 questions related to the usability of the Web site. In addition, during each round, we captured the participants' mouse movements to obtain additional, independent indicators of navigational problems. Each participant received US$0.50 as compensation for their efforts. To further assess the efficacy of using crowdsourcing for user studies, we asked five domain experts to evaluate the online store using the same criteria.

4.1 Online Store Setup

We set up a mock online store using the open-source e-commerce solution PrestaShop v1.4 and created a number of product categories that were identical to

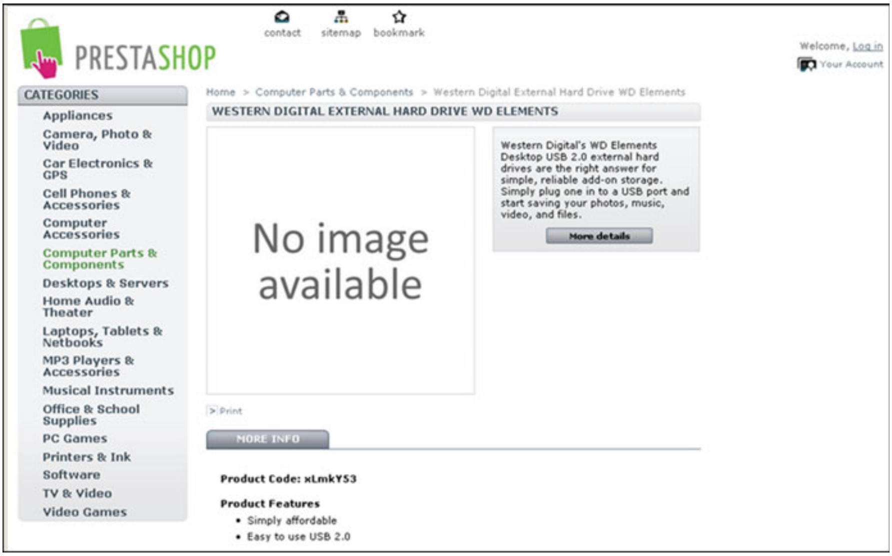


Fig. 44.1 Screenshot of online store

the subcategories used by Amazon.com in the "Electronics & Computers" category (e.g., TV & Video, Cell Phones & Accessories, and Computer Parts & Components). We created a sample product within each product category (resembling the products presented within the respective categories at Amazon.com, as shown in Fig. 44.1). After the initial round of testing, we modified the online store based on the feedback obtained from the study participants. In particular, we grouped several categories together, as well as added search functionality to the site.

4.2 Recruiting of Subjects

We recruited participants using MTurk by emailing 70 turkers (a subset of turkers who have worked on other, unrelated HITs for the first author). MTurk allows to filter turkers according to certain criteria, such as location, HIT approval rate (i.e., the percentage of HITs completed by a turker that met the respective requesters' expectations), or other criteria set by the requester. As many large Web sites cater to visitors from different regions, we purposely did not limit the turkers to those coming from a certain region.

After the initial round of testing, we notified each turker who participated in the first round via email[1] that an additional HIT was available. This allowed us to obtain repeated observations from the initial participants.

[1] Using a modified version of the "NotifyWorkers" script provided by Berinsky et al. (2010).

We also recruited five experts intimately familiar with Web site design and usability issues to evaluate both versions of the online store. These experts are instructors of introductory and/or advanced university courses on Web design or human-computer interaction.

4.3 Measures

We collected perceptual and objective measures from the participants. In particular, we drew on Loiacono et al.'s (2007) WebQual instrument to measure the factors most closely related to the objectives of our study (informational fit-to-task, ease of understanding, and intuitive operations; see Appendix A). Further, we included open-ended questions related to the major difficulties encountered when completing the task and solicited suggestions for improving the navigation of the Web site. Kittur and Kraut (2008) as well as Oppenheimer et al. (2009) stressed the importance of incorporating objective, verifiable questions, so as to increase the reliability of the data; hence, following their recommendations, we asked the respondents to find a product code (provided in each product's description) before answering the questions.

In addition to these measures, we sought to track the participants' actual behavior when visiting the online store, so as to identify items that may be indications of navigational problems. Specifically, it has been argued that a user's mouse movements can serve as a reliable indicator for the areas of attention (Atterer et al. 2006; Mueller and Lockerd 2001). Hence, we used ClickTale (www.clicktale.com) to record the participants' behavior in the online store.

5 Results and Discussion

We limited the participation in the study to 15 participants in the first round; the participants completed the evaluation within less than 48 h from the beginning of the study. Of these, 11 turkers (73.3 %) completed the second round of testing within 10 h of launching the second round (the remaining workers did not complete the second round). Considering that we intentionally limited our study to a small number of potential workers, these results suggest that large numbers of workers can be recruited with relative ease. Table 44.1 shows the means and standard deviations for the responses to the WebQual items. While some differences can be observed between Round 1 and Round 2, the low sample size prevented us from conducting statistical tests and limited our current analysis to qualitative observations.

In the initial round of testing, all turkers indicated problems in finding the right product categories (to various degrees); for example, one user stated, "I found the first and third with no problem because I knew which catagory *[sic]* to look under.

Table 44.1 Means and standard deviations for responses to WebQual instrument

	Round 1	Round 2
Informational fit-to-task	18.20(2.51)	14.82(4.34)
Ease of understanding	18.67(2.02)	18.54(3.00)
Intuitive operations	17.40(4.98)	18.09(2.96)

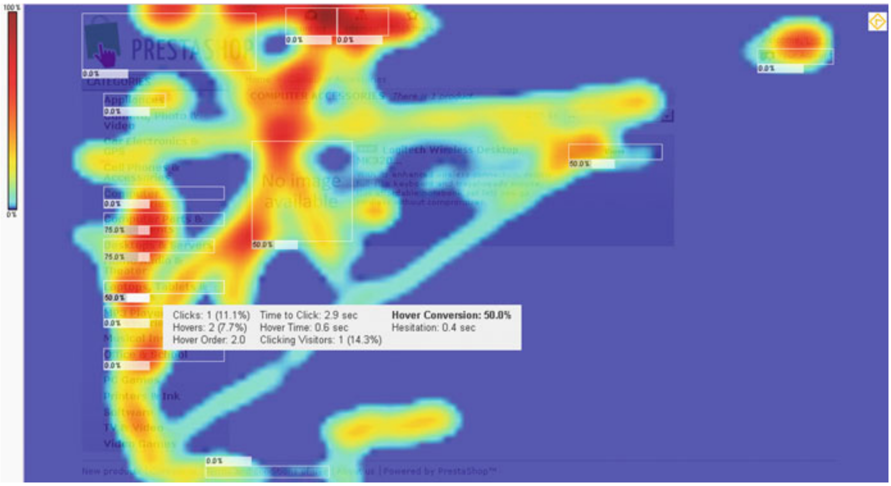


Fig. 44.2 Sample heatmap summarizing visitors' mouse movements

The middle product was more difficult. I looked at a few catagories *[sic]* that I thought it might be under, then just went down the list until I found it." This result is interesting, as the sample products were identical to those presented by Amazon.com in the respective categories. Hence, we conducted a visual inspection of the users' mouse movements (see Fig. 44.2) to gain further, qualitative insights into potential usability problems. This inspection also indicated some degree of confusion regarding the product categories, with users clicking on three different (but related) product categories in attempting to find the product; further, many users apparently attempted to consult the store's site map for guidance on where to find the product.

When asked about suggestions for improving the site, 80 % of the users indicated that adding a search function would be useful; other users suggested adding subcategories. In contrast, only 40 % of the experts focused on the search functionality, with the remaining experts focusing on less essential improvements, such as adding the number of products in each category or better separating the category items. Overall, it appears that, whereas the experts give very useful recommendations on how to improve the Web site, an analysis of the turkers' comments demonstrates that there are some other issues (such as the search functionality) that users care strongly about.

After adding search functionality and grouping categories, we conducted an additional round of testing. In this round, 8 of the 11 users found the site easier to use, with 2 being undecided; one user found the site more difficult to use, presumably as he/she did not immediately realize how to expand the product categories (by clicking on a plus symbol next to the parent category). This is also reflected in the lower mean for "intuitive operations" in Round 2. Although only one user reported this problem, this is an indicator of a usability problem that the designer should be aware of. In addition, several turkers provided recommendations on further improving the site's navigation, such as by adding filtering options. In general, the experts perceived the site to be easier to use. Interestingly, however, two of the experts highlighted that creating subcategories was actually detrimental to navigation, as it took longer to get to the actual product listing.

As any study, our current study is not without limitations. First and foremost, we limited the number of workers participating in our experiment, thus not enabling us to perform statistical tests related to perceptions of the site's usability. In addition, we limited participation to turkers who had already completed HITs for us; opening up the task to the general community of turkers would have allowed us to recruit even more testers more quickly; organizations using MTurk for usability testing should consider this trade-off between building and using an ad hoc workforce of workers *known* to perform good work versus using anyone. Whereas the first option may result in higher quality work, it is more effortful and limits the speed in which results can be obtained. In contrast, allowing any turker to work on the task helps to generate results more quickly.

6 Conclusion

For many of today's organizations, and organizations involved in e-commerce in particular, high-quality Web pages are considered a critical success factor (Kim et al. 2003). This is especially true for businesses in the early stages. These businesses often do not have an established user base and cannot afford to use actual customers as beta testers, yet also cannot afford to lose customers due to poor Web site design.

Our results have shown that MTurk can be a very useful tool for usability evaluations of Web sites. In particular, the current study serves as a proof of concept that MTurk can help to recruit participants extremely quickly and relatively effortlessly, allowing organizations to perform usability testing with a large number of users at relatively low cost. Whereas traditional ways of observing and recording users (such as using cameras) are not possible using the current approach, the recording of users' mouse movements can help to infer where people may have faced problems. In our current study, the participants focused on the most pressing aspects in the first round and only then noted other aspects of navigation; this is in contrast to the experts, who provided more detailed recommendations from the beginning. This suggests that using multiple rounds of testing should be an essential part of usability testing using crowdsourcing.

Acknowledgments The work described in this chapter was substantially supported by a research grant from City University of Hong Kong (Project No. 7200147).

Appendix A

Survey items used:

1. WebQual (Loiacono et al. 2007)

   Informational fit-to-task (INFO)

   1. The information on the Web site is pretty much what I need to carry out my tasks.
   2. The Web site adequately meets my information needs.
   3. The information on the Web site is effective.

   Ease of understanding (EUDSTD)

   1. The display pages within the Web site are easy to read.
   2. The text on the Web site is easy to read.
   3. The Web site labels are easy to understand.

   Intuitive operations (INTUIT)

   1. Learning to operate the Web site is easy for me.
   2. It would be easy for me to become skillful at using the Web site.
   3. I find the Web site easy to use.

2. Open ended questions:

   What were the major difficulties involved in finding the products?
   What are your suggestions to improve the navigation of the Web site?

Additional items for Round 2:
Compared to the prior version, products are easier to find.
(7-point scale anchored at Strongly agree – Strongly disagree)
Compared to the prior version, I find the site...
(7-point scale anchored at Much easier to use – Much more difficult to use)

References

Agarwal R, Venkatesh V (2002) Assessing a firm's web presence: a heuristic evaluation procedure for the measurement of usability. Inf Syst Res 13(2):168–186

Atterer R, Wnuk M, Schmidt A (2006) Knowing the user's every move: user activity tracking for website usability evaluation and implicit interaction. In: Proceedings of the 15th international conference on World Wide Web. ACM, New York, pp 203–212

Berinsky AJ, Huber GA, Lez GS (2010) Using mechanical turk as a subject recruitment tool for experimental research. Retrieved from http://huber.research.yale.edu/materials/26_paper.pdf
Buhrmester M, Kwang T, Gosling SD (2011) Amazon's mechanical turk. Perspect Psychol Sci 6 (1):3–5
Flanders V Web pages that suck – learn good web design by looking at bad web design. In: Web pages that suck. Retrieved 18 Apr 2011, from http://www.webpagesthatsuck.com/
Garrett JJ (2011) The elements of user experience: user-centered design for the web and beyond. New Riders, Berkeley
Howe J (2006) The rise of Crowdsourcing. Retrieved 18 Apr 2011, from http://www.wired.com/wired/archive/14.06/crowds.html
Howe J (2009) Crowdsourcing: why the power of the crowd is driving the future of business, 2nd edn. Crown Publishing Group, New York
Kalbach J (2007) Designing Web navigation. O'Reilly Media, Inc, Sebastopol
Karat J (1997) Evolving the scope of user-centered design. Commun ACM 40(7):33–38
Kim SE, Shaw T, Schneider H (2003) Web site design benchmarking within industry groups. Internet Res 13(1):17–26
Kittur A, Kraut RE (2008) Harnessing the wisdom of crowds in Wikipedia: quality through coordination. In: Proceedings of the 2008 international ACM conference on supporting group work, San Diego, CA. ACM, New York, pp 37–46
Krug S (2006) Don't make me think! A common sense approach to web usability, 2nd edn. New Riders, Berkeley
Loiacono ET, Watson RT, Goodhue DL (2007) WebQual: an instrument for consumer evaluation of web sites. Int J Electr Commer 11(3):51–87
Mason W, Suri S (2010) Conducting behavioral research on Amazon's mechanical turk. SSRN eLibrary. Retrieved 18 Feb 2011, from http://papers.ssrn.com/sol3/papers.cfm?abstract_id=1691163
Mason W, Watts DJ (2010) Financial incentives and the "performance of crowds". ACM SIGKDD Explor Newslett 11:100–108
Mueller F, Lockerd A (2001) Cheese: tracking mouse movement activity on websites, a tool for user modeling. In: CHI'01 extended abstracts on human factors in computing systems. ACM, New York, pp 279–280
Nielsen J Usability testing with 5 Users. Jakob Nielsen's Alertbox. Retrieved 18 Apr 2011, from http://www.useit.com/alertbox/20000319.html
Oppenheimer DM, Meyvis T, Davidenko N (2009) Instructional manipulation checks: detecting satisficing to increase statistical power. J Exp Soc Psychol 45(4):867–872
Page SE (2007) The difference: how the power of diversity creates better groups, firms, schools, and societies. Princeton University Press, Princeton
Raymond ES (2001) The cathedral and the bazaar: musings on Linux and open source by an accidental revolutionary. O'Reilly, Sebastopol
Spool J, Schroeder W (2001) Testing web sites: five users is nowhere near enough. In: CHI'01 extended abstracts on human factors in computing systems. ACM, New York, pp 285–286
Surowiecki J (2004) The wisdom of crowds. Doubleday, New York
Valacich JS, Parboteeah DV, Wells JD (2007) The online consumer's hierarchy of needs. Commun ACM 50:84–90
Venkatesh V, Agarwal R (2006) Turning visitors into customers: a usability-centric perspective on purchase behavior in e-channels. Manage Sci 52(3):367–382
Wodtke C, Govella A (2009) Information architecture: blueprints for the Web. New Riders, Berkeley
Woolrych A, Cockton G (2001) Why and when five test users aren't enough. In: Proceedings of the IHM-HCI 2001 conference, vol 2. Cépadèus Éditions, Toulouse, pp 105–108
Zittrain J (2008) Ubiquitous human computing. Philos Trans R Soc A Math Phy Eng Sci 366 (1881):3813–3821

Chapter 45
Using UML 2.0 for Modelling Software Processes at Siemens AG

Stefan Dietrich, Peter Killisperger, Thomas Stückl, Norbert Weber, Tomas Hartmann, and Eva-Maria Kern

1 Introduction

Traditional process-driven software development approaches which are often large and complex are widely used in industry. Examples are software development within Siemens AG (Schmelzer and Sesselmann 2004) and Motorola Inc. (Fitzgerald et al. 2003) with the former having a USD 3 billion annual turnaround in software and about 20,000 developers. The tasks which must be conducted during development are usually defined in so-called software processes. They are comparable to reference processes that describe in a generic way which tasks need to be considered in software development projects. In order to increase their applicability in various projects, they cover a wide range of functionality and are therefore complex and large in size.

At Siemens AG, business units define software processes within a company-wide Siemens Process Framework (SPF) (Schmelzer and Sesselmann 2004). For presenting process information to users, semiformal event-driven process chains (EPC) (Scheer 2000), function allocation diagrams (FAD) (Scheer 2000) and textual descriptions are used.

Although a lot of effort is invested in developing software processes of high standard fulfilling requirements and best practices as described by process frameworks such as CMMI (2011) and by providing process information in various views (e.g. role-specific view and activity-detail view), the application of the documented processes in daily software development is still often insufficient.

S. Dietrich (✉) • T. Hartmann • E.-M. Kern
Wissensmanagement & Geschäftsprozessgestaltung, University of German Armed Forces, Munich, Germany
e-mail: stefan.dietrich@unibw.de; tomas.hartmann@unibw.de; eva-maria.kern@unibw.de

P. Killisperger • T. Stückl • N. Weber
Corporate Technology, Corporate Research and Technologies, Siemens AG, Munich, Germany
e-mail: peter.killisperger@siemens.com; thomas.stueckl@siemens.com; norbert.weber@siemens.com

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_45,

One reason is the use of business process modelling languages and textual descriptions for describing software processes. Issues include:

- Too much information is presented to the user resulting in complexity and lack of simplicity.
- Notations are used which are nonintuitive/meaningless for software developing practitioners (Derniame et al. 1999).
- The used process description languages do not support formalisation and semantic rules (Oestereich et al. 2003).

The unified modelling language (UML) (OMG 2003) is used extensively in software development for designing, visualising and documenting systems (Killisperger 2010), and intensive tool support is provided (e.g. Rational Rhapsody). UML might therefore provide a better means for modelling software process in practice.

Several approaches exist for modelling software processes in UML (e.g. UML4SPM Bendraou et al. 2005; Di Nitto et al. 2002). This shows that UML in general fulfils the requirements of software process modelling. However, in practice, little use is made of UML when describing software processes because existing approaches either:

- Restrict to a subset of diagrams offered by UML (e.g. Di Nitto et al. 2002) which limit the richness and flexibility of UML
- Provide extensions to the standard UML notation (e.g. SPEM, (OMG 2007) which are not familiar to software developing practitioners and do not allow use of standard UML tools

In a previous project with a car component manufacturer in which software processes were modelled partly by using UML 2.0 diagrams and notations first experiences whether gathered at Siemens AG. This chapter describes the subsequent efforts to use exclusively standard UML 2.0 for describing and modelling the content of software development processes. We describe the analysis of UML diagram types for modelling process information in four distinct views Siemens AG uses for presenting software processes (process-oriented, activity-oriented, product-oriented and role-oriented views).

This chapter has the following structure: first, we present the approach Siemens AG currently uses for modelling their software processes including the process-oriented, activity-oriented, role-oriented and product-oriented views. Based on this, in Sect. 3, we analyse how UML 2.0 diagram types can be used for presenting the same information in the same views. Examples of software processes of Siemens modelled in UML 2.0 are provided. Existing approaches of using UML in software process modelling are analysed in the Sect. 4, followed by a conclusion.

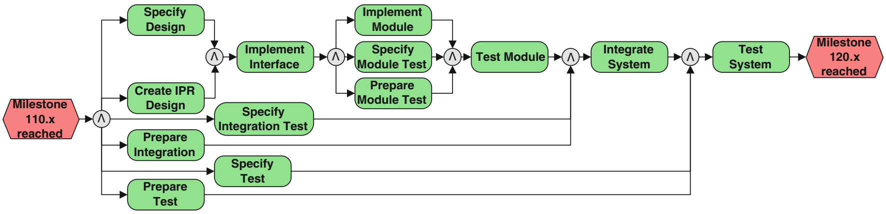


Fig. 45.1 Process-oriented view of subprocess "develop Subsystem (SW)" as EPC

2 Current Practice of Modelling Software Processes

In order to reduce the complexity of process models for software developing practitioners, four distinct process views are used at Siemens AG: (1) process-oriented view, (2) activity-oriented view, (3) product-oriented view and (4) role-oriented view. The following section describes each of the views, the information provided by them and an example.

1. *Process-oriented View*
   The process-oriented view describes the chronological structure (i.e. the order) of the activities of a (sub) process (e.g. activities of a phase in the software process). Activities can represent processes themselves (i.e. complex activities) and thus can create a hierarchy of processes.
   Siemens typically uses event-driven process chains (EPC) for visualising this view. Figure 45.1 shows an example of the subprocess "develop subsystem (SW)" as defined by a business unit of Siemens AG.
2. *Activity-oriented View*
   The activity-oriented view provides detailed information of a single activity. Information of this view typically includes a description of the activity, the tasks the activity consists of, inputs and outputs of the activity, involved roles with their type of involvement (e.g. responsible for and executes) and additional information such as links to templates, guidelines and tools to use.
   Siemens AG often provides this information in function allocation diagrams (FAD) or in textual form. Figure 45.2 shows an example of the latter for activity "specify test (SW)".
3. *Product-oriented View*
   The product-oriented view is similar to the activity-oriented view, with the difference that the focus is on an individual work product (which appears as input or output in the activity-oriented view). Information provided in this view includes a description and the activities the work product is input and/or output of.
   Siemens AG typically provides information in this view in a textual description. Figure 45.3 provides an example of a product-oriented view of work product "test specification (SW)".

Activity: Specify Test (SW)			
Roles	**Input**	**Output**	**Auxiliaries**
■ Tester (SW) ■ User Interface Expert ■ Safety Expert ■ Developer (SW) ■ Technical Writer ■ Requirements Engineer ■ Service Expert ■ Dependability Expert ■ Security Expert ■ Test Manager (SW)	■ Architecture Specification (SW) ■ Design Specification (SW) ■ Integration Test Specification (SW) ■ Module Test Specification (SW) ■ Test Concept (SW) ■ Test Specification (SW)	■ Test Specification (SW) ■ Tracing Database	■ Boundary-Value Analysis (Method) ■ Equivalence Partitioning (Method) ■ Performance Test (Method) ■ Random Test ■ Risk-Based Testing (Method) ■ Test Management System (Tool)
Detailed description: Identify and specify the test cases for the SW verification (=functional and non-functional) tests, according to the SW test concept. For an incremental development this may be the enhancement for the next iteration steps at least incorporate all modification requests. Establish and maintain a bidirectional horizontal tracing between software and relevant systems with software test cases.			
Actions	**Action description**		
Describe test case (SW)	Document the test cases with all necessary attributes: stimulation (e.g. entry data), test algorithm, expected reaction (e.g. output data), test type/stage classification. Document if these tess shall run on the development or the target environment or both.		
Define test case (SW)	Elaborate the test cases for the SW tests base on the system and SW requirements. Ensure that each requirement is covered by a test case.		
Select regression test (SW)	Derived from the test goals and considering the test results from previous versions and increments, identify the regression test cases.		
Trace design interface (SW)	Trace software design entities to software interfaces.		
Trace design (SW)	Trace software architecture entities to software design entities.		

Fig. 45.2 Activity-oriented view of activity "specify test (SW)"

Result: Test Specification (SW)	
Description	
Detailed description: This specifies the Sw test suite in detail, that is used to verify that the functional and non-functional SW requirements are accomplished. It comprises the test cases used to verify or validate implementation of SW requirements. Test cases shall comprise: > name of test case > description of test case > test case ID > relevant test stages (Acceptance, Field, Certificate, System, System Integration, SW System, SW Integration, SW Module,...) > regression test marker (used to mark test cases that shall be used for regression tests) > subject of the test	
is Output of	**is Input for**
■ Implement MR ■ Specify Test (SW)	■ Test System (SW) ■ Specify Test (SW)

Fig. 45.3 Product-oriented view of work product "test specification (SW)"

[linked to task list]

Description	Tasks

Description
Abstract: The SW tester is responsible to verify that the SW sub-system accomplishes the intended functionality, consistent with the SW requirements (functional and non-functional) **Major responsibilites:** Drive: - Verification of functional correctness within the defined subset as described in the SW test specification - Verification of conformity against defined underlying standards and legal/regulatory requirements (e.g. to certification, type approval requiremets) - Verification of non-functional requirements (e.g. performance, load and stress tests) - Write SW test specification - Detect defects, analyze defects and write fault reports - Report to SW test manager

Tasks
Create IPR
Integrate System (SW)
Prepare Test (SW)
Specify Test (SW)
Test System (SW)
Update Test Concept
Write Manual (SE)

Fig. 45.4 Role-oriented view of role "SW tester"

4. *Role-oriented View*

   The role-oriented view contains detailed information of a particular role of the process including its description (i.e. responsibilities), the activities the role is involved in and the way of involvement. Figure 45.4 provides an example of the role-oriented view for role "SW tester" in a textual description.

Table 45.1 Suitability of UML diagrams for views on software development processes

		UML diagram			
		Activity	Class	Sequence	Use case
View	*Process-oriented*	X		X	X
	Activity-oriented		X		X
	Product-oriented		X		X
	Role-oriented		X		X

3 Using UML 2.0 for Modelling Software Processes

By analysing existing UML-based approaches, we identified four different types of UML diagrams which might be suitable for modelling software processes at Siemens AG in the four views presented in the previous section. An overview of the diagrams is provided in Table 45.1.

In the following, the identified UML diagram types are analysed regarding their suitability for presenting the process information of the (1) process-oriented, (2) activity-oriented, (3) product-oriented and (4) the role-oriented views.

1. *Process-oriented View*

   In order to display information of this view, activity-, sequence- and use-case diagrams have been identified as suitable diagram types.

   Activity diagrams have similarity with event-driven process chains (EPC) as they are used at Siemens AG in the process-oriented view. However, activity diagrams only allow the logical connectors “AND” and “XOR”, whereas EPCs allow in addition the logical connector “OR”. The inclusive OR cannot be illustrated in UML (Russell et al. 2006) because of the underlying application of UML as system development tool (Oestereich et al. 2003; Wöss 2009). However, an inclusive OR can be modelled by using multiple XOR connectors which is a work around if the OR connector splits up or joins only few control flows but is a cumbersome and rather confusing solution otherwise.

   Use-case diagrams can be used to provide the same process information as activity diagrams. Each activity is symbolised as a use case. The actor symbol is used to describe the role which is responsible for correct execution. Include/extend relationships illustrate the chronological order similar to the control flow in activity diagrams. An activity which is included by another one will first be executed resulting in an implicit order. If numerous relations are used (e.g. include, extend, responsible for), displaying even little information can result in a process model which is cumbersome to read for practitioners.

   The information of the process-oriented view can also be described by using *sequence diagrams*. The order of activities of a process can be modelled by using communication channels describing activity calls or messages between two process objects. Sequence diagrams are a useful tool for modelling small-scale processes. However, with only a moderate number of process elements to

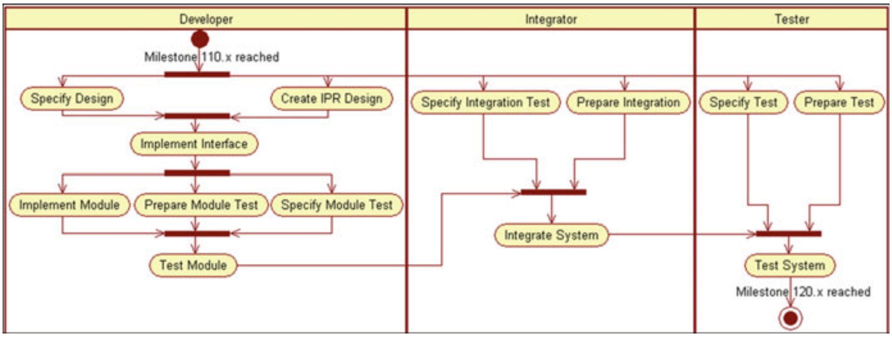


Fig. 45.5 Process-oriented view of subprocess "develop subsystem (SW)" as activity diagram

display, sequence diagrams get fuzzy and confusing and do not allow presenting process information in a clear way.

By comparing process examples modelled in activity, use-case and sequence diagrams, we came to the conclusion that activity diagrams are the best choice for modelling the process-oriented view. Figure 45.5 shows the same subprocess as in Fig. 45.1, but this time modelled by using an activity diagram.

2. *Activity-oriented View*

Potential candidates for illustrating the activity-oriented view of Siemens software processes in UML are class diagrams and use-case diagrams.

When using *class diagrams*, each process element the activity has a relationship with is represented by an individual class in the diagram, with associations detailing the type of relationship (e.g. "responsible role" for the role being responsible for the correct execution of the activity). The actions (i.e. tasks) an activity consists of are represented by using the composition relation. The order the actions have to be executed in is described by using associations specifying the sequence. The detailed description of the activity and its actions are specified by using the note element. Information on the types of classes used in the diagram is modelled by using the inheritance connection (e.g. role "user interface expert" inherits from class "role", work product "design specification" inherits from "specification" and "work product"). However, since all process elements are modelled as classes by using the class symbol, different types of process elements (e.g. activity, work product and role) are difficult to distinguish, especially if a large number of process elements have to be included in the diagram. Package symbols are used to group classes of the diagram (e.g. all actions are grouped to a package) in order to improve readability of the diagram.

Another type of diagram potentially suitable for modelling the activity-oriented view of Siemens software processes is the *use-case diagram*. The activity to be described is modelled as centralised use case. Information about relationships of the activity (e.g. input and output, auxiliaries) is illustrated as separate use cases with associations (e.g. output and useful auxiliaries). Also, the actions the activity consists of are modelled as individual use cases. Include/extend relationships are used to illustrate the chronological order of the actions (i.e. sequence). An action which is included by another one is first executed resulting in an implicit sequence. Roles involved in the execution of the activity are described by using the actor symbol. As for class diagrams, only the note element offers a useful way to display the detailed description of the activity and its actions. Since most information is modelled using the use-case symbol, readability of the diagram can suffer. Here, grouping of the use cases by using the package symbol allows structuring (e.g. actions, inputs and outputs can be grouped in packages).

Both class and use-case diagrams offer a way to describe the activity-oriented view. Figure 45.6 shows the activity-oriented view of activity “specify test (SW)” as use-case diagram. Due to space limitations, the detailed descriptions of the actions of the activity are not included in the diagram.

3. *Product-oriented View*

   Similar to the activity-oriented view, class diagrams and use-case diagrams are potential UML diagram types for modelling the product-oriented view of software processes which provides information on work products and their relationships.

   Class diagrams can be used to model relevant aspects like activities the work product is input and output of. Each process element is detailed by an individual class. Associations, composition and inheritance relations are used to express relationships of the work product with other process elements. The note element is used to display the detailed description of the work product. The expected content of the work product and its structure are described by using separate class symbols. The issue of readability due to using the same notation for different types of process elements can be improved by using the package symbol for grouping process elements (e.g. grouping activities the work product is input of).

   Use-case diagrams are also a potential type for modelling the product-oriented view. Here, the work product is described as centralised use case. Process elements the work product has relationships with (e.g. input and output activities) are modelled as separate use cases connected by associations. The package symbol can be used to group use cases and thus improve the readability of the diagram (e.g. all use cases describing activities the work product is input of are grouped by a package). The detailed description, expected content and structure of the work product can only be modelled with the note element in a useful way.

   We came to the conclusion that class diagrams are the best way to model the product-oriented view. Figure 45.7 shows the product-oriented view of work product “test specification (SW)” using a class diagram.

4. *Role-oriented View*

   In order to describe the role-oriented view, the diagram types class and use case can be used.

   When using *class diagrams* for the role-oriented view, we do not use separate class symbols for process elements having a relationship with the process

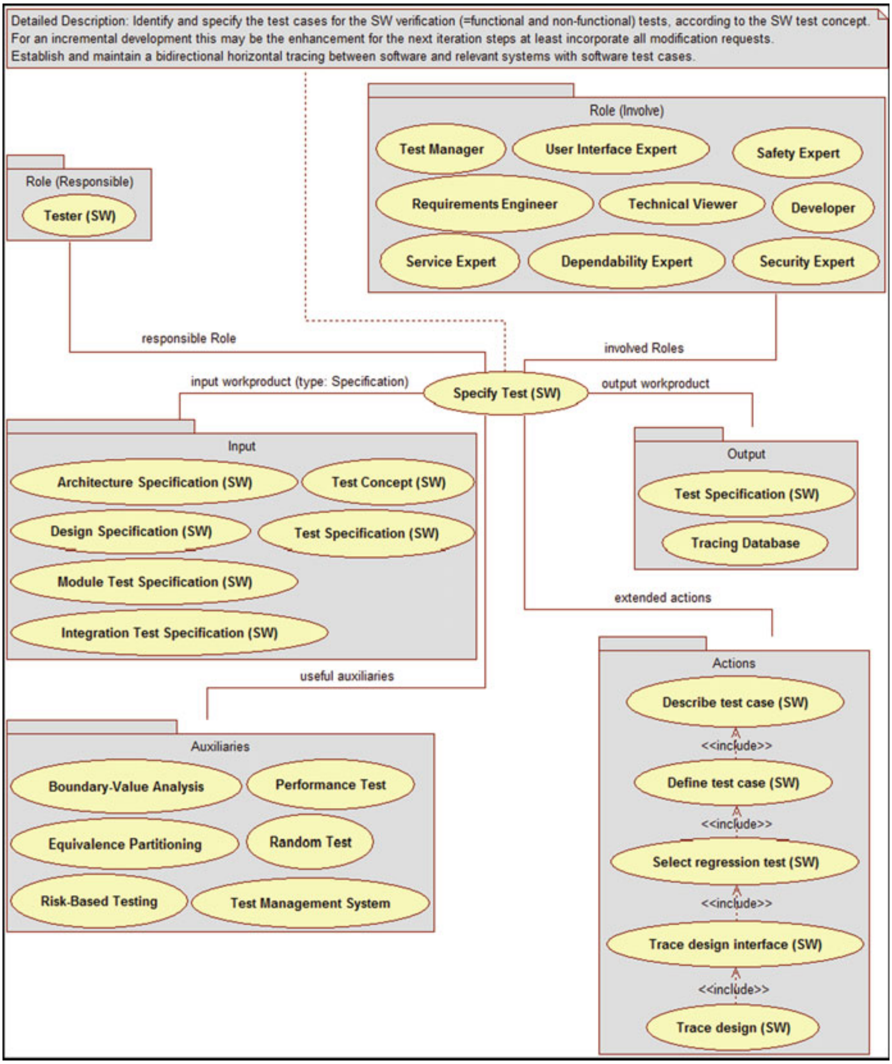


Fig. 45.6 Activity-oriented view of activity “specify test (SW)” as use-case diagram

element in focus (i.e. the role) as we do when considering class diagrams for the activity-oriented and product-oriented view. Here, properties of a role (e.g. role description) are modelled by using the attribute part of the class symbol. Activities the role is involved in are listed in the operation section of the class symbol. However, modelling the latter information in this way does not allow specification of the type of involvement. In order to display this information, individual class symbols for each process element the role has a relationship with have to be used.

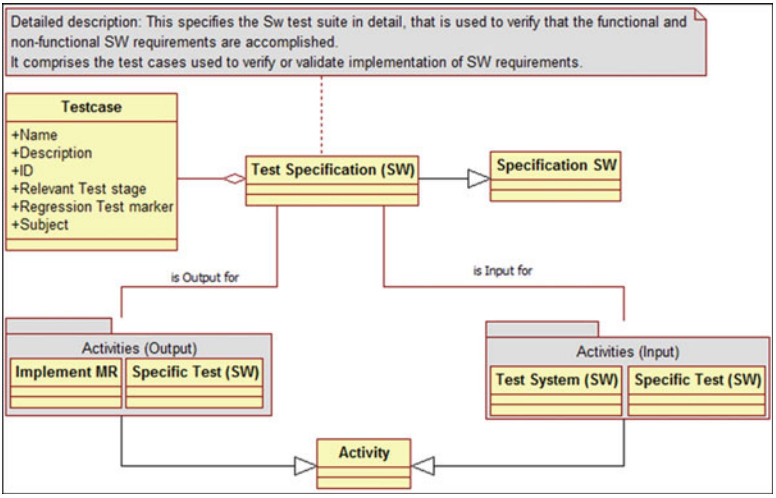


Fig. 45.7 Product-oriented view of work product "test specification (SW)" as class diagram

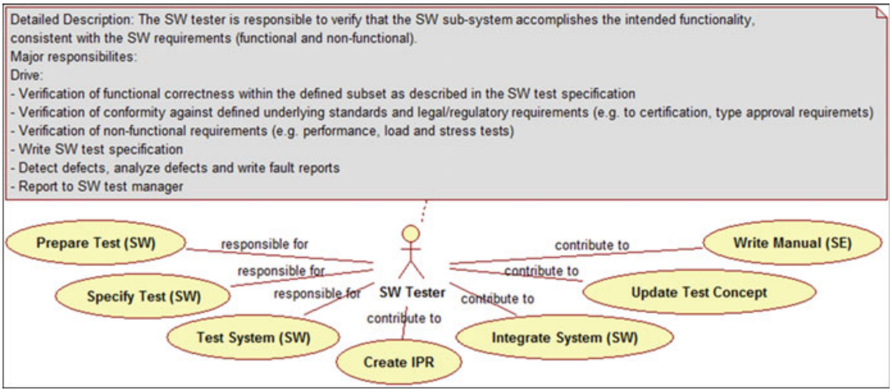


Fig. 45.8 Role-oriented view of role "tester" as use-case diagram

When describing information of the role-oriented view by using *use-case diagrams*, each activity the role has a relationship with is described as a use case of the diagram. The type of involvement is described by associations. The description of the role in the role-oriented view is modelled by using the note element.

Both class- and use-case diagrams allow modelling the role-oriented view as used by Siemens AG. Figure 45.8 shows the role-oriented view for the role "tester" using a use-case diagram.

4 Related Work

Often, business process modelling languages such as EPCs were used to specify and document processes implemented by business applications. However, the actual application was designed by using UML which is a de facto standard in software development which made the transfer of information from, e.g. EPC to UML necessary.

This resulted in the development of several UML-based approaches for modelling business processes since the late 1990s including EPE (Eriksson-Penker extension) (Eriksson and Penker 2000) and OOBPM (object-oriented business process management) (Oestereich et al. 2003).

In the following, approaches for modelling software processes in UML were developed. These included SPEM (Software & System Process Engineering Metamodel) (OMG 2007), UML4SPM (UML for Software Process Modelling) (Bendraou et al. 2005) and the approach developed by De Nitto et al. (2002).

SPEM is a meta-model language to describe software processes. Since the release of 2.0, it contains an own notation extending standard UML 2.0 elements to describe software process models. The idea of SPEM is to illustrate processes as abstract objects which are executed by role operations (so-called activities) on real objects (e.g. software tools). This allows several types of software processes. Basis of this concept is a system of seven packages which are based on one another. Each package defines a certain process structure. Since the process models are highly formalised, they can be automatically validated. SPEM has strong tool support, e.g. EPF (Eclipse Process Framework). Disadvantages of SPEM include the restriction to the meta-model of SPEM and UML symbols defined in addition to the standard UML 2.0 notation. The former implies that organisations using a different meta-model for their processes have to stick to the SPEM meta-model. The latter has the effect that only tools supporting additional UML symbols defined by SPEM can be used.

UML4SPM has been developed with the goal to provide an approach which is easy to integrate in existing UML tools and easy to learn. The approach is based on the meta-model of UML but uses some additional elements specific to software processes. The architecture of UML4SPM is based on a package system similar to the architecture of SPEM. UML4SPM allows automatic execution of process models (Bendraou et al. 2007, 2009). However, tool support for UML4SPM appears not to be in a mature state. Standard UML tools cannot be used since UML4SPM does not use standard UML 2.0 but defines UML extensions. Inherent part of UML4SPM is a meta-model for processes resulting in the need to adopt the prescribed meta-model even if organisations use a different meta-model (e.g. Siemens AG Killisperger 2010).

The approach developed by *Di Nitto* uses UML 1.3 without any extensions. Di Nitto's approach aims to create software processes which can be automatically executed by Di Nitto's OPSS (ORCHESTRA Process Support System). Three different types of UML diagrams are used to describe processes: activity, class and state machine diagrams. The limitation to standard UML 1.3 (i.e. no additional notation elements) offers a wide range of standard UML tools to be used. However, the restriction to only activity, class and state machine diagrams limits the richness and flexibility of process modelling.

As described above, several approaches for modelling software processes by using UML have been developed. However, none of these approaches allow modelling Siemens software processes in the four specified views (i.e. process, activity, product, role-oriented views) by using standard UML 2.0 tools. Approaches are either limited to an earlier version of UML, do not allow use of diagram types required for modelling software processes at Siemens AG or define additional notation elements which do not allow use of standard UML 2.0 tools.

5 Conclusion

Siemens AG is currently undertaking efforts to improve their software process-related activities. Part of these efforts is the improvement of the readability of processes for software developing practitioners. Processes are currently described by using semiformal process description languages and textual descriptions. This chapter described the analysis of standard UML 2.0 diagrams regarding their suitability for describing Siemens' software processes in four distinct views (i.e. process-oriented, activity-oriented, product-oriented and role-oriented views) which Siemens uses for presenting processes to practitioners. The analysis showed that although UML 2.0 has not been explicitly developed for describing software processes, it is suitable for describing Siemens' software processes. The theoretical suitability was confirmed by applying the developed concept on real-world software processes of Siemens AG.

References

Allweyer T (2005) Business process management. W3L, Bochum (in German)

Becker J, Mathas C, Winkelmann A (2009) Business process management: focused on IT. Springer, Berlin/Heidelberg (in German)

Bendraou R, Gervais M-P, Blanc X (2005) UML4SPM: a UML 2.0-based metamodel for software process modelling. In: Briand L, Williams C (eds) Model driven engineering languages and systems: 8th international conference, MoDELS 2005. Springer, Berlin, pp 17–38

Bendraou R, Sadovykh A, Gervais M-P, Blanc X (2007) Software process modelling and execution: the UML4SPM to WS-BPEL approach. In: Muller P (ed) Proceedings of the 33rd EUROMICRO conference on software engineering and advanced applications (SEAA 2007). IEE Computer Society, Los Alamitos, pp 314–321

Bendraou R, Jezequel JM, Fleurey F (2009) Combining aspect and model-driven engineering approaches for software process modelling and execution. In: Wang Q, Garousi V, Madachy V, Pfahl D (eds) Trustworthy software development processes: international conference on software process, ICSP 2009. Springer, Berlin/New York, pp 148–160

CMMI (2011) Capability Maturity Model Integration (CMMI) version 1.3. http://www.sei.cmu.edu/cmmi/tools/cmmiv1-3/. Accessed 3 May 2011

Derniame J, Kaba BA, Wastell D (1999) Software process: principles, methodology, and technology. Springer, Berlin/Heidelberg

Di Nitto E, Lavazza L, Schiavoni M, Tracanella E, Trombetta M (2002) Deriving executable process descriptions from UML. In: Tracz W, Magee J, Young M (eds) Proceedings of the 24th international conference on software engineering (ICSE 2002). ACM Press, New York, pp 155–165

Eriksson HE, Penker M (2000) Business modelling with UML. Wiley, New York

Fitzgerald B, Russo N, O'Kane T (2003) Software development method tailoring at Motorola. Commun ACM 46(4):65–70

Garrido JM (2009) Object oriented simulation: a modelling and programming perspective. Springer, Heidelberg

Killisperger P (2010) Instantiation of information systems development processes. AVM, Munich

Mili H, Tremblay G, Jaoude GB, Lefebvre E, Elabed L, Boussaidi GE (2010) Business process modelling languages: sorting through the alphabet soup. ACM Comput Surv 43(1):4:1–4:56

National Research Council (2002) Modelling and simulation in manufacturing and defense systems acquisition: pathways to success. National Research Council, Washington, DC

Oestereich B, Weiss C, Schröder C, Weilkins T, Lenhard A (2003) Object-oriented business process management with UML. dpunkt.verlag, Heidelberg (in German)

OMG (2003) OMG Unified Modelling Language (OMG UML), superstructure. Version 1.5. http://www.omg.org/spec/UML/1.5/. Accessed 18 Apr 2011

OMG (2007) SPEM 2.0, software & system process engineering meta-model, specification. Version 2.0. http://www.omg.org/spec/SPEM/2.0/. Accessed 18 Apr 2011

Russell N, van der Aalst WMP, Hofstede AHM, Wohed P (2006) On the suitability of UML 2.0 activity diagrams for business process modelling. In: Stumptner M, Hartmann S, Kiyoki Y (eds) Proceedings of the 3rd Asia-Pacific conference on conceptual modelling (APCCM 2006), vol 53. Australian Computer Society, Darlinghurse, pp 95–104

Scheer AW (2000) ARIS- business process modelling, 3rd edn. Springer, Berlin/Heidelberg

Schmelzer HJ, Sesselmann W (2004) Business process management in practice: increase productivity – increase value – satisfy customers, 4th edn. Hanser, Munich (in German)

Sommerville I (2007) Software engineering, 8th edn. Addison-Wesley, Harlow/New York

Staud J (2006) Business process analysis: EPC and object-oriented business process modelling for business standard software, 3rd edn. Springer, Berlin/Heidelberg (in German)

Schütze J (2009) Basic approaches for modelling communication processes at SME networks. Gabler, Wiesbaden (in German)

Wöss P (2009) Current concepts for modelling business processes: a critical comparison. Igel, Oldenburg (in German)

Chapter 46
A Formal Approach to Designing and Managing Service Ecosystems

Aditya Ghose

1 Introduction

Most large-scale service delivery settings involve complex collections of interdependent services. The services of interest, in this context, could be web services or business services – much of the discussion below applies in either case. Consider the services that would be offered by an airport (the focus in this example being on business services), such as passenger check-in, baggage handling, passenger security screening, customs, cargo handling, food courts, lounges, aircraft refelling, aircraft maintenance and air traffic control (to name just a few). There are several aspects of service delivery settings such as these that make it necessary to define specialized machinery to support the design and redesign of service collections. First, the number, scale and complexity of the services on offer are large. Second, most of these services are interdependent (both in terms of design and execution). For instance, the design of the passenger check-in service is constrained by the design of the passenger security screening service (Should visas be checked at check-in or at the security screening stage?). Third, changes to any one of these services are likely to impact several other services. In other words, *change propagation* is potentially complex, with the impact of a given change manifested via the redesign of services quite far removed from the service where the initial change was made. Fourth, there are multiple alternative ways in which changes to a given service might be implemented. For instance, a change to aviation authority regulations requiring that all passengers must be checked against a national "watch list" could be implemented by requiring airlines to perform this check or by having this check performed by the customs service at emigration checkpoints. Fifth, there is an important imperative to minimize the extent of change (to protect

A. Ghose (✉)
Decision Systems Laboratory, School of Computer Science and Software Engineering, University of Wollongong, Wollongong, NSW 2522, Australia
e-mail: aditya@uow.edu.au

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_46,

investments in existing service infrastructure, for instance. Sixth, some of these services exist to realise component functionalities of other services. For instance, a small airport supporting short-hop flights might offer a catering service only because some of the airlines using the airport would like to offer a lounge service to their premium passengers. Seventh, there are multiple design alternatives for determining a service landscape that realises the required functionalities (some of the preceding examples have illustrated this on a smaller scale). Finally, these collections of services must operate under complex constraints imposed by the domain (including compliance constraints).

There are no easy ways of dealing with the design, maintenance and full life-cycle management of such complex collections of services. In this chapter, we argue that a formal service ecosystems view can provide a particularly useful solution to the problem. Our intention is to leverage the ecosystems metaphor by using mathematical characterisations of such ecosystems – in particular of equilibria. In our conception of service ecosystems, service designs will play a role analogous to that of biological entities in a biological ecosystem. As in biological ecosystems, service designs are created (or discovered, using automated toolkits Ghose et al. [2007]; modified during their lifetimes; and eventually discarded). Like biological ecosystems, service designs undergo constant change, driven by changing requirements or changes in the operating context. Like biological ecosystems, perturbations in a service ecosystem propagate across its constituent services, driven by the need to maintain a range of critical interservice relationships. These include:

– Functional dependencies: These exist between a pair of services when one of the services depends on the other for realising some of its functionality. We may interchangeably describe these as realisation links. In many settings, these links provide the existential rationale for a service, i.e. the reason why a service exists (these are critical in any account of servitization).
– Consistency links: In many cases, service designs might be related via consistency constraints. These are distinct from realisation links in the sense that the services might not depend on each other for realising their functionality but might have intersecting functional signatures (the set of objects/artefacts impacted by the service).

As in biological systems, service ecosystems are characterised by competing forces (such as the competing pulls of alternative ways of realising a given service functionality or alternative resolutions to inconsistency between a pair of service designs). Finally, like biological ecosystems, service ecosystems settle into equilibria after being perturbed. An equilibrium in an ecosystem is a "steady state", where the competing forces balance each other out. Changes to an ecosystem perturb these equilibria, but the system eventually settles into a new equilibrium that accommodates these changes. We will deem a service ecosystem to be in an equilibrium if all interservice realisation and consistency links are satisfied and there is no alternative equilibrium that further minimises change to the prior state of the service ecosystem. Several key tasks in services engineering, including the implementation, deployment and life-cycle management of services, require the computation of service ecosystem equilibria.

The recent interest in services science has led to research one a more general notion of service management, using a multidisciplinary synthesis of thinking from the disciplines of computing, management, marketing and potentially several others. Business services can be of many different kinds. The notion includes within in its ambit business process outsourcing services, clinical services, customer contact services as well as IT-enabled services, to name a few representative examples. Our objective in this chapter is develop techniques that can apply to the full spectrum of services from web services on the one extreme to business services on the other.

Parts of this framework leverage intuitions from earlier work on model ecosystems Ghose and Koliadis [2008]. We believe that the service ecosystems framework leads to an adequate methodological basis for service life-cycle management.

2 Service Modeling

A service model in BSRL Ghose et al. [2010] consists of the following components: (1) *Service ID* (2) *Preconditions* (3) *Post-conditions* (4) *Inputs* (5) *Outputs* (6) *Goals:* These are the intended effects (post-conditions) of a service (note that not all post-conditions are intended – e.g. it might not be my goal to debit my account with a certain amount of money but that might be the effect of seeking to achieve the goal of purchasing an airline ticket using an online ticket booking service). (7) *Assumptions:* These are conditions on whose validity the execution of a service is contingent but whose validity might not be verifiable when a service is invoked or during its execution. Assumptions need to be monitored during the execution of a service – if a violation is detected (once the truth status of the condition is revealed), the service may have to be aborted. Assumptions are common in informal service descriptions, but might not be identified as such. In our work modeling business services offered by government agencies, we have found references (in textual service descriptions) to lists of "client responsibilities". These are statements of what service clients are responsible for doing, in order to enable the provider to fulfill the relevant service. These are clearly not preconditions, since they cannot be evaluated when a service is invoked. Indeed, checking to ensure that a client has fulfilled all client responsibilities is impractical in general. Instead, one can use the nonfulfillment of these responsibilities as a trigger for aborting the execution of a service or for abrogating the contractual obligations of the service provider. "Force majeure" clauses in contracts are also examples of assumptions (i.e. the provider commits to delivering a service provided no natural disaster intervenes). (8) *QoS specifications* (9) *Delivery schedules* (10) *Payment schedules* (11) *Penalties* (12) *Value model:* For each stakeholder in the service, a distinct value model is included in the service description. A value model represents how a service delivers value to a given stakeholder. A value model can serve as the basis for service design and for redesign in the face of change (where the impact on value models of alternative redesigns provides the basis for deliberation on how best to implement change). BSRL uses the strategy modeling language (SML) Ghose et al. [2010] for modeling value – the intuition being that we can best describe the value proposition of an artefact in terms of the strategies it helps realise. SML supports strategy modeling in terms of *goals*, *plans* and *objective functions* (details omitted due to space constraints). (13) *Resource model:* The ability to understand how a service needs to be provisioned is a critical component of service design. This understanding also underpins any attempt at service optimisation. A resource model describes available resources in a manner as expressive as a UML class diagram, with the usual part-whole and generalisation-specialisation relationships. In addition, a special uses relationship is required to describe how a given resource might use another. In general, a set of BSRL service models might share a common resource ontology – the resource model for a service is then a reference to a set of resource classes/ instances in this ontology. These might be at different levels of abstraction. For instance, a service might be described as using a "printer" resource or more specifically an "inkjet printer" resource or even more specifically an "HP inkjet printer resource". Note that the notion of a resource is general and might include in its ambit people, tools, energy, other operating inputs and so on.

We note that not all service models will populate every component of the template described above. We do not commit to a specific language for representing pre- and post-conditions, goals and assumptions. These could be described informally in natural language or formally (such as in temporal logic, as used in goal-oriented requirements engineering van Lamsweerde et al. [1998]).

3 Service Dependencies

It is useful to make a distinction between *basic* and *derived* services in an ecosystem. A basic service is one that is designed to realise user goals, requirements or strategies. A baggage-handling service in an airport is an example of a basic service. A derived service is one that is designed to execute some of the functionality of a basic service. Services obtained via decomposition of a parent service are derived services. Outsourced services are the most common examples of derived services. An airport may, for instance, choose to outsource the baggage lifting and trucking service to an external service provider and the security screening service to another external provider whilst keeping the baggage tagging and automated sorting services in-house. In this example, the basic service of baggage handling can be viewed as being offered by the airport to meet an obvious client need. Whilst baggage handling appears as a distinct service in the ecosystem, parts of its functionality appear in the service descriptions of the security screening and the baggage lifting/trucking services, respectively. These latter two are derived services in this scenario. Formally, the parent basic service will be viewed as being functionally dependent (and related by functional dependency links) to the derived services. Derived services may, on occasion, further outsource their functionality to other derived services (and thus be functionally dependent on these).

In the following, we will make frequent reference to the notion of inconsistency. In a formal sense, a pair of formal theories are inconsistent if there does not exist a model (in the sense of model-theoretic semantics) that satisfies both theories. Given a sound and complete inference procedure, the derivation of falsity from the conjunction of two theories flags an inconsistency. However, a weaker, more informal notion of inconsistency can also be of interest. A pair of artefacts can be deemed to be inconsistent if they violate any of a set of *inconsistency rules*. Our use of the notion of inconsistency in this chapter is easily implemented via formal theorem provers, but manual or semiautomated approaches leveraging the more informal notion of inconsistency can also be of practical value. In every instance of consistency checking, we will assume that a background knowledge base *KB* is brought to bear, i.e. in checking the consistency of theories $T1$ and $T2$, we actually check the consistency of $T1 \cup T2 \cup KB$. Services in an ecosystem may be found to be inconsistent for a variety of reasons. We present a taxonomy below:

Effect inconsistency: Services with inconsistent effects (or post-conditions) may appear in flawed service ecosystem designs. For instance, a service that offers food and beverage plus duty-free shopping outlets has the effect of slowing passenger transit time between security screening points and departure gates. An usher service has the opposite effect of speeding up passenger transit time between security screening points and departure gates. Clearly, both services should not exist in the same ecosystem, except if the services are invoked under mutually exclusive conditions (i.e. the service preconditions are mutually exclusive). Thus, if the preconditions of the usher service specify that the service is invoked only under conditions of heavy passenger load, then it is reasonable for the usher service to coexist with the food/beverage and duty-free shopping service (whose preconditions specify that the service is to be offered during periods of normal passenger load). Formally, an effect inconsistency is deemed to occur in a service ecosystem if it contains services whose post-conditions are inconsistent but whose preconditions are consistent.

Assumption inconsistency: An assumption inconsistency occurs in a service ecosystem if there exists a set of services related via functional dependency links whose assumptions are inconsistent. In other words, derived services should not be based on assumptions that are inconsistent with the assumptions of the services they are derived from. For instance, the design of the passenger check-in service might be based on the assumption that the checking of passengers against a national security "watch list" is conducted by the customs services at a point further downstream in the passenger handling workflow deployed at the airport. Note that this is an assumption, as opposed to a precondition since it is impossible to verify whether the checking has been completed prior to the invocation of the check-in service – the service assumes that it will be performed by another service at a later point in time (recall that assumptions are conditions that ought to hold for a service to execute but whose truth or falsity cannot always be reliably established before or during the execution of the service). If the same ecosystem contains a customs processing service that assumes that the “watch list”-based checking is performed by the airline passenger check-in service, then we have an instance of assumption inconsistency.

QoS inconsistency: The definition of QoS inconsistency will leverage a notion of constraint entailment defined below. We will say that a set of constraints is satisfiable if and only if there exists an assignment of values to the decision variables (i.e. the variables referred to in the constraints) such that all of the constraints in the set are satisfied. We will use a constraint negation operator ($\neg$) which asserts the negation of the associated constraint predicate. Thus, $\neg(x < 20)$ is $x \geq 20$ and so on. A constraint c is said to entail a constraint c’ if and only if $c \cup \neg c'$ is unsatisfiable. A quality-of-service inconsistency occurs in a service ecosystem if there exist a pair of services S1 and S2 where S1 depends on S2 and there exist QoS constraints c1 (associated with service S1) and c2 (associated with service S2), where both c1 and c2 have intersecting signatures (i.e. they refer to at least one common QoS variable) and c2 does not entail c1. Consider, for instance, the baggage-handling service, which offers a guarantee of baggage processing within 25 min (encoded, say, by the constraint $PT < 25$, where PT refers to processing time). Consider, further, a situation where the baggage lifting/trucking service which guarantees a maximum processing time of 35 min ($PT < 35$). Intuitively, we have a problem because the outsourced service might potentially lead to the violation of the QoS guarantee offered by the baggage-handling service. This is also captured by the QoS inconsistency condition since the constraint c2 ($PT < 35$) does not entail the constraint c1 ($PT < 25$).

Resource inconsistency: Under the assumption that every service ecosystem is associated with a set of resource constraints (specifying, for instance, the total budget available), resource consistency stipulates that the resource requirements of individual services in an ecosystem do not exceed the maximum available for a given resource type. We can thus flag situations where the current design is likely to violate what are often fairly hard resource constraints.

4 Equilibrium in Service Ecosystem

In this section, we will address the question of how a stable, “workable” version of a service ecosystem might be identified from a potentially flawed design. This requires the definition of an *equilibrium*, where the competing pulls of service ecosystem are resolved. Given that most service ecosystems involve multiple sets of stakeholders, often with vested interest in the provision of particular services in particular ways, the resolution of competing pulls is an exercise of considerable practical value. We will define the basis for decision support functionality for helping service designers identify equilibria from service ecosystem designs which are not in equilibrium (i.e. which are *equilibrium-perturbed*).

A key requirement for a well-designed service ecosystem is that there should be no redundant replication of functionality across services. One way of capturing this requirement is to ensure that there do not exist services S1 and S2 in the ecosystem such that the post-condition of S2 is entailed by the post-condition of S1 (if this were to be the case, then S2 would be entirely redundant, since all of its functionality would be achieved by S1). However, we need to allow for the possibility of both the service that outsources part of its functionality (S1) and the derived service (S2) that executes this outsourced functionality to coexist in the ecosystem. In such situations, the post-conditions of S2 would be entailed by the post-conditions of S1.

Formally, service ecosystem is a directed graph $\langle Services, Links \rangle$ *where each node* $s \in Services$ *is a (single) service model represented in BSRL and each edge* $l \in Links$ *represents a functional dependency (with the directed edge pointing toward the derived service).* We assume that all services in an ecosystem may execute concurrently. We do not represent consistency relations explicitly in the graph, since every service must be consistent with every other service under each of the consistency relations discussed above.

Formally, a service ecosystem will be deemed to be nonredundant if and only if there do not exist services S1 and S2 in the ecosystem, where S1 is not functionally dependent on S2, such that the post-condition of S2 is entailed by the post-condition of S1.

Formally, a service ecosystem is in equilibrium if it is nonredundant and there are no inconsistencies. In general, multiple possible equilibria might be identified from a given service ecosystem – these represent the different ways the various inconsistencies and redundancies might be resolved.

The idea of *identifying* an equilibrium from a service ecosystem is based on the following intuitive observations:

- An equilibrium is itself a service ecosystem.
- An equilibrium can be obtained from a service ecosystem by modifying it so that there are no inconsistencies amongst the constituent services and no redundancies in service functionalities.
- These changes must be made by adding, removing or modifying individual services or functional dependency links.
- Whilst modifying a service ecosystem to obtain an equilibrium, we would seek to minimise the extent of change. An equilibrium is thus the "closest" variant of an existing ecosystem that satisfies the conditions of consistency and nonredundancy. The notion of minimal change in our definition of service ecosystem equilibrium is important, since we wish to ensure that information is discarded (by modifying service designs, or discarding them, to satisfy functional dependencies and consistency links) only if there is a strong justification for doing so. In a similar vein, we may generate new service designs but only when there is adequate justification for doing so. Measuring and minimising the extent of change to service designs is a complex task.

Critical to the process of identifying an equilibrium is the ability to assess proximity between ecosystems. To obtain such a notion of proximity, we will require the ability to assess proximity between individual service models. In the context of BSRL, this requires us to assess proximity between the constituent elements of a BSRL service model. In assessing proximity between BSRL service models, we will ignore penalties, payment schedules and resources – intuitively, these contribute little to our assessment of how similar two services might be. For simplicity, we will only consider goals and not full value models in the assessment of proximity (although the analysis can be easily extended). Also for simplicity, delivery schedules will be viewed as a subcategory of QoS constraints.

In terms of representation, we can classify the component elements of a BSRL service model into three categories:

Sets: Inputs and outputs are simply represented as sets of objects.

Conditions: Preconditions, post-conditions, goals and assumptions are represented as sentences in an underlying formal language, which might be propositional logic, first-order logic, temporal logic or some other – much of the abstract machinery is agnostic to the specific choice. We only require that each condition be written in conjunctive normal form (CNF) with prime implicates. The CNF format ensures that we are able to view each condition as a set of clauses. The prime implicate requirement ensures that the clauses are nonredundant.

Constraints: For simplicity, we shall assume that QoS constraints and schedules in BSRL are represented as numeric unary constraints. Thus, a constraint is written as *VariableRValue*, where *Variable* is a QoS (or scheduling) variable, *Value* is a numeric value and R is one of $\{ =, \leq, < \}$ (e.g. $processing - time < 20$).

We will refer to the various BSRL model components discussed above as *set-*, *condition-* and *constraint-based service descriptors*, respectively. A set-based service descriptor D_{11} of service S_1 is *as proximal* to the corresponding descriptor D_1 of service S as descriptor D_{21} of service S_2 (where D_1, D_{11} and D_{21} might represent, for instance, sets of service inputs) if and only if $D_{21} \cap D1 \subseteq D_{11} \cap D_1$. We write $D_{11} \preceq_S D_{21}$ if this is the case. An alternative intuition is to define $D_{11} \preceq_S D_{21}$ if and only if $|D_{21} \cap D1| \leq |D_{11} \cap D_1|$. The set-based representation of condition-based service descriptors allows us to use the same machinery for proximity assessment as that for set-based descriptors.

We assess the distance between a pair of unary constraints involving the same variable by taking the smallest difference between a pair of value assignments to the variable that satisfy the corresponding constraints, respectively. Thus, the distance between $x < 10$ and $X > 20$ is 12 if the constraint domain is the set of natural numbers (the "nearest" satisfying assignments are $x = 9$ and $x = 21$, respectively). Under this definition, consistent constraints are at a distance of 0 from each other. We will use the commutative function $d : C \times C \rightarrow \mathcal{R}$ to denote this distance, where C is the set of all constraints and $\mathcal{R}$ is the set of real numbers. Let C, C_1 and C_2 be sets of constraint-based service descriptors for services S, S_1 and S_2, respectively, where C, C_1 and C_2 refer to the same of QoS variables and no variable is referred to more than once in the same set. We define $C_1 \preceq_S C_2$ if and only if d

$(c_{1i}, c_i) \leq d(c_{2i}, c_i)$ for all $c_i \in C$, $c_{1i} \in C_1$ and $c_{2i} \in C_2$. Note that scheduling constraints are represented in the same format as QoS constraints.

For the purposes of proximity assessment, we will view a BSRL service model as a 7-tuple $\langle Inputs, Outputs, Preconditions, Post-conditions, Goals, Assumptions, QoSConstraints\rangle$ (we omit Service-IDs for ease of exposition). Given a BSRL service model S and two variants $S1$ and $S2$, we will say $S1 \preceq_S S2$ if and only if the following hold: (1) $Inputs_{S1} \preceq_S Inputs_{S2}$, (2) $Outputs_{S1} \preceq_S Outputs_{S2}$, (3) $Preconditions_{S1} \preceq_S Preconditions_{S2}$, (4) $Post-conditions_{S1}$ $_S Post-conditions_{S2}$, (5) $Goals_{S1} \preceq_S Goals_{S2}$, (6) $Assumptions_{S1} \preceq_S Assumptions_{S2}$ and(7) $QoSConstraints_{S1} \preceq_S QoSConstraints_{S2}$. Given a set of service $S = \{ s_1, \ldots, s_n\}$ and two variants $S' = \{ s_1', \ldots, s_n'\}$ and $S' = \{ s_1', \ldots, s_n'\}$, S' is said to be *as proximal* to S as S' (written as $S' \preceq_S S'$) if and only if for each i, $s'_{i s_i} s''_i$. We will write $S' \prec_S S'$ if and only if $S' \preceq_S S'$ and $S' \npreceq_S S'$.

We will now define the machinery for identifying an equilibrium from a potentially *equilibrium-perturbed* service ecosystem. For the time being, we will assume a one-to-one correspondence between services in the original ecosystem and the equilibrium that is identified. *Formally, given an ecosystem* $\langle Services, Links\rangle$ *where* $Services = \{ s_1, \ldots, s_n\}$, *an equilibrium* $\langle Services', Links'\rangle$ *whereServices'* $=\{ s_1', \ldots, s_n'\}$ *can be identified if and only if (1)* $\langle Services', Links'\rangle$ *is consistent and nonredundant and (2) there exists no other viable ecosystem* $\langle Services', Links'\rangle$ *where* $Services' = \{ s_1', \ldots, s_n'\}$ *which is also consistent and nonredundant such that* $Services' \prec_{\mathrm{Services}} Services'$. *We assume that if* $\langle s_i, s_j\rangle \in Links$, *then* $\langle s_i', s_j'\rangle \in Links'$ *and* $\langle s_i', s_j'\rangle \in Links'$.

This definition gives us the machinery to assess which amongst a set of candidate variants of an original ecosystem represents an equilibrium. It is easy to see that the machinery defined above can be used to *generate* candidate equilibria (note that in general, there can be many). Details have been omitted due to lack of space. We have not considered situations where ecosystem redesigns violate the assumption of one-to-one correspondence between the corresponding sets of services. The formal machinery above can be extended with graph similarity relations to deal with these, but details are omitted due to lack of space.

5 The SEVA Methodology

In this section, we outline the SEVA methodology for the analysis and design of service ecosystems. SEVA differs from analysis and design methodologies for SOAs such as SOMA Zhang et al. [2008] in its ability to offer a more general notion of service than that of a web service. At this level of abstraction, the differences between the possible execution engines for a service (often human in the case of business services and IT-based in the case of service-oriented computing) cease to matter. The material in this section is largely based on Ghose [2010].

SEVA includes the following key steps. The first is *servitisation analysis*, where we decide what functionalities need to be packaged into a service. The second is *service interdependency identification*, where we identify the functional dependencies and consistency links between the constituent services in a service ecosystem. The third is *service provisioning*, where we correlate services to resource models to understand what resources would be made available to each service. In the following, we outline the first step, which is arguably the most complex:

A key question that we need to answer is the following: given the range of functionalities within an organisational context, how should these be partitioned into a set of services? It might be argued that any subset of the set of functionalities (including singleton subsets) can be viewed as a valid set to be packaged into a service. Ultimately, the notion of a service is useful (1) as the finest level of granularity at which functionality is associated with service providers, (2) as the basis for pricing and (3) as a unit of analysis for performance monitoring (via non-functional requirements or QoS factors). It is clear from practice that some combinations of functionalities "make more sense" packaged as a service than others. We posit that there are two dimensions to this intuitive notion of servitisation:

Manageability: A service must be manageable, in terms of being able to measure its performance relative to its (performance) requirements. Some services have intangible or difficult-to-assess outputs. The quality of an end-to-end tailoring service is easy for a customer to assess, but a measuring/fitting service that only generated the client measurements for downstream tailoring activities would be difficult to manage, since there is no easy way of deciding whether a given instance of the service was effective (a poor quality end product might have been caused by poor quality of measurements, or because of poor quality of tailoring, but there are no easy way to apportion blame). A precondition for packaging a set of functionalities into a service is the ability to identify a set of performance measures that can be directly correlated to payment in a contractual sense.

Utility: Different combinations of functionalities deliver different kinds of value. For example, a service that manufactures left-foot shoes is infeasible in isolation (as is a service that manufactures right-foot shoes). However, a service that manufactures both is meaningful, since the pair (a left-foot shoe and a right-foot shoe) constitutes a saleable unit. This forms the basis for the study of coalition formation Sandholm et al. [1999] and Sombattheera and Ghose [2008]. In this class of techniques, we assume the existence of a characteristic function, which, given any subset of a set of services (or agents), generates a numeric utility value associated with that subset. The optimal coalition structure problem seeks to identify the optimal partitioning of a set of agents (or services), i.e. the partitioning that maximises the total utility across all the partitions (as determined by the characteristic function). Clearly, servitisation requires the identification of optimal ways to partition the set of available functionalities, with each partition being packaged into a service.

The remaining two steps in the SEVA methodology are relatively straightforward. The definitions in the previous section provide for the basis for step 2 (identifying inconsistencies will require checking all possible pairs of services). Correlating service designs with resource models can also be straightforward but has important consequences, such as the fact that the resourcing of a service determines its QoS profile. For instance, the carbon footprint of a printing service is determined by the type of printer used. Similarly, the number of call centre operators allocated to a given account determines the average client wait time (calls are answered faster if more operators are allocated).

6 Related Work

The existing literature on service ecosystems does not address the problem of formal modeling of services and service dependencies, nor does it explore the notion equilibrium. In Barros and Dumas [2006], use the ecosystem metaphor to describe large collections of web services, their architecture in terms of the roles of broker, mediator and provider and aspects of the web service life cycle. In Riedl et al. [2009], offer a reference model (and case study of its application) for quality management in service ecosystems. In Sawatani [2007], Sawatani discusses service value chains, contrasts these with product value chains and argues for ways in which research might contribute to this conception of service ecosystems. In Zambonelli and Viroli [2008], Zambonelli and Viroli discuss how the natural ecosystem metaphors might contribute to the design of service ecosystems. Their approach is similar to ours in seeking inspiration from natural ecosystem metaphors, but their contributions are considerably more abstract. In Scheithauer et al. [2009], Scheithauer et al introduce a detailed approach to service modeling from a range of different perspectives, inspired in part by the Zachman framework for enterprise architectures.

7 Conclusion

Much of the current practice in the design and analysis of service-oriented architectures, as well as business service architectures, is ad hoc in nature. Services are often not modeled in adequate detail, and interservice dependencies are typically ignored. There is a clear need for systematizing these activities. This chapter offers a set of techniques to address this need.

References

Barros AP, Dumas M (2006) The rise of webservice ecosystems. IT Professional 8(5):31–38

Ghose AK (2010) The SEVA approach to service modeling and analysis. Technical report 2010-TR-01, Decision Systems Lab, University of Wollongong

Ghose AK, Koliadis G (2007) Auditing business process compliance. In: Proceedings of the ICSOC-2007, Vienna. Lecture notes in computer science. Springer, Berlin

Ghose AK, Koliadis G (2008) Model eco-systems: preliminary report (Invited talk). In: Proceedings of the fifth Asia-Pacific conference on conceptual modelling (APCCM 2008), Wollongong

Ghose AK, Koliadis G, Cheung A (2007) Rapid process discovery. In: Proceedings of the ER-2007, Auckland. Lecture notes in computer science. Springer, Berlin

Ghose AK, Le LS, Hoesch-Klohe K, Morrison E (2010) The business service representation language: a preliminary report. In: Proceedings of SMART-2010, Banff. Lecture notes in computer science. Springer, Berlin

Hinge K, Ghose AK, Koliadis G (2009) Process SEER: a tool for semantic effect annotation of business process models. In: Proceedings of the 13th IEEE international EDOC conference (EDOC-2009), Auckland. IEEE Computer Society Press, Piscataway

Koliadis G, Ghose AK (2007) Verifying semantic business process models in inter-operation. In: Proceedings of the SCC-2007, Salt Lake City. IEEE Computer Society Press, Piscataway

Riedl C, Böhmann T, Rosemann M, Krcmar H (2009) Quality management in service ecosystems. Inf Syst E-Bus Manag 7(2):199–222

Sandholm T, Larson K, Andersson M, Shehory O, Tohmé F (1999) Coalition Structure generation with worst case guarantees. Artif Intell 111(1–2):209–238

Sawatani Y (2007) Research in service ecosystems. In: Proceedings of the portland international center for management of engineering and technology (PICMET'07), Portland

Scheithauer G, Augustin S, Wirtz G (2009) Describing Services for Service Ecosystems. In: Feuerlicht G, Lamersdorf W (eds) Proceedings of the ICOSOC-2008 Workshops. Lecture notes in computer science, vol 5472. Springer, Berlin, pp 242–255

Sombattheera C, Ghose AK (2008) A best-first anytime search algorithm for coalition structure generation. In: Proceedings of the 7th international conference on autonomous agents and multi-agent systems, Estoril

van Lamsweerde A, Darimont R, Letier E (1998) Managing conflicts in Goal-Driven requirements engineering. IEEE Trans Softw Eng 24:908–926

Zambonelli F, Viroli M (2008) Architectures and metaphors for eternally adaptive service ecosystems. In: Badica C et al (eds) Intelligent distributed computing systems and applications. Springer, Berlin, pp 23–32

Zhang L-J, Zhou N, Chee Y-M, Jalaldeen A, Ponnalagu K, Sindhgatta R, Arsanjani A, Bernardini F (2008) SOMA-ME: a platform for the model-driven design of SOA solutions. IBM Syst J 47 (3):397–413

Chapter 47
Monitoring Information Quality Within Web Service Composition and Execution

Thanh Thoa Pham Thi and Markus Helfert

1 Introduction

Motivated by the idea of "assembling application components into a network of services that can be loosely coupled to create flexible, dynamic business processes and agile applications that span organizations and computing platforms" (Papazoglou et al. 2007), Service-Oriented Computing (SOC) paradigm provides approaches that enable flexible business collaboration and enterprise application integration. Web services are the key technology in SOC, in which services are considered as "autonomous, platform-independent entities that can be described, published, discovered, and loosely couple in novel ways" (Papazoglou et al. 2007). A service-oriented application includes a service provider and a service requester. A service discovery agency (e.g., UDDI) may act as intermediate between provider and requester and provides functionality to promote available services. The service provider defines a service description and publishes it (to the agency). After retrieving a suitable service, the service requester is able to invoke that service (The Open Group: service oriented architecture). In this regard, service composition encompasses the process of searching and discovering relevant services, selecting suitable web services of best quality, and finally, composing these services to achieve an overall goal that usually in a business context aims to support an underlying business process.

On reviewing prominent approaches for service discovery, it appears it mainly involves functional attributes of services advertised in the service description. These include service type, operation name, input/output data format, and semantics (Jeong et al. 2009). In order to select suitable services, a form of quality of

T.T.P. Thi (✉) • M. Helfert
School of Computing, Dublin City University, Dublin, Ireland

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_47,

service (QoS) evaluation is usually used as approach for service selection among many services of similar functionality. In literature, many approaches have been proposed to measure QoS with nonfunctional quality criteria. In this context, QoS dimensions often refer to nonfunctional criteria which include *execution price*, *execution duration*, *reputation*, *reliability*, and *availability* (Zeng et al. 2003). The functional quality of composite service is usually related to the connection of input data and output data between component services. For instance, Lecue and Mehandjiev (2009) developed a method to measure the semantic links between component services; Lee and Choi (2009) proposed to constrain the input data and the output data of services for the composition. Otherwise, it is often assumed that the service functions execute according to the stated and published service description.

However, this does not ensure that all output data during an execution of a composite service based on an underlying process are correct as with any execution and operation; this may not be the case. In contrast to other research, we consider this problem in data quality perspective and provide a framework that can help to detect some of the problems during the execution of the services.

We focus our investigation at the service deployment and execution phase. Motivated by the similarity to an information manufacturing system, we view a composite service as a process that produces information products. In order to produce high-quality information, the manufacturing process needs to be of high quality. This view follows a data quality and information quality perspective (Wang 1998), which has been defined by widely accepted data quality dimensions such as *accuracy*, *completeness*, *consistency*, and *timeliness* (Wand and Wang 1996). Similar to the manufacturing of products, processes that produce poor-quality data are manifested in unsatisfied user requirements. As a consequence, in order to monitor the quality of service composition, we need to evaluate the compliance of service composition and execution with user requirements. Thus, we need to observe the data production process in form of composite services.

In order to introduce and describe this concept, in this chapter, we first investigate data quality issues in service composition and execution and then propose a framework for analyzing data produced by services. We relate our framework to common workflow and service composition approaches. In order to realize the proposed framework, we have developed a service log mechanism that captures all data updates in a workflow system. By analyzing, the service log information, we are able to identify services and service instances that cause poor data quality, and thus this helps to improve the service selection and composition.

The remainder of the chapter is organized as follows. Section 2 presents quality issues in web service composition and execution. Section 3 discusses some related work. Section 4 presents our framework for analyzing data in service composition and execution. Section 5 deals with a case study and the evaluation of our framework. Section 6 presents some concluding remarks and outlines some opportunities for future work.

2 Quality Issues in Web Service Composition and Execution

The process of web service composition and execution (WSCE) consists of defining a workflow that realizes the required functionality followed by its deployment and execution on a runtime infrastructure (Agarwal et al. 2008). The tasks in the workflow are individual and mapped to single services. The composite service is an implementation of a *business process* or a workflow that describes user requirements. The service orchestration defines the order of single service execution and the order of messages exchanged between services in accordance to the workflow. The execution of the composite service accesses to *data*. Figure 47.1 describes quality issues in WSCE context.

It is widely recognized that the quality issues in web service composition and execution can be categorized in two dimensions: *functional* and *nonfunctional* quality. The functional dimension refers to the compliance of one individual service with its service description. This is extended with the compliance of a series of composite service with a defined workflow that captures the user requirements. The nonfunctional quality often addresses the performance of web services including execution price, execution duration, reliability, availability, and reputation (Zeng et al. 2003). A quality model for measuring quality of service composition with functional and nonfunctional levels has been developed in (Lecue and Mehandjiev 2009). In which, the functional level mainly focuses on quality of semantic links between services. Two generic quality criteria considered for semantic links are *common description rate* and *matching quality* of the *output* of the first service and the *input* of the second one. Measuring the nonfunctional criteria focuses on *execution price* and *response time*.

Meanwhile, Fassnacht and Koese (2006) proposes a conceptual model of quality of e-service with three dimensions: *environment quality*, *delivery quality*, and *outcome quality*. Each dimension is refined with subdimensions, in which *information quality* is a subdimension of *delivery quality*. The authors consider *information*

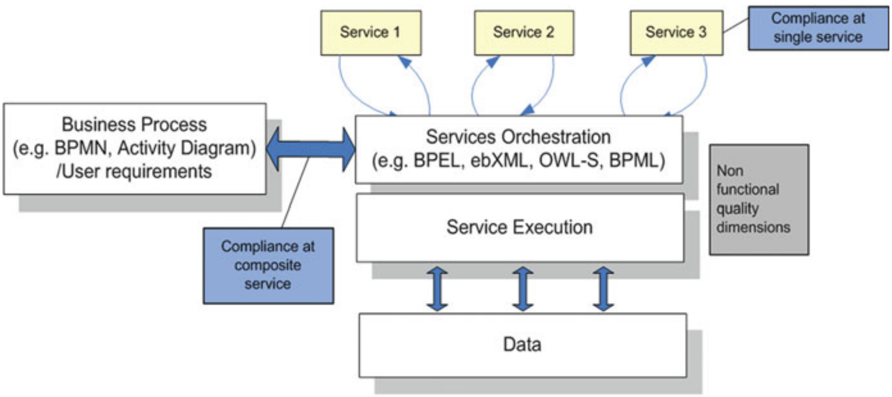


Fig. 47.1 Functional and nonfunctional quality issues in WSCE context

Table 47.1 Data quality issues in the context of web services composition and execution

Quality dimension	Issues at single service	Issues at composite service
Accuracy, completeness	Incorrect input data, typo mistake	Inaccurate orchestration of component services regarding the corresponding business process
	Incorrect output data due to inaccurate implementation of the functionality of the service	Inaccurate implementation of composite service in transferring, calculating, or converting data received (to transfer) from (to) a service
Consistency	Inconsistent data inside a single service	Nonuniform output data of a service to be directly inputted to the following service

quality as "the extent to which complete, accurate, and timely information is provided for the customer during the interaction process with the user interface."

Taking into account quality issues in WSCE, many approaches for static or dynamic web service composition and execution have been developed (Agarwal et al. 2008; Narendra and Orriens 2007; Silva et al. 2009; Zhang 2004). However, quality issues of mapping functional requirements to service composition and execution are current.

By adapting data quality management perspective, a composite service can be considered as an information manufacturing process. Therefore, the quality of data produced by a composite service can be used to evaluate the functional quality of that service.

The data quality literature explored the most four dimensions of data quality which are accuracy, completeness, consistency, and timeliness (Wand and Wang 1996). Corresponding to the functional quality of WSCE, in this chapter, we explore three dimensions of data quality, accuracy, completeness, and consistency (Table 47.1). Analyzing those dimensions helps to clarify the causes of poor data quality in WSCE.

3 Some Web Service Composition Approaches Taking into Account Functional Quality

Today, many web service composition languages have emerged such as BPML (Arkin et al. 2002), OWL-S, and BPEL4WS (Andrews et al. 2003). The later, BPEL4WS is a popular language that defines service composition on the basis of a business process. It includes data flow controls among services to be composed. In addition, business rules are a typical used to constrain the service composition and ensure integrity checks during the execution.

Regarding functional validation of composite services, Lee and Choi (2009) proposes a proactive approach to ensure data quality between composite services by using data constraints. Data constraints enforce the correctness of data transformation between services. This approach also develops a method to regulate data constraint outside the service needed for the composition to fill semantic gaps in data transformation that may occur in service reusability and composability.

In (Yoo et al. 2010), the authors have proposed Petri Nets-based approach to validate the connectivity between single services for a service composition. Meanwhile, Charfi and Mezini (2004) deals with developing a hybrid composition approach to control data flow between single services in accordance to the underlying process. The approach separates business policies and constraints from the core composition logic. This allows to increase the flexibility of composition. At a single service level, functional quality of service (FQoS) metrics have been proposed measuring the similarity-based approximate matching between a query and service description (Jeong et al. 2009). In addition, Yang et al. (2003) proposes a rule-based approach to handle service composition life cycles to support dynamic binding and flexible service composition. Each phase in the life cycle has related rules that need to be taken into account and help to ensure the integrity. The approach takes into account business rules as well as constraints on resource, time, and costs.

These approaches are mainly based on business rules and constraints designed at the service composition phase. However, these approaches are limited to track and monitor service composition and execution often resulting in data quality problems. Furthermore, they do not investigate the correctness and effectiveness of the service composition after the services are executed. As a consequence, most approaches are unable to detect divergence of services execution form user requirements.

4 Framework for Analyzing Data Quality

The proposed framework follows a business rules-based approach to data quality. Business rules are “statements that defines or constrains some aspects of a business” (The Business Rules Group). Aspects include the business policy, the business structure, the organization structure, business objects, and business process. A business rule is normally described by the natural language, which can be reconstructed in form of Event-Condition-Action (Bubenko and Wangler 1993; Herbst 1995), in form of If-Then expression, or in form of <Subject> Must <constraints> (Morgan 2002). In addition, business rules can be expressed in formal languages such as the logic predicate or the first-order logic language or pseudo SQL statement to facilitate business rule implementation. In data quality, business rules are related to business objects or data objects as well as business processes.

Base on common business rule approaches, we developed a framework for WSCE that is presented in Fig. 47.2. The main advantages of this framework are as follows:

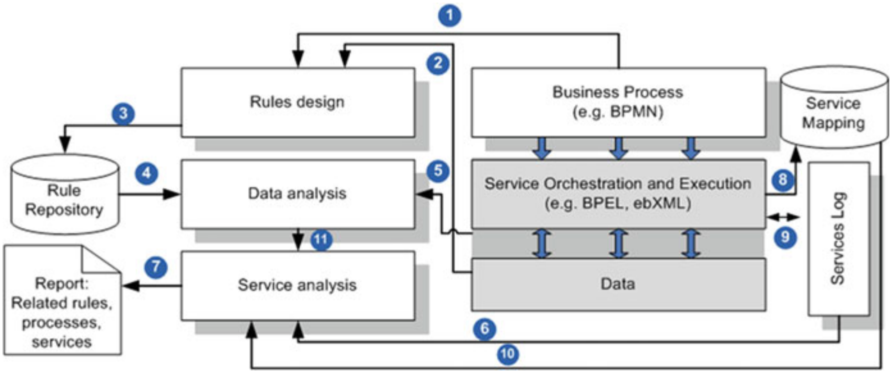


Fig. 47.2 Framework for monitoring data quality within WSCE

- Monitoring the mapping of selected services with tasks of a specific conceptual business process.
- Monitoring the service execution and data produced.
- Consequently, we are able to identify services causing data quality problems.

Figure 47.2 depicts the proposed framework. The *service orchestration and execution* component and *data* component are out of our framework. The detail of other components and relationships between them described by flows of data are explained in the following.

4.1 Business Process

This block represents the conceptual business process (BP) model that a service composition must comply with. A BP model can be described with BPMN, UML, or EPC model. In our framework, we are interested in the input and output of data of activities/tasks in the BP. A part of the metadata of the BP model is stored in the service mapping and rule repository (Fig. 47.3). The business process model and data components are the basis for business rules design (Flow 1 and Flow 2).

4.2 Business Rules Specification

This component concerns specifying business rules. A business rule is related to one or many activities/tasks in a business process and/or data objects. Business rules can be designed manually based on business expert's experiences. However, some complex rules such as dynamic rules, which relate to status change of data

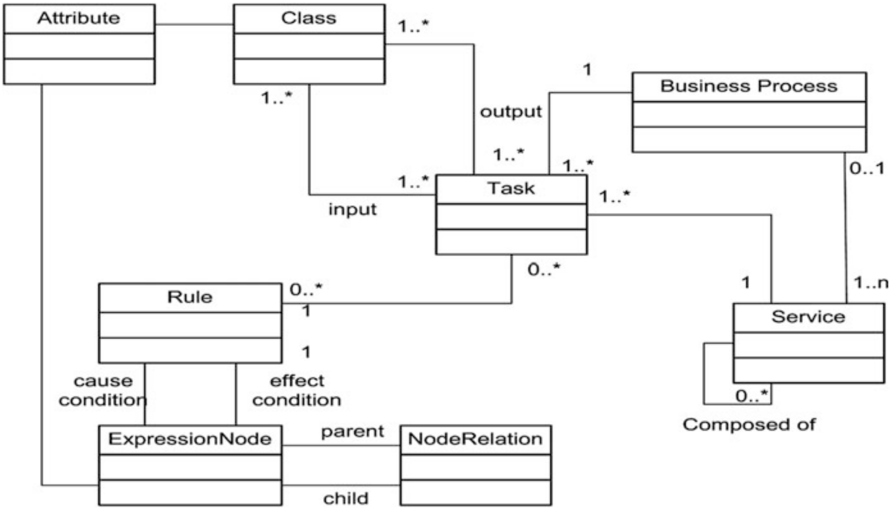


Fig. 47.3 Rule and service-process mapping data model

objects, can be detected and designed with help of some techniques (Pham Thi and Helfert 2010).

Rule specification language is a pseudo logic predicate language which can be automatically converted into code for analyzing data. The rule is usually in form of If *<Boolean logic expression>** Then *<Boolean logic expression>** or an assertion of *<Boolean logic expression>**.

*<Boolean logic expression>** is composed of one or many <Boolean logic expressions> combined together with logical operators. A *<Boolean logic expression>* = *<left expression> <comparison operator> <Right expression>*.

<left/right expression> is a mathematical expression including data attributes, data values, mathematical operations, or aggregation functions.

The specified rules are stored in a *rule repository* (Flow 3) such as relational databases or XML files. The *<Boolean logic expression>** is described with the binary tree structure (Fig. 47.3). The *Expression node* class presents a data attribute or data value or an operator or a function. The *NodeRelation* class expresses the relation between a parent node and a child node within left/right position.

4.3 Service Mapping Repository

This repository captures the mapping between services to be composed to tasks/activities defined in the underlying business process. A task corresponds usually to a service, and a service can correspond to one or many tasks. However, in the case that a task corresponds to many services, the service composition is significantly more complex and is currently not subject of this study. The information is usually stored within the service composition phase (Flow 8). Figure 47.3 depicts the structure of the *rule repository* and service *mapping repository*. The links between service, activities, and process describe mapping information.

4.4 Service Log

The service log captures specific events occurred during the service execution. In our framework, we are particularly interested in events related to data updates and changes. It is necessary to record what service instance in what composite service instance writes what data to the database. A service needs to have information where to write the data; in other words, information about database server, table, and column are known. Furthermore, the record ID is also captured after writing the data. All these information should be stored in the service log (Flow 9).

Although there are approaches to specify log formats (Gaaloul et al. 2008), current approaches such as the Common Log Format (http://httpd.apache.org/docs/logs.htm) and Combined Log Format of W3C are not sufficient for our approach as they are not directly able to represent the required information. Therefore, we propose a practical oriented log file. The log file entries contain the following information:

```
<Entry>
  <Service location> ...</Service location> // registry
  location
  <Service ID>... </Service ID>
<Service instance ID>... </Service instance ID>
<DbServerID>... </DbServerID>
<TableName>... </TableName>
<ColumnName>... </ColumnName>
<RecordID>...</RecordID>
<Data Value>...</Data Value>
</Entry>
```

The *RecordIDValue* contains the value of the primary key attributes of a record in the table *Table Name* in the database *DbServerID*. Data value is the value written to the DB at the column *Column Name*. The primary key attributes of a table is identified based on the metadata of the database.

4.5 Data Analyzing

This component analyzes data in the database against rules defined in the rule repository (Flow 4 and Flow 5). Data that violates some rules will be marked. Information on its Table, Column, and RecordID is reported. Furthermore, based on the information in the rule repository (Fig. 47.3), the task(s) related to these rules is also identified and reported. Each line in the report is composed of the following information: DbServerID, TableName, ColumnName, DataValue, RecordID, RuleID, and TaskID.

4.6 Service Analyzing

This component aims to identify services related to poor data quality. From the data analyzing result (Flow 11) and the mapping information between Task and Service in addition to the service logs (Flow 6 and Flow 10), we are able to identify particular service instances that caused poor data quality.

In order to deploy our framework, we divided the framework into two phases: *preparation* and *analyzing*. The preparation phase concerns with implementing service logs and recording information of the mapping between services and business processes. The analyzing phase addresses the modeling of business rules and then subsequently analyzing data against these rules.

5 Case Study and Discussion

The following case study illustrates our framework with a common service that provides the booking of travelling packages. Initially, customers search information about available travel packages and then subsequently may book a flight and a hotel. Once the booking is completed, the user pays the total amount and confirms the booking. Alternatively, the user may cancel the booking. We developed a service-oriented application for travel package booking and analyzing data quality of the application along the two phases: preparation and analyzing.

In the *preparation* phase, initially following *business process* block in the framework, a conceptual business process model for the booking travelling package is modeled (Fig. 47.4). Next, the service composition process is realized (*service orchestration and execution* block); the mapping information between the tasks of the BP and individual services are stored (*service mapping* repository). We suppose this is a design-time service composition. The service composition can be described with BPEL based on the orchestration depicted in Fig. 47.5.

The composite service *Booking package* is composed of a set of available services: *Book Flight* service, *Book Hotel* service, and *Payment* service. The flows of data/message between services are also described in the orchestration.

Mapping services and tasks in BP are shown in the following table:

Task	Service
Search flight	Book flight
Book flight	
Search hotel	Book hotel
Book hotel	
Payment	Payment

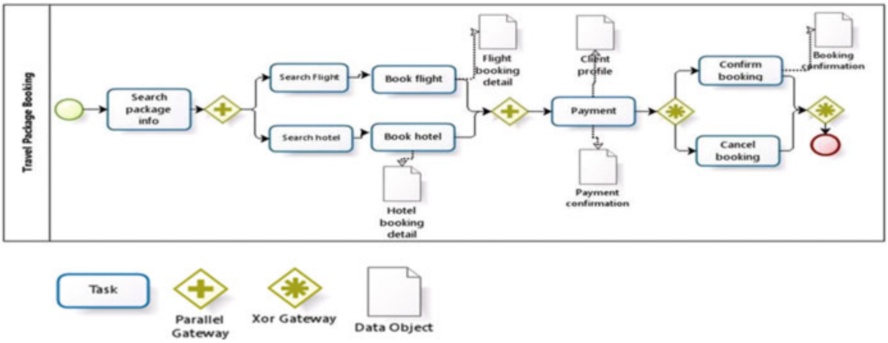


Fig. 47.4 Business process of travel package booking described with BPMN

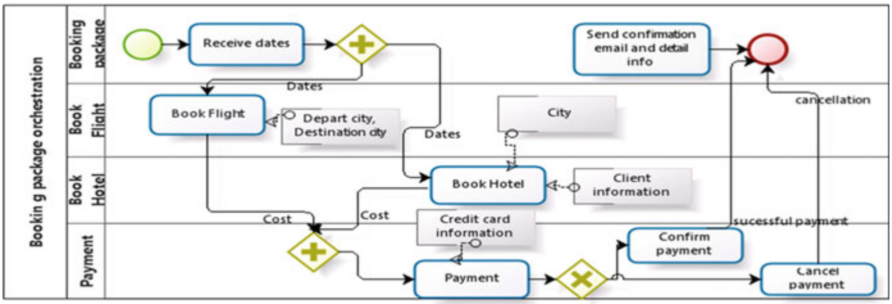


Fig. 47.5 Service orchestration

Other tasks are realized at the composite service levels such as *Search package information* and *Send confirmation email*. The service log techniques are implemented to record data updates events (*service log* repository). The database schema of the application is described in Fig. 47.6.

Once our framework was set up and developed in the preparation phase, subsequently we use our framework in the *analyzing* phase. After deployment of the framework, the data in database and service logs serve for the analyzing. First, business rules are designed in regard to the business process model and data schema depicted in Figs. 47.4 and 47.6.

For our case scenario, we suppose these are the following rules designed:

R1: If the booking is confirmed then the payment must be full paid. This rule relates to *Payment* task and *Confirm booking* task and is described with pseudo SQL as follows:

```
If the PackageBooking.BookingStatus = 'confirmed' then
  Payment.PaymentStatus='Full Paid' AND
  PackageBooking.BookingID=Payment.BookingID
```

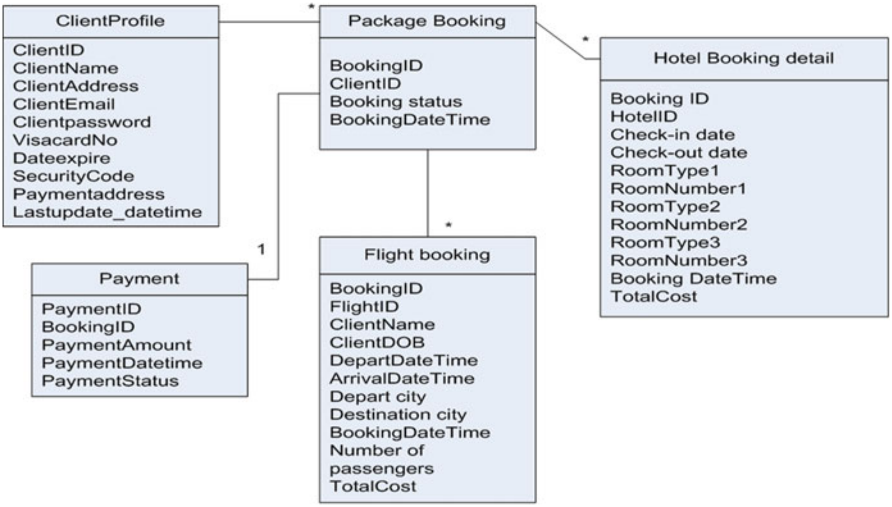


Fig. 47.6 Database schema of travel booking

R2: For every booking, the payment fee must be equal to the total cost of flights, hotels, and including all tax. This rule relates to *Payment* task, *Book Flight* task, and *Book Hotel* task and is described in pseudo SQL:

```
Payment.PaymentAmount=HotelBooking.TotalCost + SUM
  (Flightbooking.FlightCost) AND Payment.BookingID=-
  Flightbooking.BookingID AND HotelBooking.
  BookingID=Payment.BookingID
```

Once the rules that are set up are stored in *rules repository*, the data in the database is analyzed against these rules (*data analysis* blog). These rules can be automatically generated into code to verify data in database. The data-analyzing component is able to distinguish main conditional expressions and join expressions in a rule.

A join expression links foreign keys and primary keys attributes of two tables. For example, for rule R1, the condition is

`If the PackageBooking.BookingStatus = 'confirmed' then Payment.PaymentStatus='Paid'`, the join expression is `PackageBooking.BookingID=Payment.BookingID.`

The data analyzing focuses on the condition expressions; possible attributes related to data errors regarding the two rules are

`PackageBooking.BookingStatus, Payment.PaymentStatus, Payment.PaymentAmount, HotelBooking.TotalCost, and Flightbooking.FlightCost`

Table 47.2 Analyzed data report

DBServerID	Table name	Column name	Data value	RecordID	RuleID	TaskID
DB1	Booking	BookingStatus	"Confirmed"	1200	R1	Payment, confirm booking
DB1	Payment	PaymentStatus	"Unpaid"	3200	R1	Payment, confirm booking

If there is any data that violates a rule, then the related process will be identified based on information stored in the rule repository. For example, suppose there are incorrect data listed in Table 47.2. Also, assume there is a confirmed booking however which is not fully paid. This relates to two records, one in the booking table and the other in the payment table. The rule R1 is related to the tasks Payment and Confirm booking.

Next, the *service analysis* is realized. The service log file describes the data of the `BookingStatus column, RecordID 1200` is produced/written by the composite service *BookingPackage*, meanwhile the data of `PaymentStatus column, RecordID 3200` is produced/written by the service *Payment*.

From the above information, we can identify that the cause of the incorrect data related to Table 47.3 is the Payment service and the BookingPackage composite service.

The case study is motivated on a real-world application, however, significantly simplified and abstracted. Also, we have assumed that key techniques have been implemented within the realistic scenario in order to apply our approach, for example, techniques to store the mapping of service and tasks of a conceptual business process model at runtime service composition, as well as techniques to record and manage the service log. With the case study, we illustrated how our framework can be used. In the discussion with practitioners, our approach has advantages for identifying problems with service execution and thus resulting in unsatisfied user requirements (i.e., caused by poor data quality), and it is also needed to conduct a performance evaluation of our approach in the future.

6 Conclusion

Motivated from a data quality perspective on service composition, in this chapter, we have presented a framework for monitoring web service composition and execution; in particular, we investigate if the service and composite service execution comply with user requirements. User requirements are represented in form of business process and business rules. Our framework follows a data quality management approach and incorporates business rules concepts. We propose that the service composition is based on business processes, and the mapping between tasks in business process and services is recorded. We also propose a new service log format to support the analyzing. Our approach differs from other prominent

Table 47.3 Extracted service log information

Service location	Service ID	Service instance ID	DB ServerID	Table name	Column name	Record ID	Value
url://	BookingPackage	10	DB1	Booking	BookingStatus	1200	Confirmed
url://	Payment	12	DB1	Payment	PaymentStatus	3200	Unpaid
url://	BookFlight	5	DB1				

approaches by investigating service deployment and data produced by services. We have developed a tool for data quality analyzing, and we illustrated the framework within a case scenario that shows that our approach can assist to improve service selection and composition. We have developed business rules specification and data analyzing.

In the future, we aim to extend and develop techniques and tools to support and improve the proposed framework such as techniques to handle service logs and analyzing services, as well as evaluating the performance of the proposed approach.

References

Agarwal V, Chafle G, Mittal S, Sribastava B (2008) Understanding approaches for Web service composition and execution. In: COMPUTE'08 proceedings of the 1st Bangalore annual compute conference. ACM, New York

Andrews T et al (2003) Business process execution language for web services, version 1.1

Arkin A et al (2002), Business process modelling language – BPML 1.0

Bubenko J, Jr, Wangler B (1993) Objectives driven capture of business rules and of information systems requirements. In: Proceedings of the international conference on systems, man and cybernetics, pp 670–677

Charfi A, Mezini M (2004) Hybrid Web service composition: business process meet business rules. In: ACM ICSOC'04. ACM, New York

Fassnacht M, Koese I (2006) Quality of electronic services: conceptualizing and testing a hierarchical model. J Serv Res 9(1):19–37

Gaaloul W, Baïna K, Godart C (2008) Log-based mining technique applied to Web service composition reengineering. Serv Oriented Comput Appl 2:93–110

Herbst H (1995) A meta-model for specifying business rules in system analysis. In: Proceedings of CaiSE'95. Springer, Berlin, pp 186–199

http://httpd.apache.org/docs/logs.htm

Jeong B, Cho H, Lee C (2009) On the functionality quality of service (FQoS) to discover and compose interoperable web services. Expert Syst Appl 36:5411–5418

Lecue F, Mehandjiev N (2009) Towards scalability of quality driven semantic web service composition. In: International conference on web services, Los Angeles, CA

Lee JW, Choi B (2009) Framework for data quality assurance between composite services. Int J Softw Eng Knowl Eng 19(3):307–337

Morgan T (2002) Business rules and information systems: aligning IT with business goals. Addison-Wesley, Boston

Narendra NC, Orriens B (2007) Modeling Web service composition and execution via a requirement-driven approach. In: ACM SAC'07, Korea

Nath AK, Singh R (2010) Evaluating the performance and quality of Web services in electronic marketplaces. e-Serv J 7(1):43–59

Papazoglou MP, Traverso P, Dustdar S, Leymann F (2007) Service-oriented computing: state of the art and research challenges. IEEE Comput 40(11):64–71

Pham Thi TT, Helfert M. (2010) Discovering dynamic integrity rules with a rules-based tool for data quality analyzing. In: CompSysTech 2010, Sofia, Bulgaria, pp 89–94

Silva E, Pires LF, Sinderen MV (2009) On the support of dynamic service composition at runtime. In: ICSOC'09. Springer, Berlin

The Business Rules Group. www.thebusinessrulesgroup.org

The Open Group: service oriented architecture. http://www.opengroup.org/projects/soa/

Wand Y, Wang R (1996) Anchoring data quality dimensions in ontological foundations. Commun ACM 38(11):86–95

Wang R (1998) A product perspective on total data quality management. Commun ACM 41:2

Yang J, Papazoglou MP, Orriens B, Van Heuvel WJ (2003) A rule based approach to the service composition life-cycle. In: Proceedings of the 4th international conference on Web information system engineering. IEE Computer Society, Washington, DC

Yoo T, Jeong B, Cho H (2010) A Petri Nets based functional validation for services composition. Expert Syst Appl 37:3768–3776

Zeng L, Benatallah B, Dumas M, Kalagnanam J, Sheng QZ (2003) Quality driven Web service composition. ACM WWW

Zhang D (2004) Web service composition for process management in E-business. J Comput Inf Syst 45(2):83–91

Chapter 48
Barriers to Client Collaboration in Agile Offshore Information Systems Development

Simon McGinnes

1 Introduction

As IT outsourcing becomes more popular, organisations are under pressure to reduce costs and improve outcomes by exporting more comprehensive and mission-critical IS development projects to IT firms. But IT outsourcing is risky. The high incidence of failure in outsourced IS development has prompted calls for measures to ensure success, including tighter contracts, more detailed advance requirements specification and more rigid change control (Tafti 2005). There is an assumption that outcomes will be improved by stronger enforcement of procurement practices and greater attention to contractual compliance, although it is not clear that evidence supports this conclusion.

Agile methods offer a potential alternative route to success and are being used in offshore IS projects. Many anecdotal success stories have been published, while a number of researchers have discussed modifications to agile methods intended to make them more suitable for outsourced projects (Sutherland et al. 2008; Sachdev and Iyengar 2007). But the predominance of advice about best practice in outsourcing points squarely away from agility and towards traditional top-down methods of project planning and management, as illustrated in Fig. 48.1 (Willcocks and Lacity 2006).

Outsourced government IT projects, which have a poor record of success, have tended to adopt particularly non-agile approaches (Ruzzier et al. 2008; Ewusi-Mensah 2003; NAO 2004). Overall, a consensus has yet to be reached about whether and how agility should be applied in outsourced IS development and, particularly, in offshore outsourced IS development. Many issues remain to be resolved, especially the problem of overcoming geographical and cultural separation between client and developer (Ramesh et al. 2006). Some well-justified

S. McGinnes
School of Computer Science and Statistics, Trinity College Dublin, Dublin, Ireland
e-mail: Simon.McGinnes@cs.tcd.ie

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_48,

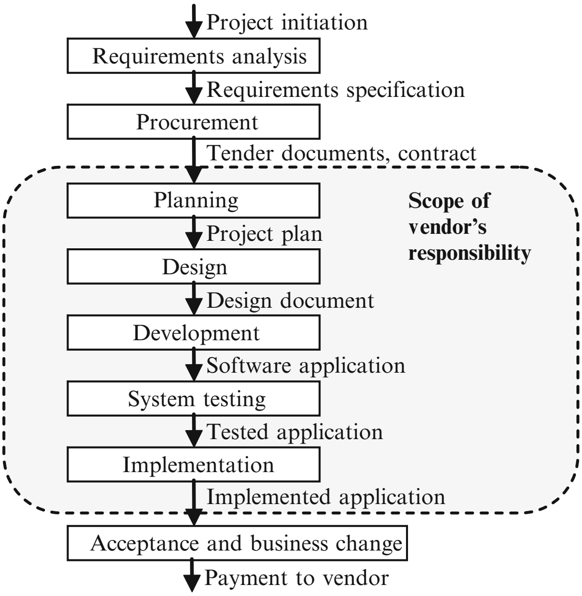


Fig. 48.1 Conventional project structure in IS outsourcing

theoretical basis is needed to help guide decisions, and this chapter is motivated by that need. The aim is to examine and explain the mismatch between agility and offshore outsourcing with regard to client collaboration. It is hoped that exposing the underlying issues about the conduct and structure of outsourced offshore IS development projects will allow them to be understood clearly so that further research can offer testable ways of addressing the client collaboration gap.

This chapter is organised as follows. The remainder of this section reviews prior research on agile methods in outsourcing and collaboration in IS development. Section. 2 examines the reasons why client collaboration can be particularly difficult in offshore IS development projects. Section 3 concludes with a summary of conclusions and suggestions for further work.

1.1 Background and Prior Research

Agile methods aim to increase development flexibility and speed without sacrificing quality. They emphasise collaboration in place of bureaucracy, documentation and top-down control. A range of development practices are used including timeboxing (DSDM), pair programming (XP) and sprints (SCRUM). Organisations wishing to use agile methods must relax their commitment to advance requirements specification; instead, the agile developer endeavours to meet business needs and priorities as they evolve during collaboration with the client (Sauer 2006).

Table 48.1 Interactions implicit in the waterfall life cycle (adapted for outsourcing)

Stage	Inputs/Outputs	Activity including interactions
Requirements	In: terms of reference Out: requirements doc.	Analyst interviews management and end users, who sign off requirements doc.
Procurement	In: requirements doc. Out: RFT; proposals; evaluation docs; contract	Client publishes RFT with requirements; vendors submit proposals; client selects one. Client and vendor agree a contract
Project planning	In: contract; proposal Out: Project plan	Client and vendor agree on a project plan based on proposal
System design	In: requirements doc. Out: design spec.	Designer produces a design specification to meet documented requirements
Development	In: design spec. Out: code	Developers write code to meet the design specification
System test	In: design spec.; code Out: test plan, cases, results; application	Testers test the code to ensure compliance with the design specification; developers correct defects
Implementation	In: application Out: implemented appl.	Implementation team installs the application ready for acceptance testing
Acceptance/ Business change	In: requirements doc. Out: acceptance; modified business processes	End users test application for compliance with documented requirements. Client adapts business processes accordingly

Much published research on agile methods in offshore IS development consists of experience reports and proposals about changes to agile practices and project structures intended to facilitate offshore outsourcing. Although claims are made about the benefits of agile methods, they have yet to transform practice universally, arguably because of structural and conceptual barriers. For example, a conflict exists between agility and the waterfall life cycle on which outsourcing contracts are often based (Fig. 48.1) (Gonzalez et al. 2010). Agile methods require extensive interaction between developer and client, which is difficult to achieve when the two are separated geographically (Sauer 2006). The waterfall life cycle is based on an implicit model of collaboration in which important interactions during development are largely restricted to formal communication (Laplante and Neill 2004). Table 48.1 summarises these interactions, most of which revolve around documents.

The waterfall is widely considered unrealistic because it reduces a complex process to a simple chain of events, "misrepresenting collective, interactive work as a linear, clear cut, and predictable workflow" (Ash et al. 2004). At root is a narrow view of collaboration; the developer is seen as someone who works mainly in isolation, translating specifications into software. According to this view, it should not matter where developers are located. But reality is more complicated; IS development relies on ad hoc, informal collaboration between developer and client. Formal interactions and handover of documents can be important, but when informal interaction is curtailed for one reason or another, problems can ensue.

Although the waterfall life cycle has been criticised, for these and other reasons, it remains the basis for many project plans and is entrenched in management thinking. It is also fundamental to public procurement processes (Laplante and Neill 2004). As recently as 2004, the UK government introduced revised contracts for its major IT outsourcing arrangements based on the waterfall model, with poor results (Ballard 2011).

In the context of IS development, completely unhindered communication is rarely ideal. If developers must frequently deal with client enquiries, productivity can suffer. Project managers who must devote time reporting to their own managers have less time to manage projects. Analysts with demanding end users may feel obliged to over-specify requirements. In each of these situations, there is an optimal level of interaction beyond which negative outcomes may ensue.

Nevertheless, any realistic model of IS development must account for a significant proportion of time spent by developers in informal collaboration with clients and colleagues. Much of this informal interaction may produce no concrete deliverables or documentation, yet it is nonetheless important. There is a risk that this critical form of communication will be off the radar to management and therefore omitted from project plans and quality metrics. Key decisions may proceed regardless of the quality of informal communication. Arguably, project failures have arisen as a consequence of this invisible gap in communication (Sauer 2006).

When IT professionals create IS, they must understand the organisation and its processes and context well. Few employees need such breadth and depth of organisational knowledge. The developer must be able to visualise the future organisation as it will exist once planned IS have been implemented. Divergent end user needs and aspirations must be reconciled in a coherent and useful way. This requires an extraordinary level of insight, upon which successful development relies, although that fact may be unacknowledged by formal processes (Hanisch et al. 2004).

Gaps in this organisational knowledge can translate into system defects, yet much organisational knowledge is acquired only informally or incidentally. For instance, a good deal of discussion normally occurs during requirements gathering, but most is never formally documented. Instead, it becomes part of the analyst's tacit knowledge: background or context which is needed to make sense of documented requirements. Developers who need to understand requirements properly must acquire the same kind of tacit knowledge. Other useful tacit knowledge about the organisation and context is acquired in the course of various tasks in the development process.

Expecting this informal knowledge to be encapsulated in documentation is unrealistic; it is too rich and interconnected. Developers must become immersed in the organisation, one way or another, to gain sufficient contextual understanding. This requires informal interaction (Desouza 2003). When such interaction is blocked by outsourcing arrangements (including contractual provisions or geographical separation, for example), problems may be expected.

2 Impact of Outsourcing on Client Collaboration

Success in IS outsourcing is subjective: a project is successful if the stakeholders (particularly, the client) believe it to be so. Subtle and often unstated factors such as expectations and perceptions are important. Above all, it is critical to have good communication and collaboration between vendor and client. Yet collaboration is particularly at risk in offshore projects; barriers to collaboration include contractual constraints, physical separation and possibly linguistic and cultural differences (Sachdev and Iyengar 2007). The following sections examine the impact of these factors on client collaboration and suggest reasons why particular management choices may reduce the chance of success.

2.1 Effect of Team Location on Client Collaboration

Because of physical separation between client and developer, outsourcing naturally favours indirect and formal communication over face-to-face and informal interaction. The informal communication gap can to some extent be addressed using technologies such as email, chat, telephone and videoconferencing. Collaboration between developers who are geographically separated can also be facilitated by suitable development tools (Navoraphan et al. 2006). However, for interaction between developer and client, face-to-face communication remains indispensable (Fraser et al. 2005). For this reason, many vendors choose to work in person with their clients for at least the most important interactions such as project planning and requirements gathering (Krishna et al. 2004).

Nevertheless, permanent collocation with the client is rarely possible, and decisions must be made about where to site developers. This is not a trivial question. Outsourcing may be seen as an innocuous change that will not affect software practices (Avison et al. 2006), but development relies heavily on informal communication and the tacit knowledge that it brings. Care is necessary in deciding how to apportion tasks to different locations since many forms of collaboration may be required. To illustrate, Fig. 48.2 quantifies the number of possible collaborative groupings (g) in project teams (including client and developers) of increasing size (n). The number of potential collaborative groupings increases much faster than team size.

In a homogeneous situation, all roles interact equally, and so it is best to collocate. But, in real projects, collaboration is uneven and optimal groupings of roles may emerge, where intragroup collaboration outweighs intergroup collaboration. Such groups may then be separated geographically with less risk. This decision can be supported by examining the type, frequency and importance of the collaborations. Figure 48.3 illustrates the most important interactions in an offshore project. Analyst, designer and developer are shown as distinct roles; although in practice, they may be performed by the same person.

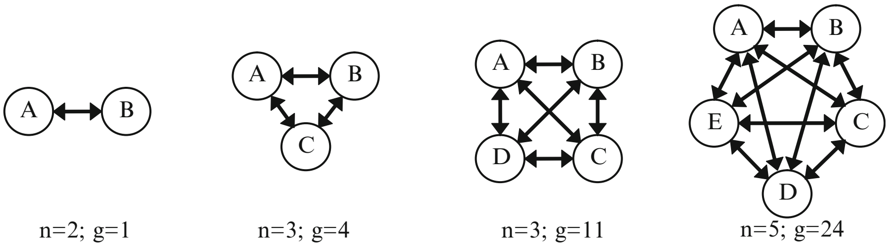


Fig. 48.2 Potential interactions between project participants

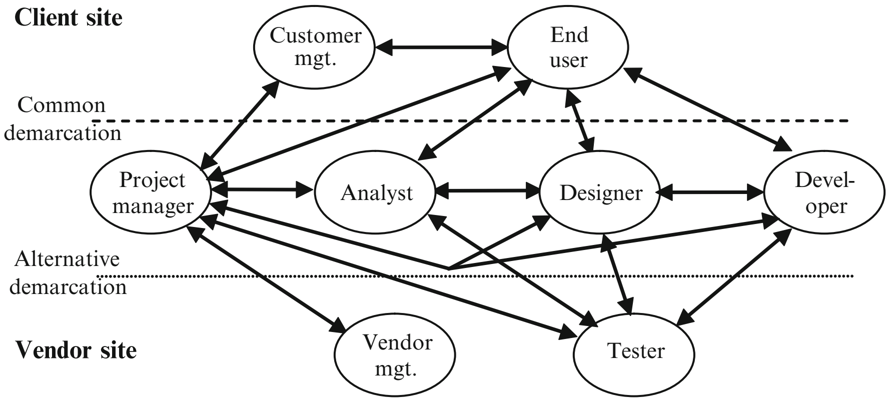


Fig. 48.3 Placement of team roles in an offshore project

Figure 48.3 makes the problem of how to distribute work more apparent. One common demarcation line between client and vendor (shown with a dashed line) is at precisely the point where communication barriers such as language and cultural differences are likely to be greatest. Each team member wishing to collaborate with the client must do so electronically or be prepared to travel, neither of which is ideal. An alternative demarcation (shown with a dotted line) removes this constraint but interrupts collaboration with offshore team members and effectively requires the bulk of the team relocate to the client's site.

Both of these options have obvious flaws. Figure 48.4 shows another model which is sometimes adopted in offshore projects when non-agile methods are used. Most developers are geographically separated from the client. The analyst is part of a "beachhead" team sent to the client's site to gather requirements, or a client employee, or a third-party consultant (as in public procurement where transparency rules may insist on separation of roles). This model has four main collaborative arcs: (a) face-to-face interaction between analyst and user, to establish requirements; (b) indirect communication between analyst and designer via a requirements statement; (c) face-to-face collaboration between designer and developer in system construction; and (d) electronic delivery of the software applications to the client.

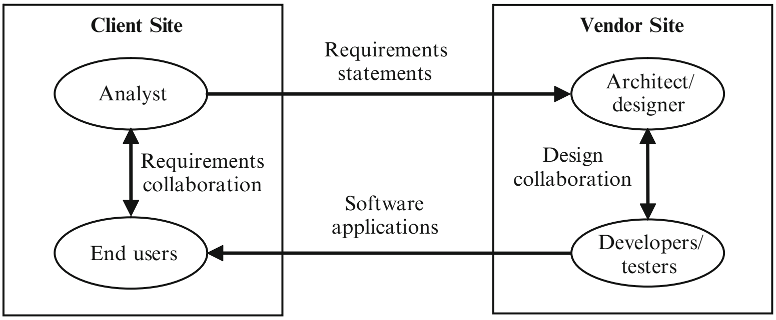


Fig. 48.4 Common grouping in non-agile offshore development

This model is in fact similar to some offshore agile development models which have been proposed, although they are often more complex, with additional roles and multiple teams (Roussev and Akella 2006). Its chief strength is that analyst and client can collaborate closely. However, it allows primarily for traditional development; key agile practices cannot easily be used because requirements must be communicated in document form to the offshore development team and developers cannot work closely with the client. Therefore, important benefits of agility are not available.

2.2 *Effect of Communication Barriers on Client Collaboration*

In an offshore project, client and developer may be separated by a range of factors such as geography, culture, training and socio-economic status. Unsurprisingly, communication difficulties are common (Kliem 2004). Figure 48.5 illustrates the potential communication barriers in offshore projects, expressed as a series of "layers".

The first barrier (layer A) concerns the well-known impact of differences in IS projects between business-oriented end users, liable to employ their own business-related jargon, and IT specialists who unthinkingly resort to "technobabble" (Jeyaraj 2005). Needless to say, the resulting miscommunications can lead to systems that do not reflect real business needs, irrespective of outsourcing.

When IT work is distributed geographically, communication is further limited by physical separation (layer B). Technological props such as teleconferencing and videoconferencing can help but are not an adequate substitute for face-to-face communication and physical proximity. When face-to-face interaction is replaced by formal reports, emails, telephone conversations and the like, important but subtle and unstated messages can easily be lost. For example, key information may lie buried in the detail of a report or obscured by poor writing. Aural cues such as inflection and emphasis are absent from email and text messages. Visual cues such as facial expression and posture are invisible in voice-only communications. A

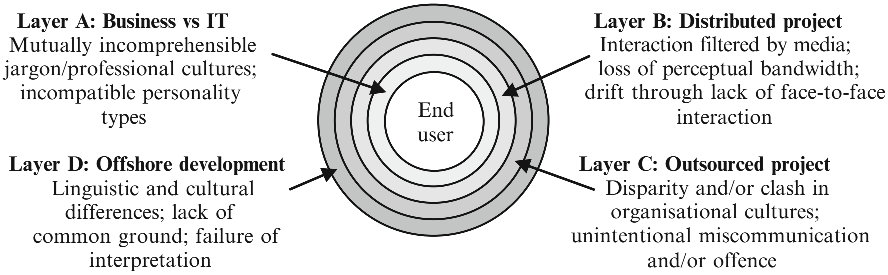


Fig. 48.5 Communication layers in outsourced projects

shrug, a raised eyebrow, folded arms, tone of voice and even silence can all convey crucial information that is easily lost in electronically mediated communication (Hinde 1972). Some IT professionals, particularly those trained as facilitators, are sensitised to these issues, but the majority are not (Huang and Trauth 2006).

If work is also outsourced, translation becomes necessary between divergent organisational cultures, norms, customs and implicit beliefs (layer C). Different organisations can have very different assumptions about work and business. For example, there are typically significant differences between a sales-driven, fast-moving international IT consultancy (vendor) and a procedure-bound, risk-averse, slow-moving government department (client). This disparity hinders collaboration because communications are unconsciously filtered through conflicting worldviews and mismatched mental models (Hill and Levenhagen 1995). A common manifestation of this problem is the ease with which language in contracts and requirements can be interpreted in conflicting ways by vendor and client, without it (or the consequent risk) being obvious to either party (Krishna et al. 2004). In extreme cases, developers may even be barred from liaising directly with the client, as in the case where a vendor seeks to limit information flow to protect commercial interests (Newell et al. 2007).

If the outsourced work is placed offshore, further communication difficulties arise (layer D). First, it may be necessary to deal with different languages, which creates an administrative burden, slows the flow of communication and may introduce miscommunication. Nuances expressed in requirements and designs can easily be lost in translation (Macnamara 2004). Even if vendor and client share a language, communication issues are still possible owing to variations in dialect and cultural norms. In one project, Irish and US developers miscommunicated despite sharing a language and ostensibly similar cultural backgrounds. Specific words were interpreted differently with detrimental consequences. Cultural differences in working practices emerged, with one group making promises that they did not expect to be held to, and the other group assuming that commitments would be honoured. Similar experiences have been reported repeatedly in offshore projects (Summers 2008).

The effect of cultural differences on communication is a very real issue which has been addressed extensively (Macnamara 2004). However, the risk is easily overlooked by IT practitioners. One's own culture is effectively transparent; IT specialists are rarely trained in cultural sensitivity and not particularly renowned for their empathetic skills. There have been urgent calls for curriculum reform in IS courses to incorporate cultural awareness and cross-cultural communication (Gorgone et al. 2006).

To conclude, several forms of translation may be needed in offshore IS development. They can be thought of as layers of opacity that combine to obscure messages (Macnamara 2004). We have used the example of communication about business needs since miscommunication of requirements is a perennial source of difficulty in IS projects (Duggan and Thachenkary 2004). However, similar effects apply to communication for all purposes.

While a range of measures can be taken to address this problem, linguistic and intercultural translation must be performed by a suitably bilingual or bicultural individual. For example, if software requirements need to be interpreted by a developer in a different cultural context from the one they were written in, then the developer must be familiar with both contexts (Jeyaraj 2005). This applies to translation between business and IT contexts, and between organisational cultures, as much as it applies between different national cultures and languages. This need for translation places a significant constraint on the allocation of team members because of the additional skills needed.

2.3 Effect of Contractual Provisions on Client Collaboration

Another problem in IS outsourcing concerns the desire to avoid risk by fixing delivery dates, prices and requirements scope. This is common practice in public procurement, where vendors bid to provide services at fixed cost. It stems from the need for accountability; budget holders must be able to demonstrate value and transparency when spending public money and awarding contracts. Therefore, organisations seek to constrain the minimum deliverable that a vendor will provide by fixing scope, cost and timescales in advance, so as to avoid later disputes (Ambler and Constantine 2000).

However, there is an extensive literature arguing against this way of organising IS development, including research on agile methods (Abrahamsson et al. 2005). For example, the practice of basing contracts on detailed requirements specifications has been opposed for a variety of reasons. It can be effectively impossible to write definitive requirements for non-trivial systems, no matter how much detail is included (Holtzblatt and Beyer 1995); more detail tends to defeat clarity by making specifications impenetrable, complex and inconsistent. Clients often do not know what they need or want and, when pushed to state requirements, may specify or agree to requirements that turn out to be incorrect or infeasible. Every specification is open to interpretation; vendors and clients have conflicting goals and will (subconsciously or otherwise) interpret specifications in self-serving ways, especially when there are financial implications. In offshore development, there is every possibility that requirements will be misinterpreted because of the offshore team's lack of tacit knowledge about the client organisation and its business context. Developers often complain that end users "change their minds" as a project progresses, but the act of developing an IS literally changes the mind of everyone involved, including end users and developers, colouring their interpretation of specifications and rendering documented requirements obsolete. The cost implications of any requirement cannot be known until the requirement has been met; therefore, work to meet requirements will always be an exercise in juggling limited resources. Requirements will be prioritised, altering the makeup of the resulting IS and creating knock-on effects in terms of feasibility and business impact.

Consequently, outsourced IS projects which rely on advance requirements specification often suffer from misunderstandings about requirements, scope creep and related issues. These are problems in any project, but in offshore work they place the all-important vendor-client relationship under great stress. There is no simple solution since a matter of principle is involved: when entering into fixed-price, fixed-scope, fixed-duration contracts, both parties effectively agree that there is a single, definitive interpretation of the requirements specification. Developers may therefore tacitly acknowledge shifts in requirements but be officially unable to act on them, since they are contractually obliged to honour specifications that they know to be obsolete or incorrect. Renegotiation of contract terms is necessary so often in outsourced IS projects that it may well be the norm (NAO 2004).

More enlightened procurement practice recognises the need for flexibility in requirements, interaction and iteration in software development, as promoted by agile methods (Cottmeyer 2008). Trust is critical; many appropriate alternative contract forms can be used, according to circumstance, including time and materials, cost plus and shared risk/reward (Gopal et al. 2003). However, these contract forms remain the exception rather than the norm, not least because procurement practices are entrenched and, in the case of much public procurement, enshrined in legislation (Beuter 2005).

3 Conclusion

Good informal communication and collaboration between developer and client is a key prerequisite for success in all IS development and is essential in agile development (Lee et al. 2006). But outsourcing complicates IS development; it alters the shape of development teams and changes the ways in which they collaborate with each other and, crucially, with their clients. Offshore outsourcing, in particular, reduces the available opportunities for effective client collaboration. This research has argued that miscommunication and other collaboration issues can arise as a consequence of the ways in which offshore IS projects are organised. Therefore it is important in offshore IS development projects to effectively address barriers to client collaboration, especially if the benefits of agility are desired. Three major barriers are identified: physical separation, obstacles to communication caused by cultural and other differences, and unhelpful contractual provisions.

Given the present momentum of the offshore outsourcing trend, further research is urgently needed on ways of effectively addressing the client collaboration gap in offshore IS development. This is likely to take the form of proposed models and practices, but it should also include rigorous empirical research designed to test alternative ways of organising agile offshore outsourced IS development projects, as an alternative to the present emphasis on anecdotal experience reporting.

References

Abrahamsson P, Salo O, Ronkainen J, Warsta J (2005) Agile software development methods: review and analysis. VTT Publications, Espoo

Ambler SW, Constantine LL (2000) The unified process inception phase. CMP Books, Lawrence

Ash JS, Berg M, Coiera E (2004) Some unintended consequences of information technology in health care: the nature of patient care information system-related errors. J Am Med Inform Assoc 11(2):104

Avison D, Gregor S, Wilson D (2006) Managerial IT unconsciousness. Commun ACM 49 (7):88–93

Ballard M (2011) Gov IT procurement talks enter third year. Available at: http://www.channelregister.co.uk/2006/12/08/intellect_ogc_still/, Accessed 4 May 2011

Beuter R (2005) European public procurement reform: main innovations in the public sector directive – a preliminary assessment. EIPAscope 2005(3):5–11

Cottmeyer M (2008) The good and bad of agile offshore development. In: Proceedings of the agile 2008 conference. IEEE, 2008

Desouza KC (2003) Facilitating tacit knowledge exchange. Commun ACM 46(6):85–88

Duggan EW, Thachenkary CS (2004) Integrating nominal group technique and joint application development for improved systems requirements determination. Inf Manage 41(4):399–411

Ewusi-Mensah K (2003) Software development failures. MIT Press, Cambridge

Fraser S, Martin A, Adams M, Chilley C, Hussman D, Poppendieck M, Striebeck M (2005) Off-Shore agile software development, Extreme programming and agile processes in software engineering, pp 267–272

Gonzalez R, Gasco J, Llopis J (2010) Information systems outsourcing reasons and risks: a new assessment. Ind Manage Data Syst 110(2):284–303

Gopal A, Sivaramakrishnan K, Krishnan MS, Mukhopadhyay T (2003) Contracts in offshore software development: an empirical analysis. Manage Sci 49(12):1671–1683

Gorgone JT, Gray P, Stohr EA, Valacich JS, Wigand RT (2006) MSIS 2006: model curriculum and guidelines for graduate degree programs in information systems. ACM SIGCSE Bull 38 (2):121–196

Hanisch J, Campus CW, Terrace N (2004) Requirements engineering during global software development: some impediments to the requirements engineering process–a case study. In: Proceedings of the 12th European conference on information systems, pp 14–16

Hill RC, Levenhagen M (1995) Metaphors and mental models: sensemaking and sensegiving in innovative and entrepreneurial activities. J Manage 21(6):1057

Hinde RA (1972) Non-verbal communication. Cambridge University Press, Cambridge

Holtzblatt K, Beyer HR (1995) Requirements gathering: the human factor. Commun ACM 38 (5):31–32

Huang H, Trauth EM (2006) Cultural diversity challenges: issues for managing globally distributed knowledge workers in software development, managing IT professionals in the internet age. Idea Group, Inc, Hershey

Jeyaraj J (2005) Technical communication and cross cultural miscommunication: usability and the outsourcing of writing. Professional communication conference, IPCC 2005. Proceedings, International, pp 92–100

Kliem R (2004) Managing the risks of offshore IT development projects. Inf Syst Manage 21 (3):22–27

Krishna S, Sahay S, Walsham G (2004) Managing cross-cultural issues in global software outsourcing. Commun ACM 47(4):62–66

Laplante PA, Neill CJ (2004) Opinion: "The demise of the waterfall model is imminent" and other urban myths. Queue 1(10):10–15

Lee G, DeLone W, Espinosa JA (2006) Ambidextrous coping strategies in globally distributed software development projects. Commun ACM 49(10):35–40

Macnamara JR (2004) The crucial role of research in multicultural and cross-cultural communication. J Commun Manage 8(3):322–334

NAO (2004) Improving IT procurement: the impact of the Office of Government Commerce's initiatives on departments and suppliers in the delivery of major IT-enabled projects, HC 877 Session 2003–2004, 5 Nov 2004. National Audit Office, London

Navoraphan K, Gehringer EF, Culp J, Gyllstrom K, Stotts D (2006) Next-generation DPP with Sangam and Facetop. In: Proceedings of the 2006 OOPSLA workshop on eclipse technology eXchange, pp 6–10

Newell S, David G, Chand D (2007) Exploring trust among globally distributed work teams. System sciences, 2007. HICSS 2007. 40th annual Hawaii international conference on system sciences, pp 246c–246c

Ramesh B, Cao L, Mohan K, Xu P (2006) Can distributed software development be agile? Commun ACM 49(10):41–46

Roussev B, Akella R (2006) Agile outsourcing to India: structure and management. Management of the object-oriented development process, p 109

Ruzzier J, Sohal AS, Katna P, Zyngier S (2008) Success and failure in IT outsourcing by government agencies: two Australian case studies. Int J Bus Inf Syst 3(2):107–119

Sachdev V, Iyengar K (2007) Will agile methodologies work in offshore outsourcing. SWDSI07 San Diego, USA

Sauer J (2006) Agile practices in offshore outsourcing–an analysis of published experiences. In: Proceedings of the 29th information systems research seminar in Scandinavia, IRIS

Summers M (2008) Insights into an agile adventure with offshore partners. In: Proceedings of the agile 2008 conference. IEEE, 2008

Sutherland J, Schoonheim G, Rustenburg E, Rijk M (2008) Fully distributed scrum: the secret sauce for hyperproductive offshored development teams. In: Proceedings of the agile 2008. IEEE, 2008

Tafti MHA (2005) Risks factors associated with offshore IT outsourcing. Ind Manage Data Syst 105(5):549–560

Willcocks LP, Lacity MC (2006) Global sourcing of business and IT services. Palgrave Macmillan, Basingstoke

Chapter 49
On Data, Information, and Knowledge Representation in Business Process Models

Ligita Businska, Inese Supulniece, and Marite Kirikova

1 Introduction

Data, information, and knowledge are terms that are widely used in various fields of human activities and have caused discussions about their meaning since ancient times. Although there are various research works on scientific theories of data, information, and knowledge interpretation, e.g., in psychology, epistemology, social science, philosophy, cognitive science, and information theory, these terms still are used intuitively and often have no explicit unified definition even within one area of research.

This chapter shows that ambiguity exists concerning above-mentioned concepts as well as their use in several business process modeling notations (Expert Paper 2007; ARIS Method; ARIS community; IDEF3; IDEF0; BPMN; UML; Activity diagrams in UML 2.0; Business Modeling 1998; GRADE user guider; Arbeitsbericht; Heisig 2006; Woitsch and Karagiannis 2005; Weidong and Weihui 2008). This chapter presents experiments with the notations most frequently mentioned in scientific literature showing that by them data and information objects are not differentiated and often data, information, and material flow are represented by the same artifacts. The goal of this chapter is not to provide explicit and unified definitions. It strives analytically and experimentally to compare the most frequently studied business process modeling languages/notations (namely, extended event-driven process chain (eEPC) in ARIS tool (Expert Paper 2007; ARIS Method; ARIS community), Integration Definition for Function Modeling (IDEF0) (IDEF3; IDEF0), business process modeling notation (BPMN 2.0) (BPMN), Unified Modeling Language (UML 2.0) activity diagram (UML; Activity diagrams in UML 2.0), and business modeling language GRAP;ES-BM in GRADE

L. Businska (✉) • I. Supulniece • M. Kirikova
Riga Technical University, 1/4 Meza, Riga, Latvia
e-mail: ligita.businska@cs.rtu.lv; inese.supulniece@rtu.lv; marite.kirikova@cs.rtu.lv

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_49,

tool (Business Modeling 1998; GRADE user guider)) from the point of view data, information, and knowledge.

Additionally to business process modeling notations, this chapter covers notations for modeling of knowledge intensive processes where basic phenomenon is knowledge. In these processes, the principal success factor is adequate modeling of knowledge conversions. At the same time approaches that focus on knowledge management within the business process level have limited capabilities. Usually knowledge is modeled using specific knowledge modeling notations (e.g., KMDL (Arbeitsbericht), GPO-WM (Heisig 2006)), and only few of them include process perspective (e.g., PROMOTE (Woitsch and Karagiannis 2005), RAD (Weidong and Weihui 2008)).

Knowledge might be considered as one of the business process dimensions, because knowledge is created as a result of process execution, knowledge is used to perform a process, and it is distributed among process participants (Heisig 2006). Benefits provided by extending business process models with knowledge dimension are described in Supulniece et al. (2010). This chapter considers the Knowledge Modeling and Description Language (KMDL 2.2) which has widest knowledge modeling capabilities among the process-oriented knowledge modeling languages (Supulniece et al. 2010).

The rest of this chapter is structured as follows – Sect. 2 briefly describes data, information, and knowledge representation within selected modeling notations. Section 3 illustrates differences and similarities in data, information, and knowledge dimension by presenting the deposit contract conclusion process in selected modeling notations. This chapter concludes with Sect. 4, where preliminary research results and further research are discussed.

2 Data, Information, and Knowledge Modeling

The existence of an organization is achieved by the coordination of member activities because members without the coordination are individuals rather than an organization. Coordination of BP activities within an organization is achieved with information exchange between organization members. This is the basis for knowledge generation and distribution. Information is usually defined as interpreted data. In the remainder of this chapter, we consider data as facts about the universe of discourse. Knowledge then is the model of the universe of discourse which is used for interpretation of data. However, in cases where particular business process modeling languages are discussed, we stick to their "native" interpretations of data, information, and knowledge (if such interpretations are available).

Business process modeling notations support data and information inclusion into business process models. For example, a data element is presented in various notations (Expert Paper 2007; ARIS Method; ARIS community; IDEF3; IDEF0; BPMN; UML; Activity diagrams in UML 2.0; Business Modeling 1998; GRADE user guider); however, interpretation of this element differs per notation.

Additionally notations do not separate accurately data from information, allowing the use of the same symbols for both concepts. Usually knowledge is modeled by means of specific knowledge modeling notations (e.g., KMDL (Arbeitsbericht), GPO-WM (Heisig 2006)), and only a few of them include process perspective (e.g., PROMOTE (Woitsch & Karagiannis 2005), RAD (Weidong and Weihui 2008)). These BP-oriented knowledge management methods focus on storing and sharing knowledge. As a result, they lack the ability to model in an adequate manner the decisions, actions, and measures, which are causing a sequence of processes. Most of these methods are convenient only for knowledge management experts and require additional training for nonexperts (Supulniece et al. 2010).

In this section, we will give brief overview of data, information, and knowledge concept representation and interpretation of the following notations: GRAPES-BM in GRADE tool (Business Modeling 1998; GRADE user guider), eEPC diagrams in ARIS (Expert Paper 2007; ARIS Method; ARIS community), KMDL (Arbeitsbericht), IDEF0 (IDEF0), UML 2.0 activity graphs (UML), and BPMN 2.0 (BPMN).

2.1 Data Representation

BPMN – BPMN is constrained to support only the concepts of modeling that are applicable to business processes; therefore, data and information models are out of the scope of its specification (BPMN). While BPMN shows the flow of data (messages) and the association of data artifacts to activities, it is not a data flow language. To model the items (physical or information items) that are created, manipulated, and used during the execution of a process, the following constructs are used: data objects, messages, and data associations (Fig. 49.1a–c). Data is represented with the following four elements: data objects, data inputs, data outputs, and data stores (Fig. 49.1a). *ARIS eEPC* – ARIS eEPC notation allows multiple ways to show data in a process model (Expert Paper 2007): either as abstract clusters or technical terms that describe information, or depicted as information carriers, such as documents and CDs. ARIS eEPC does not explicitly separate data and information; thus, the same objects could be used to represent both concepts. A document reflects real information holders (Fig. 49.2a), but a cluster is data collection and is used to represent the data model (Fig. 49.2b).

GRAPES-BM – The graphical representation of a business process diagram in GRADE (Business Modeling 1998; GRADE user guider) is similar to a flowchart. It is possible to specify data or information items on the event arrows that connect the tasks of the process, thus showing data and information flows. Besides, there are two special symbols to represent the use of data in BP diagram: a data store and a data object (Fig. 49.3). A data store symbol indicates a persistent store of data or materials. A typical use of the data store is to represent a related database. On the other hand, data stores may simply represent a stock of goods. A data object represents just one unit. It is a temporary store of information or materials.

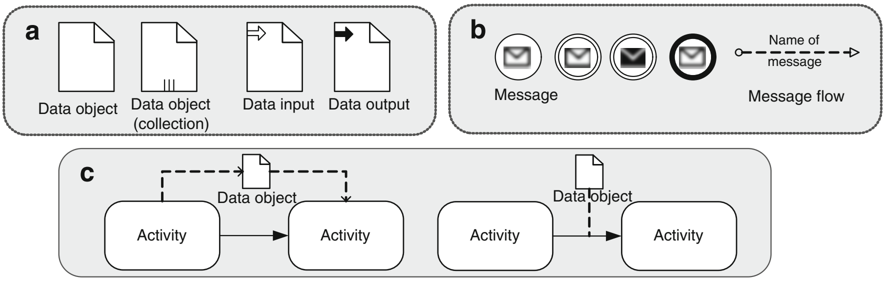


Fig. 49.1 BPMN elements for data and data flow modeling (**a**) Data objects, (**b**) Messages, (**c**) Data associations

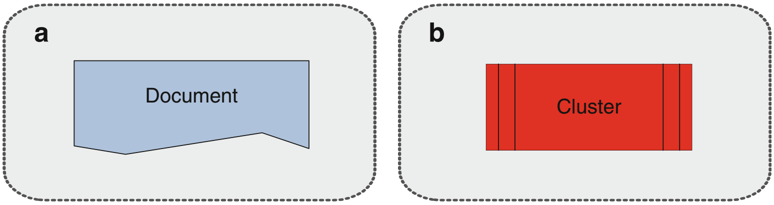


Fig. 49.2 ARIS eEPC elements for data and data flow modeling (**a**) Document, (**b**) Cluster of data

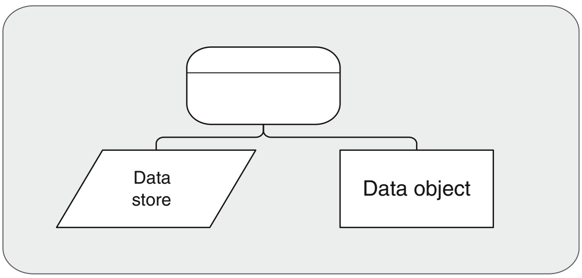


Fig. 49.3 GRAPES-BM elements for data and data flow modeling (**a**)A: Data store and data object, (**b**) Data and information flows

UML2 Activity Diagram (UML) – *UML2 activity diagram* is used for business process modeling, for modeling the logic captured by a single use case or usage scenario, or for modeling the detailed logic of a business rule. It is based on data flow models – a graphical representation of how data move around an information system (Activity diagrams in UML 2.0).

The specification of this notation defines only a data store element. A data store node is the central buffer node for non-transient information. The data store notation is a special case of the object node notation. See Fig. 49.4 for example.

IDEF0 – There are several methods under IDEF family, e.g., IDEF0 for function modeling for process modeling (IDEF0). The IDEF0 does not separate data and

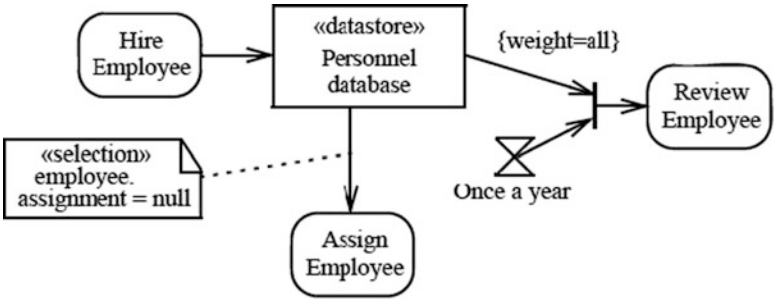


Fig. 49.4 Data store node example (Activity diagrams in UML 2.0)

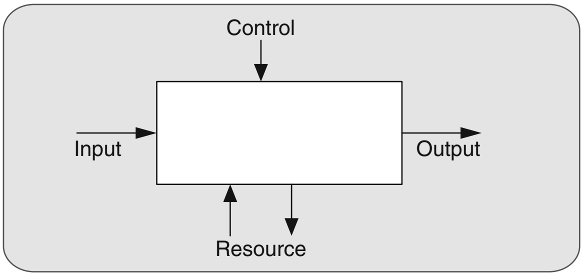


Fig. 49.5 IDEF input/output arrows for data flow modeling

information. These concepts could be modeled by one and the same symbol – an arrow: input arrow and output arrow (Fig. 49.5). Arrows convey data or objects related to functions to be performed.

2.2 *Information Representation*

BPMN – Information does not have special element in BPMN, although message, message flow, conversation, and choreography refer to information and information flow (BPMN). Message is an object which depicts the contents of communication between two participants, which is transmitted through message flow (see Fig. 49.1b). Message flow is a connecting object that shows the flow of messages between two participants. A conversation (Fig. 49.6a) represents a set of message flows grouped together based on a concept and/or a correlation key. A conversation involves two or more participants. Choreography is a type of process, which differs in purpose and behavior from a standard BPMN process (Fig. 49.6b). Choreography formalizes the way business participants coordinate their interactions. The focus is not on orchestration of the work performed by these participants, but rather on the exchange of information (messages) between these participants. Choreography is a

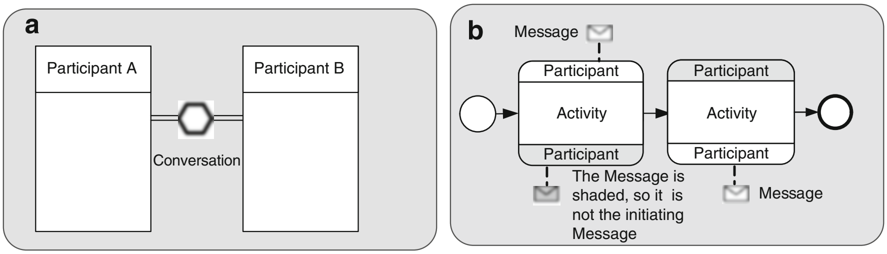


Fig. 49.6 Example of additional process types in BPMN (**a**) Collaboration, (**b**) Choreography

definition of expected behavior; it brings into view message exchanges and their logical relation as conversations.

GRAPES-BM – Information does not have a specific symbol; however, it is possible to specify information items on the event and path arrows that connect task, event, and decision nodes. Additionally events with the category "message" corresponding to objects produced by one task and transmitted to another (Business Modeling 1998; GRADE user guider) could be considered as information object. This may be an informal object, such as a notice that something has been completed, or a formal one, such as an invoice and bill, when the information is formally associated with the event via a corresponding data type.

UML2 Activity Diagram (UML; Activity diagrams in UML 2.0), *ARIS eEPC* (Expert Paper 2007; ARIS Method; ARIS community), and *IDEF0* (IDEF0) – These diagrams can be used for modeling information flows in business processes; however, the original notations do not provide an explicit definition for the information concept. Moreover, UML activity diagram is more expressive for modeling data flows inside information system and is less suitable for business process modeling.

KMDL – This language is not primarily intended for information flow representation, yet it includes a special symbol for information. Information is modeled as information object, which usually exists as a text, drawing, or diagram on paper or electronic form (e.g., a document, audio file, and bitmap). Information exists outside the people and can include explicit knowledge of people (Arbeitsbericht). Information objects can be in the form of an input or output object of conversion.

2.3 Knowledge Representation

ARIS eEPC – Knowledge is represented by two object types, knowledge category and documented knowledge, and can be model by two model types, knowledge structure diagram and knowledge map (ARIS Method). The object type knowledge category is an object with its content referring to specific knowledge (both implicit and explicit), such as project management knowledge, specific industry knowledge, and specific technology knowledge (Fig. 49.7a). The documented knowledge object

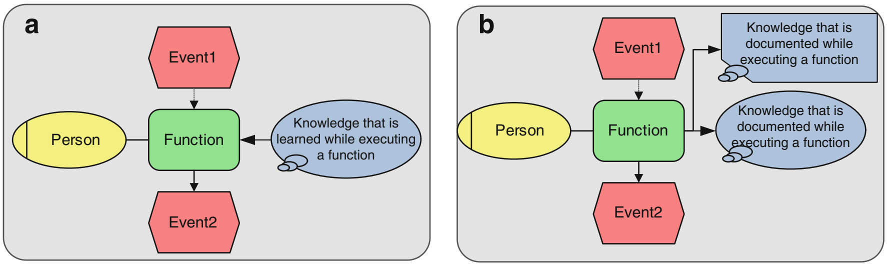


Fig. 49.7 Knowledge representation in ARIS eEPC (**a**) Knowledge category, (**b**) Documented knowledge

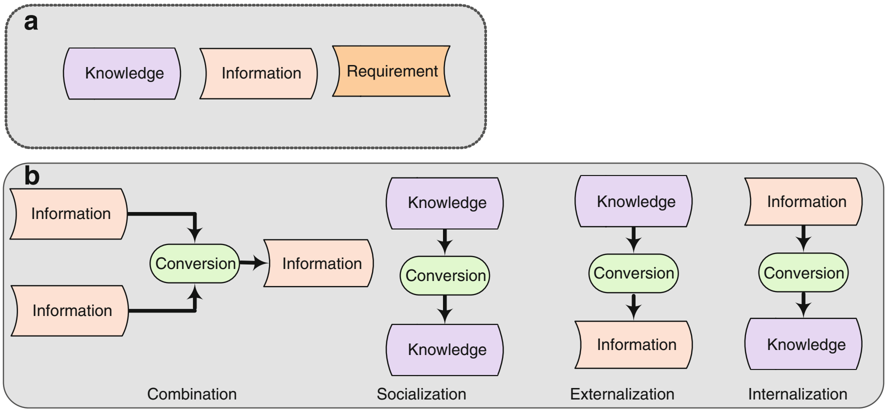


Fig. 49.8 Knowledge representation in KMDL (**a**) Knowledge and information objects, (**b**) possible types of conversions

type deals with knowledge which is explicitly documented or at least can be documented in principle (Fig. 49.7b). The knowledge structure diagram knowledge categories can be organized into subgroups based on their content. A knowledge map depicts the distribution of various knowledge categories within an organization.

KMDL (Arbeitsbericht) – In KMDL 2.2, there is a basic distinction between three views: (1) process-based view, (2) activity-based view, and (3) communication-based view.

In the first process-based view, tasks are combined in a sequence of actions. For every task roles and the used information systems can be specified.

A task in the process-based view can consist of multiple knowledge conversions that could be specified in the second activity-based view. Start and end objects of conversions can be both information objects and knowledge objects (Fig. 49.8b). Knowledge objects can be related to persons, teams, or undefined persons. Additionally, it is possible to define knowledge as requirements for appropriate task (Fig. 49.8a), namely, functional, methodical, social, or technical requirements. The technical requirements can be covered by functions of information systems. The coverage of the remaining requirements is ensured by – also differentiated – knowledge objects of persons/teams. Beside the types of requirements, it is possible to differentiate between mandatory and optional requirements.

Finally, the third communication-based view provides the basis to model communication within an organization. Knowledge objects are artifacts which refer to tacit knowledge of the person, and they might be used for creation of the skill catalogue.

Within KMDL, it is possible to model only tacit knowledge which belongs to certain person or group. Knowledge is anchored in the activities and skills of the knowledge carrier and additionally in her/his ideals, values, and experiences. In contrast, explicit knowledge could be modeled with information objects that are attached to certain conversion in business process model.

3 Comparisons of the Modeling Notations

In this section, three notations are selected for comparison (BPMN, ARIS eEPC, and KMDL) because taken together they represent almost all aspects of data, information, and knowledge modeling discussed in the previous section. To illustrate the described differences of data, information, and knowledge representation, a simplified business process example from the banking industry is used. The presented business process does not reflect all details and flow alternatives as this is not the aim of this chapter. Business process starts when client contacts the bank with the aim to open a deposit account. Bank employee presents to the client available products/services and their conditions. Based on this information, client chooses fixed-term deposit (also known as a term deposit, a bond, or fixed deposit in other countries). Selection of other deposit products is not modeled. Then the bank employee presents possible “terms” (period of time) and respective interest rate offers, and the client selects the one. The bank employee asks if the client has the bank account. The process branch for the client without account is not modeled to keep example as small as possible. If the client has the account, then the employee starts identification process to perform manipulations with client’s account. For identification, the employee ask the client’s document, e.g., passport, ID, or similar document. Then he/she enters some client data in the information system to check validity of the document if the client has rights to manipulate with the account and other information according to internal regulations. When the client is successfully identified, the bank employee prepares the contract in the information system and asks the client to sign it. Simplified business process example ends when the client signs the contract.

To illustrate data, information, and knowledge interpretation in BPMN notation, three diagrams were created for the described business process – a basic processes view (Fig. 49.9), choreography model (Fig. 49.10), and collaboration model (Fig. 49.11).

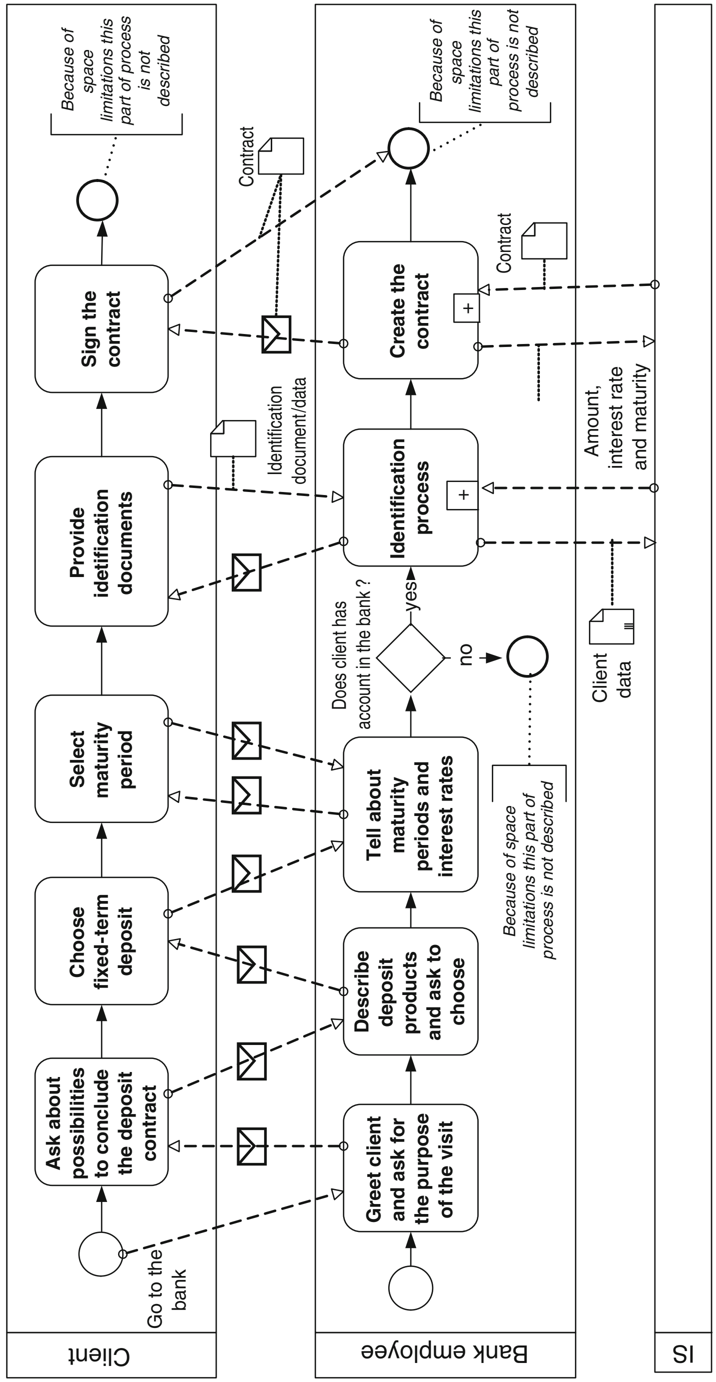


Fig. 49.9 Example of data and information flow modeling in BPMN

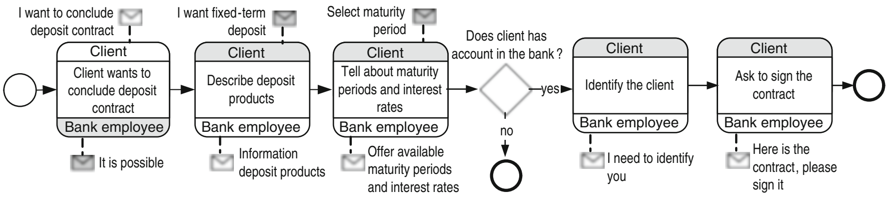


Fig. 49.10 Example of choreography modeling in BPMN

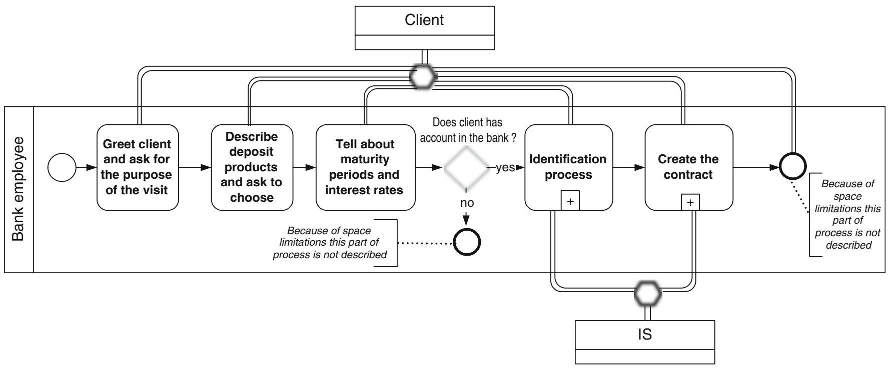


Fig. 49.11 Example of collaboration modeling in BPMN

BPMN stresses the process view representation, offering a number of symbols for modeling various processes and event types. For these tasks, BPMN is one of the most suitable modeling notations. However, data and information flow representation is considered secondary. As a result, the developed process models are not clear enough for data and the information analysis. It should be noted that BPMN and other modeling languages under discussion do not separate accurately data from information, allowing the use of the same symbols for both concepts. For example, in Fig. 49.9, the client's data is represented by data object, and the data of interest rate, maturity, and amount are modeled by data collection symbol. The same data object symbol is used to model documents. Information flow is represented with message arrows and message symbols.

BPMN provides a specific choreography model which allows to concentrate only on conversation between performers. It means that if the basic process model shows frequent communication between performers, then it is useful to model this interaction in a separate model, showing the initiators of communication for each activity and potential speech act. It is possible to consider that in this case it is modeled as nonmaterialized information flow. For example, using choreography model, it is possible to model standard conversation models in bank for bank employers who communicate with clients (Fig. 49.10). However, this model does

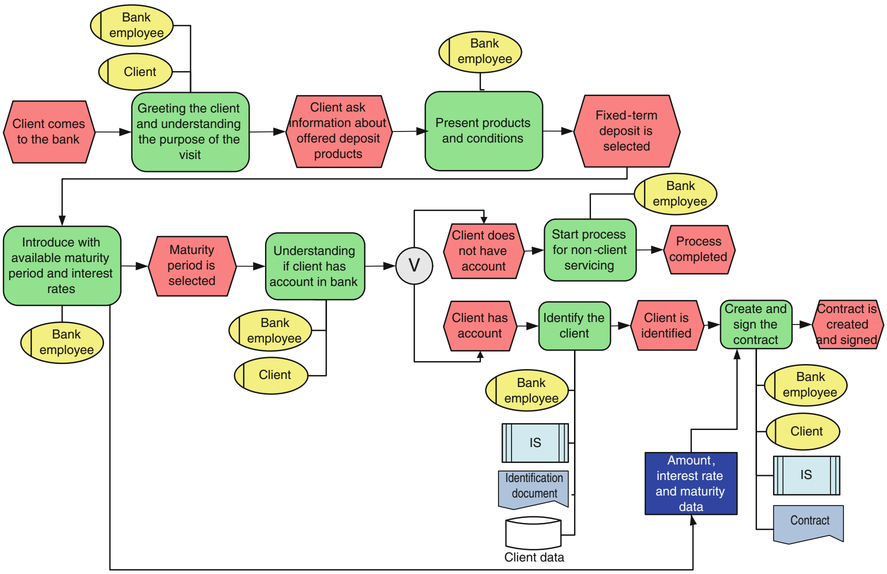


Fig. 49.12 A simplified example of data and information flow modeling in ARIS eEPC

not show how performer's knowledge changes during the conversation and does not specify the available communication tools.

The choreography model in Fig. 49.10 represents possible communication within a particular business process; in contrast, a collaboration model in Fig. 49.11 includes all possible types of interactions between performers within an organization. There is a possibility to add collaboration symbols to particular business process activities, thus showing the communication point. For example, in Fig. 49.11, it is possible to see when and by whom the bank employee interacts performing his/her tasks. Collaboration models are another way how to represent nonmaterialized information flow.

To illustrate data, information, and knowledge interpretation in ARIS eEPC notation, two diagrams are created – process view (Fig. 49.12) and process view with knowledge dimension (Fig. 49.13). To model process view, the modeler needs to choose the viewpoint because the process cannot be represented from the client's and bank employee's viewpoints simultaneously. Process view includes sequence of activities and events from the bank employee's viewpoint. A role is added for each activity. In Fig. 49.12, process is modeled from the bank employee's perspective, the activities are mainly assigned to bank employee role, and events are invented by actions of the client. According to notation, event does not have an assigned role. If a document is used during activity (e.g., contract document created), a document symbol is linked to particular activity. In comparison, BPMN uses data artifact for contract.

ARIS eEPC has an advantage over the discussed notations since it allows visualizing data and information flows though it does not separate these two

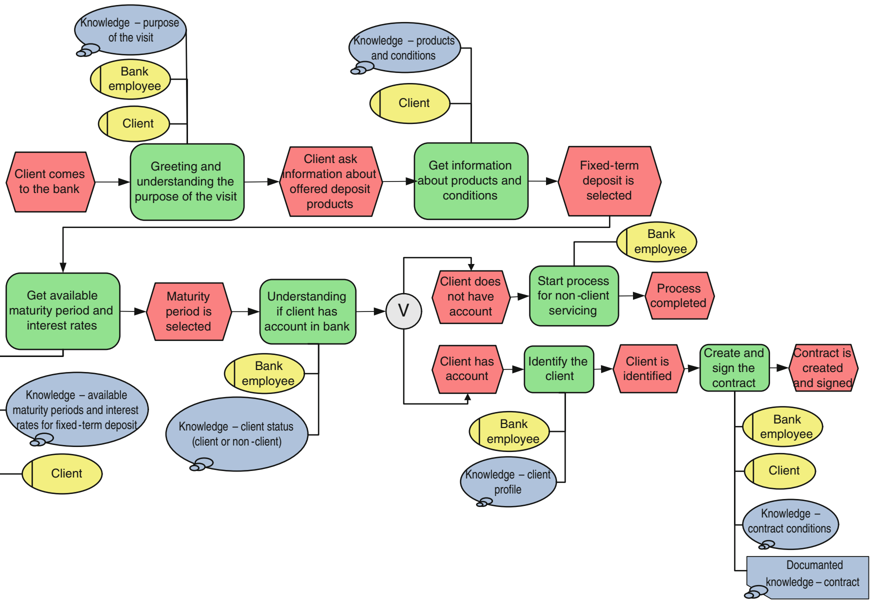


Fig. 49.13 Example of knowledge modeling in ARIS eEPC

concepts. Besides, ARIS eEPC contains many symbols to specify additional information of process activities.

ARIS eEPC process view with the knowledge dimension allows to model only output knowledge (created during activity or obtained/documented as result of activity), mainly for processes which create innovative knowledge. Requirements for the role or input knowledge are not envisaged. The bank employee in deposit contract conclusion example does not collect new knowledge if we do not interpret temporal information about client goals and purpose of the visit as knowledge. Thus, process view is redone according to client perspective, where it is possible to show new knowledge that the client acquires. In Fig. 49.13, knowledge about products and conditions is linked to "get information about products and conditions" activity as the client acquires this knowledge during the conversation with the bank employee. The same knowledge is presented also in KMDL notation (Fig. 49.15).

ARIS eEPC allows modeling separately knowledge hierarchy structure (knowledge categories, tacit/implicit, and documented knowledge) or knowledge map. The knowledge structure diagram is located in the requirements definition data view. The knowledge map, like the expanded model types for business process modeling, is in the requirements definition control view.

KMDL anticipates modeling in three views: (a) process view presents only the sequence of activities and actors (IS, person, or group), similarly to high-level process representation in other business process modeling notations; (b) activity view models particular activity in detail if this is a knowledge intensive activity. It

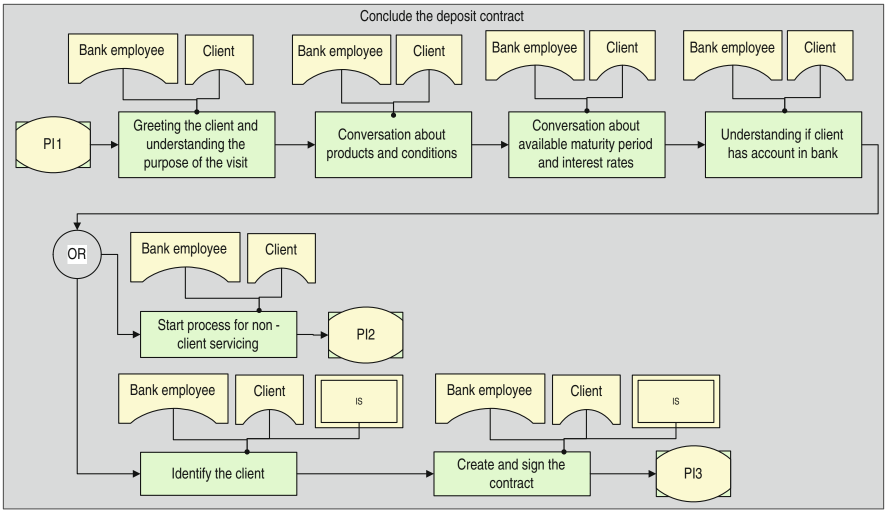


Fig. 49.14 Example of process view in KMDL

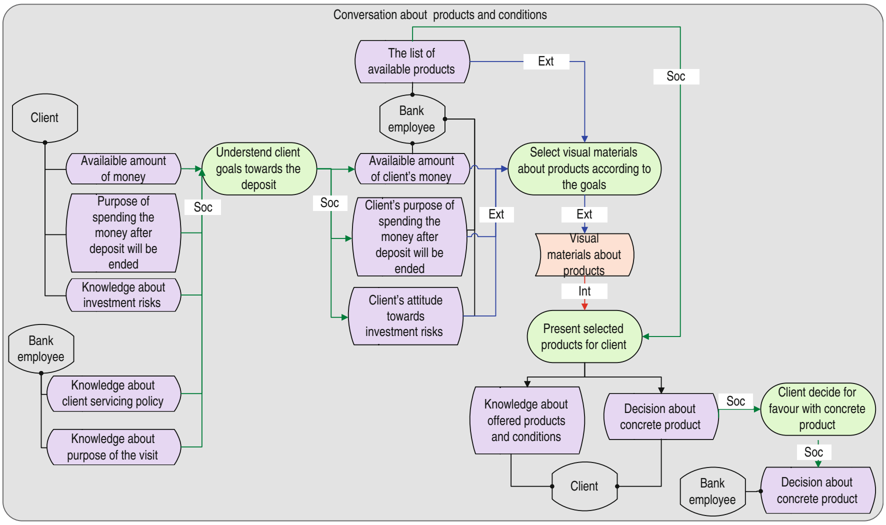


Fig. 49.15 Example of knowledge modeling in KMDL

is possible to demonstrate knowledge conversions and identify requirements for successful activity fulfillment; (c) communication view shows general communication channels, like conversation model in BPMN notation. In this chapter, we present first two views, namely, Fig. 49.14 shows process view and Fig. 49.15 represents activity view.

KMDL all materialized issues are considered as information. Actually in some cases, it might not be so, e.g., "visual materials" from an example (Fig. 49.15) can be considered as a snapshot of several professionals' knowledge. Besides, it is impossible to specify the owner of information, and consequently if more than one performer is involved in the process, it is impossible to establish the information source. Thus, in KMDL, knowledge can be modeled only as tacit, but explicit knowledge is considered as documented information. However, in many cases, it is very useful (functional, methodical, social, and technical requirements) to define explicitly what knowledge and skills are mandatory for activity performer. One more interesting KMDL feature is that it is impossible to model a situation when the process is carried out by one performer who converts existing tacit knowledge into new tacit knowledge. Finally, KMDL has no possibility to model business process logic in full if compared to BPMN and ARIS eEPC.

From the above-mentioned, we can conclude that BPMN and ARIS eEPC are more expressive for modeling process logic, decision points, and control flows, where BPMN offers extended notation for concerning control flow structures and ARIS eEPC is far more expressive in linking with other dimensions (e.g., information systems, products, organization, risks, and key performance indicators). Besides, BPMN diagrams are even directly executable by a process engine. In contrast, ARIS eEPC was originally not designed to describe processes to be executed on a process engine. Instead, it is meant as a language to capture and visualize business processes (ARIS community). Knowledge modeling is possible with KMDL and ARIS eEPC, for data modeling UML activity diagram is more convenient, but for information flows BPMN collaboration and choreography models could be used. However, the discussed modeling languages/notations do not distinguish information and data, allowing their modeling with the same symbols. As a result, this leads to ambiguity and misunderstanding of developed models.

4 Conclusions and Future Work

In this chapter, several business process modeling notations are reviewed, some are process oriented and some are knowledge oriented. The most frequently studied notations in scientific literature were selected to understand their data, information, and knowledge interpretation and modeling capabilities. From the selected list of notations, only ARIS EPC and KMDL support knowledge modeling. ARIS eEPC knowledge modeling support is incomplete, but KMDL has poor capabilities of process control flow modeling. Other notations (BPMN, UML activity diagram, IDEF0, GRAPES-BM, including ARIS eEPC) enable data and information modeling, but do not offer a strict border between these terms allowing using the same elements and symbols for data and information representation.

In the future work, it is planned to review these terms from information theory perspective, analyzing their physical nature and proposing respective definitions. Secondly, it is proposed to implement knowledge dimension in business process models based on well-established data, information, and knowledge definitions.

References

Activity diagrams in UML 2.0. Available at http://www.disi.unige.it/person/ReggioG/ISII04WWW/ActivityDiagrams.ppt

Arbeitsbericht (umfangreiche Beschreibung) – KMDL® v2.2 (2009). Available at http://www.kmdl.de

ARIS community. Available at http://www.ariscommunity.com/users/sstein/2010-04-15-epc-vs-bpmn-perfect-flamewar

ARIS Expert Paper (2007) Business process design as the basis for compliance management. Enterprise Architecture and Business Rules, March 2007

ARIS Method by IDS Scheer AG available at http://www.rmdservice.com/quality_policy/JobAid_ARIS_Method_Manual.pdf

BPMN. Available at http://www.omg.org/spec/BPMN/2.0/PDF

GRADE Business Modeling, Language Reference, Infologistik GmbH, 1998

GRADE user guider. Available at http://www.pearsonassessments.com/hai/Images/Products/GRADE/GRADE_Manual.pdf

Heisig P (2006) The GPO-WM® method for the integration of knowledge management into business processes. In: Sixth international conference on knowledge management (I-KNOW'06), Graz, Austria, pp 331–337

IDEF0. Available at http://www.idef.com/IDEF0.html

IDEF3. Available at http://www.idef.com/IDEF3.html

Supulniece I, Bušinska L, Kirikova M (2010) Towards extending BPMN with the knowledge dimension. Enterprise, business-process and information systems modeling: Proceedings, Tunisa, Hammamet, pp 69–81

UML. Available at http://www.uml.org/

Weidong Z, Weihui D (2008) Integrated modeling of business processes and knowledge flow based on RAD. In: International symposium on knowledge acquisition and modeling, Wuhan, China, pp 49–53

Woitsch R, Karagiannis D (2005) Process oriented knowledge management: a service based approach. J Univers Comput Sci 11(4):565–588

Chapter 50
Towards a Supporting Framework for Public Web Service Efficient Reuse: A Model Based on Social and Collective Usage Experience

Uddam Chukmol, Aïcha-Nabila Benharkat, and Youssef Amghar

1 Introduction

Public Web services are accessible and available Web services in the Internet. They are generally published within Web portals, catalogues or repositories directly managed by their providers (Fan and Kambhampati 2005). At the back end, service providers are not obliged to respect the UDDI standard for the implementation of their repositories (Wu and Chang 2007). Service providers offer to end users service lookup interface implemented within their repositories, and end users can invoke directly a found service from those repositories (AbuJarour et al. 2010).

Recent access to seekda[1] shows around 28,606 public Web services out of 7,739 providers, which is not negligible as a source of interest in promoting SOA adoption to a larger public than the one traditionally identified in intra-corporate environment. However, exploiting those publicly exposed Web services may come with some challenges due to the following facts:

- Poor textual documentation found in service description makes end users put more effort in understanding what a service can do, what it needs as input and what it can produce as output. Consumers may have to infer the operation names or message types used in the service in order to utilize them correctly (Li et al. 2007).
- Poor semantic extension reduces considerably the possibility to automate different essential tasks such as Web service discovery, composition and substitution. Semantic mark-up on Web services comes from Web semantic communities by extending original WSDL descriptions with various semantic formalisms based

[1] http://webservices.seekda.com

U. Chukmol (✉) • A.-N. Benharkat • Y. Amghar
Université de Lyon, CNRS, INSA-Lyon, LIRIS, UMR5205, F-69621, France
e-mail: uddam.chukmol@insa-lyon.fr; nabila.benharkat@insa-lyon.fr;
youssef.amghar@insa-lyon.fr

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_50,

on ontology. Several available formalisms do not really facilitate the automation process but bring somehow burdens to both service providers, in choosing an appropriate formalism and domain ontology to use as semantic annotation, and consumers in acquiring expertise to construct discovery query or building business process schema for composing Web services (Hageman).

- Non-situational refers to the fact that public Web services are for general-purposed usage. They are not predestined to be used in a planned composition schema or any specific context. End users can consume them appropriately in their own ways which could be different from the original applicative aims targeted by service providers (Tanaka et al. 2008).

Difficult usage begets hard reuse. In effect, Web service reuse can be simply and practically put as several or repetitive usages of the same Web service instance through time and within different situations. The service is designed and published once, then used and reused over and over again by different consumers in different contexts. Web service reuse is strongly influenced by its usage and represents a great return on investment if it is well achieved.

Public Web services that are available "on the Web" provide unlimited values for a great number of online service consumers and developers, whereas most private registries would focus on a specific, closed domain such as the corporate network (Wu and Chang 2007). Hence, in an open, loosely coupled, demand-driven business environment, a maximal reuse of public Web services is still essential to SOA adoption from a wide range of users such as service consumers, service developers and service brokers.

In this chapter, we would like to tackle the reuse of publicly available Web services from the end-user point of view. In effect, after being published, end users discover, select and consume public Web services in their applications. They are then apt to comment such services on functional or non-functional aspect. Such feedbacks are recommended as supplementary semantics of Web services and can be served to enhance the *discoverability* and the *recommendability* of those services (Loutas et al. 2009; Petrie 2009; El-Goarany et al. 2008). Other users would consider if a service can be *re*used in their application based on the other usage experiences. We attempt in this chapter to provide a simple way by which users can express their service usage experiences. We outline a set of service *reusability* criteria and show how end-user feedback on the services can be used to satisfy those criteria and reinforce implicitly the service *reusability*.

This chapter is organized as follows: Sect. 2 discusses related works on Web service reusability and offers a synthesized set of criteria on Web service *reusability*. Section 3 proposes a formalism allowing end users to express their service usage experiences. Section 4 presents the architecture of a supporting system exploiting aforementioned feedbacks to promote and enhance Web service *reusability*. We end up with a conclusion and perspective on future works.

2 Related Works

Web service *reusability* is generally discussed as an issue in service design phase and rarely in consumption phase (published and ready to use). A main challenge is how to bridge the gap between Web services and general public. In effect, the benefit from reusable Web services helps consumers in getting better return on investment because the services can be employed in different situations in their applications and providers in attracting more consumers, gaining more credibility and being more trustworthy.

The existing works on reusable Web services are mostly focused on service providers rather than considering end-user perceptions on services. In Dan et al. (2008), reusing Web services refers to good governance in SOA lifecycle. The proposal is very enterprise oriented and consists of using a common language with consistent business term to describe services, governing new service creation and discovery of existing services, providing assurance to both service users and owners and supporting new requirements against existing services. Following the same idea, the proposal in Chu and Qian (2009) considers designing reuse-oriented Web service through a top-down process by modelling business requirement as business processes, breaking them down and identifying fine-grained requirement before the real service design. These two solutions may be applied efficiently in supervised and inter/intra-corporate environment where business terms can be controlled by internal glossary or domain ontology; business processes are predefined, imposed and slightly evolving, and expertise in SOA governance is present. In contrary, public Web services use the Internet, and the latter represents an unsupervised environment but focuses more on end users. Therefore, applying the aforementioned concepts to public Web services may not bring interesting outcome.

Besides, another work is interested in providing reusable data-centric Web services (Saleh et al. 2009) through a formal specification of Web service model. Their model attempts to deal with the interpretation of data that must be marshalled between service consumers and producer systems. It is again an answer to *reusability* of a specific kind of Web service and may not cover all existing categories found among public Web services.

Nonetheless, Bechara and Feuerlicht and Wijayaweera (2007) consider reusable Web services as respecting some general criteria:

- Loose coupling: between services, it refers to minimal dependency among them and between providers and consumers; it means that the service providers shall hide service implementation detail from their consumer by using standard interface (e.g. WSDL, SOAP messages).
- Functional cohesion: a service can contain several operations, and each of them should execute a specific task in order to guarantee the modularity of services.
- Granularity: it is normally hard to achieve. Coarse-grained services return the result after aggregating several response messages. They are then easy to identify. Fine-grained services execute elementary function. They need more breakdown to identify lowest-level elementary actions that they will implement.

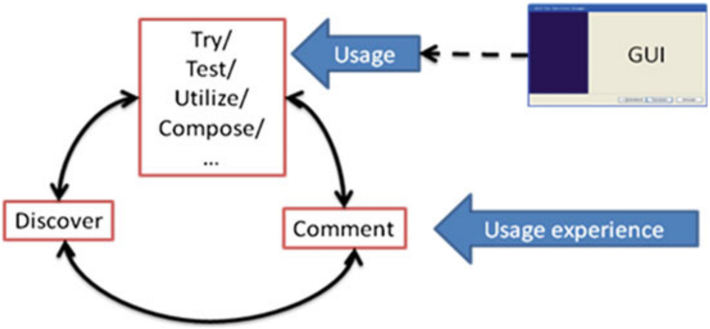


Fig. 50.1 Public web service lifecycle seen by end users

- Statelessness: it refers to the fact that during its execution, a service does not wait for any event or callback. There should not be any temporary state during a service execution because it can be very costly to take into account such states.
- Autonomy: it refers to autonomous resource usage of a service (e.g. databases, documents).
- Discoverability: ability of a service to be identified given a user request that expresses his/her requirement.
- Composability: ability of a service to participate in a composition or a mashup scenario given a requirement.

From user point of view, we reduce these *reusability* criteria into usability, discoverability and composability. Usability covers, in effect, loose coupling, functional cohesion, granularity, statelessness and autonomy. This criterion (usability) requires works from service providers to realize, particularly in service design time. Once a service respecting this criterion is published and deployed, it should only be or seen as *runnable* or executable by consumers via testing or try interface provided by service providers. Thus, service usability is designed by service providers and perceived by consumers.

Discoverability and composability, however, can depend both on textual documentation and/or semantic annotation that service providers attach with their services. The mentioned supplementary information can be consulted by service end users but may submerge them into technical difficulty in interpreting or understanding how to use the documented or annotated services. We are facing problems related to common terminology and domain ontology usage between service providers and consumers. Moreover, expertise at both sides is needed in order to work coherently on the service semantic annotations and to construct correct semantic discovery or composition requests.

In the actual work, we choose to tackle the public Web service *reusability* from end-user point of view by encouraging end users to express their service usage experience for the benefit of the others in the community. We initiate our proposal based on the following observation (see Fig. 50.1):

We consider that end users can observe what happens when they use a service and can report what goes through their perceptions to the others and make them taking profit from previous usage experiences to enhance on one hand the *reusability* of Web services and on the other hand provide real situational meta-data on published Web services to their communities. Our approach is oriented to socially provided service semantics as advocated by Averbakh et al. (2009), Jansen et al. (2008), and Leitner et al. (2009). We then have to provide possibilities to end users, allowing them to express their usage experience in a simple way and to exploit those experiences to promote public Web service *reusability*.

3 Expressing Service Usage Experience

We define the process of providing service usage experience as a socialization one. It is based on a simple observatory hypothesis which is "more a service is socialized, more it has been tested or tried". If a service is repetitively used in various situations, then it is considered as reusable. In this case, functional aspect interests us more than non-functional one. In effect, the latter is often use to compute the selectivity of Web services.

We opt for collaborative socialization process, i.e. users can comment on the services they have been using, and those comments are available to other users. Comments from different users on a service are aggregated to reflect the importance of the commented service. Thus, collaborative tagging (Papathanasiou et al. 2009) is a relevant option for expressing this kind of feedback after using, testing or trying a service.

A service (taggable object) can be tagged by several users with several tags. A tag can be used by several users to tag several services. Therefore, the tagging model can be summarized as a triplet $<$*User, Tag, TaggableObject*$>$ (see Fig. 50.2). The model illustrated above can be stored in the *folksonomy base* in our supporting system.

Expressing a service usage experience consists then of reporting on it usage situation, what operation has been used, what input is required and what output is produced. Such information can be observed during a test or a try on a service.

According to the structure of WSDL that is normally used to describe Web services, a Web service can be considered as an object that can be parsed into several operations, and each of them is respectively into several pairs of input and output. We allow end users to express such features related to a service they used in their usage experience by focusing particularly on the functional capacity of the service (see Fig. 50.3).

Using tags is both flexible and easy for nonexpert consumers. However, synonymy of terms can generate ambiguity in providing the correct meaning of a used term. Therefore, our tag is associated with a meaning. Each tag has a weight which is its occurrence number, and this weight reflects the importance of the tag related to the tagged service. We impose a simple tagging convention by using ";" as

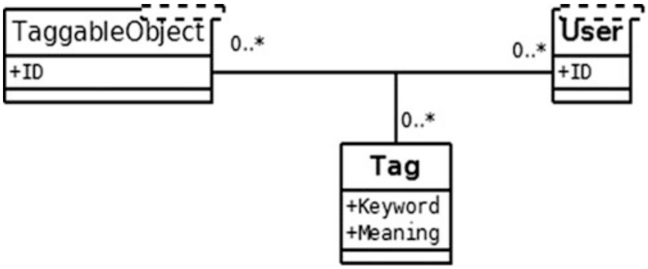


Fig. 50.2 Folksonomic tagging model

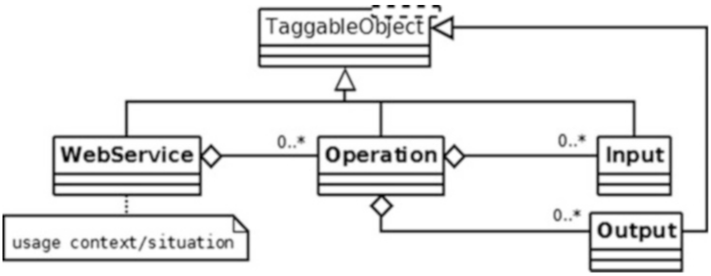


Fig. 50.3 Different taggable item in a Web service

separator, "{}" as block definition and "_" for multiple term tag (see Fig. 50.4 for an example: ***{flight, sense{flight, 0}, 10}*** means that the tag "***flight***" is associated to the first meaning found in the term meaning base (***indexed by 0***) and is tagged ***10*** times on the service).

The next section depicts how we can exploit the user-provided annotation on Web services to promote and enhance the *reusability* of public Web services. We emphasize on the fact that a reusable Web service is most likely usable, discoverable and composable. These criteria should be perceived by end users within public Web service lifecycle in order to encourage them to use and then reuse the published Web services.

4 Public Web Service Reuse Supporting Framework Architecture

The main goal of this supporting framework is twofold: service providers can use it to advertise their published Web services with help from end-user community by making their service more and more usable, discoverable and composable. Service consumers, via this framework, should be able to test, try, use, discover, compose, substitute and annotate published Web services (see Fig. 50.5).

```
WS1:
 + used in: {  {{flight, sense<fligth, 0>}, 10};
                {{confirmation, sense<confirmation, 0>}, 5};
                {{booking, sense<booking, 0>}, 2}
              }
 + with an operation performing: {{{send, sense<send, 0>}, 12};
                                  {{sms, sense<sms, 0>}, 20};
              {{text_message, sense<text_message, null>}, 8}
              }
 + by consuming: {    {{flight, sense<flight, 0>}, 5};
                      {{number, sense<number, 0>}, 10};
              {{phoneNumber, sense<phoneNumber, null>}, 3}
                                  }
 + and producing: {   {{sms, sense<sms, 0>, 10}};
                      {{flight, sense<flight, 0>}, 4};
                      {{schedule, sense<schedule, 0>}, 3}
              }
```

Fig. 50.4 Tags associated to WS1

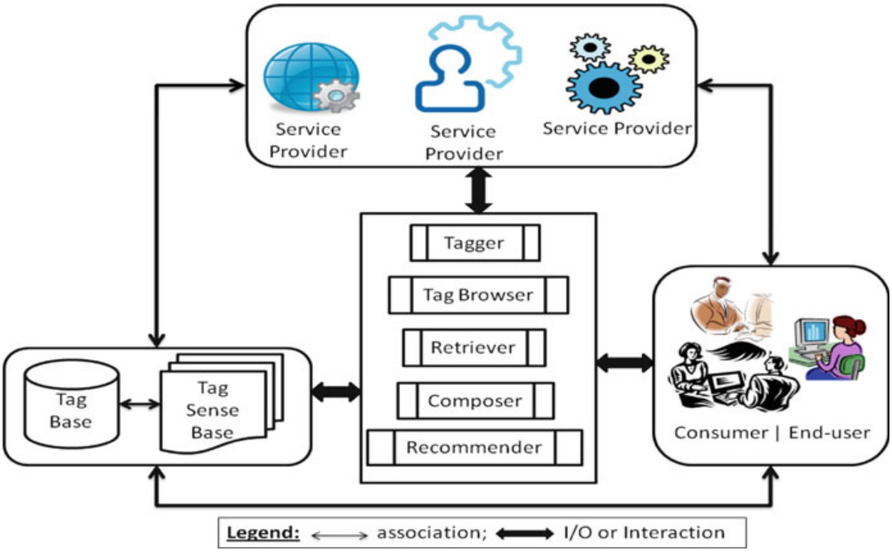


Fig. 50.5 Supporting framework for public Web service reuse

This system is divided into several modules. It also depends on a tag sense base that contains not only a terminological relationship base between tags such as synonym but also for each tag an array of possible meanings associated to that tag. Several existing lightweight knowledge bases or thesaurus can be used to

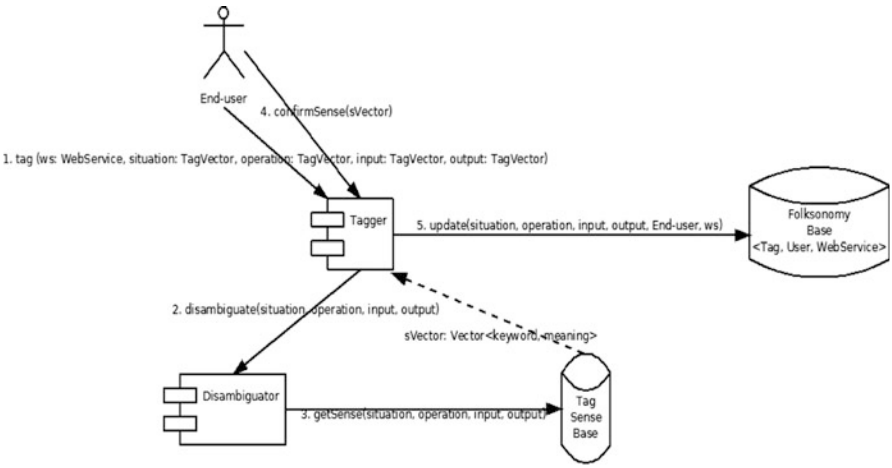


Fig. 50.6 User-system interaction while expressing a service usage experience

implement our tag sense base such as WordNet,[2] Wikipedia, Common Tag[3] or DBpedia.[4] However, we cannot use exclusively one of them to disambiguate in the tagging process. In effect, on one hand, we search for simple visual sense array proposed to end users while they are entering a tag having multiple senses and, on the other hand, we aim at finding the tag relatedness such as synonymy relation in order to extend the Web service discovery and composition space. Using WordNet with one among the rest of the aforementioned candidates seems very plausible.

Tagger module allows end users to provide comment, feedback or usage experience (tag or annotate) on a Web service. Each given tag is disambiguated thanks to tag sense base, and finally, the triplet *<user, tag, annotated item>* is stored in the *folksonomy base* (see Fig. 50.6).

Tag browser module allows end users to visualize tagged Web services and the end users having tagged services given a selected tag to browse. This module gives an insight on what tags have been used and by what users before commenting a service. It can help the tagging user to choose the already used tags to express his/her usage experience (see Fig. 50.7).

Retriever module allows end users to discover tagged Web services based on tags. Looking up Web services this way is more extensive and richer than using simple keyword because in our case, tags are disambiguated and bundled into a structure. We then exploit the lexical and structural information related to the searched services. Moreover, inside the service usage experience expression, usage situation is given, and it can help enhance the quality of discovery process. A discovery request must be formalized as how tags are organized around services.

---

[2] http://wordnet.princeton.edu

[3] http://commontag.org/Home

[4] http://dbpedia.org/About

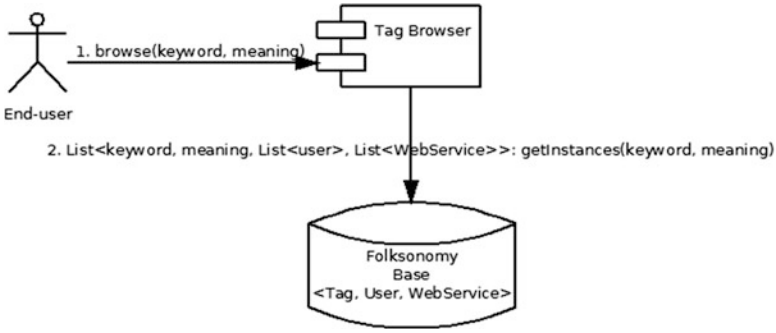


Fig. 50.7 User-system interaction while browsing a tag for services or users

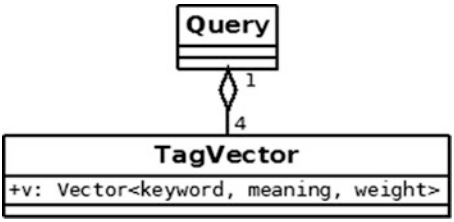


Fig. 50.8 Discovery request model

It is represented by four tag vectors (usage situation, operation, input and output), but each tag item has no weight (see Figs. 50.8 and 50.9). Tag-wise similarity is the core of the discovery algorithm, and we define that two tags are identical if and only if they share the same meaning and the same term. Two tags are similar if they have synonymy relatedness or their meanings reveal a certain degree of textual similarity (e.g. computed by co-occurrence, cosine, Jaccard or dice measures) (Cattuto et al. 2008).

Recommender module can be used as the extension of the three above modules. Its role is to exploit the co-occurrence between tags, determine which tag appears frequently with another, compute closeness of user profile generated from tagging activity in order to propose similar service tagged by user having the similar profile and propose other services tagged by similar tags.

Composer module allows end users to establish a visual composition of a number of services by drag-and-drop actions. After a service is dropped in the composition space, the system has to look up for the most appropriate candidate to pipe from its output by exploiting the exact or similar tag match. This module is served as a concept-proof of likely composability of a chosen service. It is still very hard to achieve real composite services this way due to input, output and constraint on manipulated data type.

```
Find a service:
  + used in: {   {flight, sense<flight, 0>};
                          {confirmation, sense<confirmation, 0>}
                 }
 + with an operation performing: {  {send, sense<send, 0>};
                                           {sms, sense<sms, 0>}
                                   }
 + by consuming: {     {flight, sense<flight, 0>};
                       {number, sense<number, 0>};
                       {phoneNumber, sense<phoneNumber, null>}
                  }
 + and producing: {    {sms, sense<sms, 0>};
                       {flight, sense<flight, 0>};
                       {schedule, sense<schedule, 0>}
                  }
```

Fig. 50.9 Textual example of discovery request

5 Conclusion and Future Works

The main contribution of this chapter lies in the identification of service *reusability* features. We then propose a tagging model allowing service consumers to express their usage experience by using tags. We bring in the concepts of using tags to enhance or promote the likely usability, discoverability and composability of services. We propose also some service recommendation mechanisms and the concept of visual service composition.

Then, we provide the architecture of a supporting system that allows end users and service providers to easily enhance and promote public Web service reuse. It can also be a potential candidate to be employed for realizing different tasks such as:

- Web service discovery at design time: service-oriented application engineers can dispose of such system to assist his/her task in finding services that are relevant to the functional requirement of clients. Such system allows engineers to browse different services in the enterprise collection by simple tags and formulate easily his/her request by just following a simple guide. Usage experience of services can be shared among engineer community to improve the reuse of locally available Web services, if those engineers care enough participating in the tagging activities after testing or using the services.
- User-assisted Web service composition: when a user would like to have on-time effects or interventions on the composition of Web services, he/she can check the tag clouds at his/her disposal for atomic service that responds to his/her prompt need in order to create his/her composite service. The need formulation is easy, and our tagging model favours the annotations on input/output of Web services which are habitually useful for the composition process.
- User-oriented and semi-automatic Web services clustering or classification: with the increasing number of tags on functionality of services provided by users, the

expression of actual usage of services can be deduced. By employing efficient text or data mining algorithms, the clustering or classification of such annotated service collection is more than feasible and can bring interesting outcome.

- User-centric Web service recommendation: collaborative tagging provides meta-data in form of tags which can be exploited in the recommendation process based on collaborative or user profile-based filtering.

However, dealing with tags provokes some issues also such as tag bombing or tag spamming. We do not tackle such issues in the scope of this chapter.

As the perspective of our proposal, technical issues need to be studied in order to implement an appropriate system based on our aforementioned architecture. Several elements need to be considered such as the complexity of tag disambiguation process, the quality and scalability of service discovery and composition based on tags. Service recommendation process should be studied in more detail in order to exploit the minimum amount of data for the acceptable results.

To sum up, we attempt to promote the likelihood of public Web service reuse by socializing those services. Service providers can take benefit from such approach without heavy additional work. End users are however encouraged to express their service usage experience in a collective and collaborative environment in order to promote service *reusability*.

References

AbuJarour M, Naumann F, Craculeac M (2010) Collecting, annotating, and classifying public web services. In: Proceedings of the 2010 international conference on the move to meaningful internet systems, volume part I. Springer, Berlin/Heidelberg, pp 256–272

Averbakh A, Krause D, Skoutas D (2009) Exploiting user feedback to improve semantic web service discovery. The Semantic Web – ISWC 2009, pp 33–48

Bechara G . What is a reusable service? http://www.oracle.com/technetwork/articles/bechara-reusable-service-087796.html

Cattuto C, Benz D, Hotho A, Stumme G (2008) Semantic grounding of tag relatedness in social bookmarking systems. In: Sheth A, Staab S, Dean M, Paolucci M, Maynard D, Finin T, Thirunarayan K (eds) The semantic web – ISWC 2008. Springer, Berlin/Heidelberg, pp 615–631

Chu W, Qian D (2009) Design web services: towards service reuse at the design level. J Comput 4:193–200. Academy Publisher

Dan A, Johnson RD, Carrato T (2008) SOA service reuse by design. In: Proceedings of the 2nd international workshop on systems development in SOA environments. ACM, New York, pp 25–28

El-Goarany K, Saleh I, Kulczycki G (2008) The social service network – web 2.0 can make semantic web services happen. In: 10th IEEE conference on E-commerce technology and the Fifth IEEE conference on enterprise computing, E-commerce and E-services, 2008, pp 419–423

Fan J, Kambhampati S (2005) A snapshot of public web services. SIGMOD Rec 34:24–32

Feuerlicht G, Wijayaweera A (2007) Determinants of service reusability. In: Proceeding of the 2007 conference on new trends in software methodologies, tools and techniques: proceedings of the sixth SoMeT 2007. IOS Press, Amsterdam, pp 467–474

Hageman S, Letz C, Vossen G. Web service discovery – reality check 2.0.
Jansen S, Brinkkemper S, Hunink I, Demir C (2008) Pragmatic and opportunistic reuse in innovative start-up companies. IEEE Softw 25:42–49
Leitner P, Michlmayr A, Rosenberg F, Dustdar S (2009) Selecting web services based on past user experiences. In: 2009 I.E. Asia-Pacific services computing conference, Biopolis, Singapore, 7 décembre
Li Y, Liu Y, Zhang L, Li G, Xie B, Sun J (2007) An exploratory study of web services on the internet. In: IEEE international conference on web services, 2007. ICWS 2007. IEEE Computer Society, Los Alamitos, pp 380–387
Loutas N, Peristeras V, Tarabanis K (2009) Rethinking the semantic annotation of services. In: Proceedings of the 2009 international conference on service-oriented computing, pp 540–549
Papathanasiou MK, Loutas N, Peristeras V, Tarampanis K (2009) Combining service models, semantic and web 2.0 technologies to create a rich citizen experience. In: Proceedings of the 2nd world summit on the knowledge society: visioning and engineering the knowledge society. A web science perspective. Springer, Chania, Crete, Greece, pp 296–305
Petrie C (2009) Practical web services. IEEE Internet Comput 13:93–96
Saleh I, Kulczycki G, Blake MB (2009) A reusable model for data-centric web services. In: Proceedings of the 11th international conference on software reuse: formal foundations of reuse and domain engineering. Springer, Berlin/Heidelberg, pp 288–297
Tanaka M, Murakami Y, Ishida T (2008) Towards service supervision for public web services. IEEE Asia-Pacific services computing conference, 2008. APSCC 2008, pp 536–541
Wu C, Chang E (2007) Searching services "on the Web": a public web services discovery approach. 2007, Third international IEEE conference on signal-image technologies and internet-based system, Shanghai, China, pp 321–328

Chapter 51
Analyzing Agents' Collective Design Decisions in Collaborative Systems Development Initiatives

Majed Al-Shawa

1 Introduction

System design and development initiatives are mostly ill-structured strategic collaborative decision-making situations, with outcomes that rely on the rich contextual knowledge that each has. Despite the cooperation and collaboration among the agents who are involved in such situations, these agents still have to satisfy different and most likely conflicting strategic goals and constraints (internal and external) that they have. In most cases, the agents adopt an option (set of requirements, design, architecture) that considers only the needs and realities of few agents and ignore or suppress others' needs and realities. This leaves many collaborative strategic software implementation initiatives fail or be challenged (see the CHAOS reports by the Standish group, e.g.). In such initiatives, the agents' preferences are usually not clear, or hard to validate, and that agents' options/moves are hard to completely capture. Nevertheless, most dominant modeling and analysis tools, such as decision and game theoretic methods, assume predetermined agents' preferences, or utility functions, and predetermined set of alternatives to evaluate. This leads to a lack of applicability of such models to model and analyze real-life strategic conflicts (Keeney 1992, Raiffa et al. 2002).

The limitation of the decision theory and game theory approaches, in general (Keeney 1992, Raiffa et al. 2002) and for multiagent systems (Wooldridge 2000), led to the emergence of a new direction within the research community, namely, within the decision analysis (Keeney 1992), the AI multiagent BDI (Shaw and Bordini 2007, Winikoff et al. 2002), and the software requirements engineering (Dardenne et al. 1993) communities: Modeling Goals and Reasoning about them. But the current frameworks lack the representation mechanisms to support modeling goals (interrelationships, constraints, prioritization, etc.) and therefore

M. Al-Shawa (✉)
Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada
e-mail: mmalshawa@uwaterloo.ca

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_51,

reason about them. Recently, in Al-Shawa (2006a, 2011), we talked about the shortcomings of the current frameworks and the need to extend them at different levels to be able to make them well suited for multiagent knowledge-based systems to support strategic decision making.

Constrained Rationality is a formal qualitative value-driven enterprise knowledge management modeling and analysis framework (Al-Shawa 2011, Al-Shawa and Basir 2009), with a robust methodological approach that addresses such challenge by bringing back the decision and conflicts analysis to its roots: reasoning about goals and plans to achieve the strategic goals the agent has. The framework (1) uses each agents' individual and collective contextual knowledge about their goals and constraints to suggest the set of options the agents has, (2) takes in consideration the agent's priorities, and (3) elicits accordingly the agent's preferences over their collective alternatives. The framework allows decision makers, especially at the strategic level, model their goals, model their internal and external constraints (realities which limit or open opportunities to their goals—from this the name of the framework came), model the interrelations among these goals and constraints and how they affect each other, and then finally evaluate their plans/options based on the collective overall goals-constraints model.

This work extends our previous work, the viewpoint-based value-driven conceptual modeling (Al-Shawa 2006a,b) and the Constrained Rationality's basic goal and constraint modeling and reasoning (Al-Shawa and Basir 2009), and proposes new modeling and reasoning facilities, for collaborative decision-making situations, to help generate the agents' collective cardinal and ordinal preferences over their shared alternatives. This chapter starts with Sect. 2 providing an overview of each agent's Goal and Constraints Model (GCM) and its constructs (goal and constraint nodes and interrelationships) and how qualitative labels of goals' value properties, such as the goals' achievement, operationalization, and prevention, propagate through the nodes interrelationships to finalize how much achievement each alternative provides to the agent's strategic goals. Section 3 discusses how the agent's GCMs will be integrated and their priorities, over their individual strategic goals, are added and modeled. Section 4 shows how the carinal and ordinal preferences of the agents collectively, over each of their shared alternatives, will be calculated and represented. Finally, Sects. 5 and 6 conclude with an illustrative application of the concepts and methods proposed in this chapter, with the focus primary on the process the agents will use to model their collaborative decision-making situation and evaluate their alternatives.

2 Agent's Goals and Constraints Model

The Goals and Constraints Model (GCM), presented in Al-Shawa and Basir [2009], is a sub-model of each agent's Viewpoint model. GCM captures the agent's goals and constraints with regard to the specific situation/conflict this viewpoint model is concerned with. GCM is a graph-like structure $\langle \mathscr{G}, \mathscr{C}, \mathscr{R} \rangle$ where $\mathscr{G}$ is a set of goal

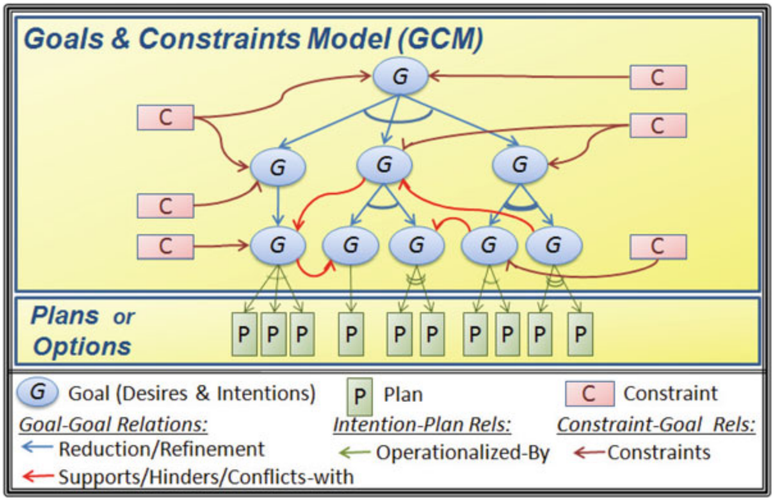


Fig. 51.1 Goals and constraints model (GCM), with simple one goalstree

nodes, $\mathscr{C}$ is a set of constraint nodes, and $\mathscr{R}$ is a set of interrelationships over the nodes of $\mathscr{G}$ and $\mathscr{C}$. Figure 51.1 shows an illustration of a simple one goals-tree GCM model.

Goal nodes in GCM represent the motivation the agent has and are modeled by first inserting the ultimate strategic goals the agent has, then go through a reduction process, by using reduction relations, refining these big goals, called *Desires*, to a set of smaller Desires, and so on until a set of primitive very refined goals, called *Intentions*, are produced. Intentions are goals that could be operationalized by means of Plans, while Desires are goals that could be operationalized by other Desires or Intentions. The end result of the goals reduction process is a goal tree, or a set of goal trees, where ultimate strategic Desires form the roots of these trees, and with Intentions at the bottom of each goal tree. In addition, *Constraint* nodes form an important component of each GCM. Constraints represent not only limitations on goals, i.e., affecting goals negatively, but also they could represent opportunities. Representing constraints as nodes, instead of variables within goal nodes, allow for complex and realistic constraint representation, as discussed in Al-Shawa Al-Shawa [2006a, 2011]. We discussed there also how goal nodes are interconnected through a set of goal-to-goal (G-G) reduction and lateral relations, and constraint nodes affect goal nodes through a set lateral constraint-to-goal (C-G) relations.

Each goal node $G \in \mathscr{G}$, part of the GCM, has three value properties, as discussed in Al-Shawa and Basir [2009]. First, *Goal Achievement* provides a measure of the achievement level of G, and denoted as Achv(G). Goals' achievement levels propagate up the goals reduction tree from the intentions at the bottom (based on results from the plans attached to those intentions) and up the goals tree until a value is assigned to the achievement level of the goal or through the G-G

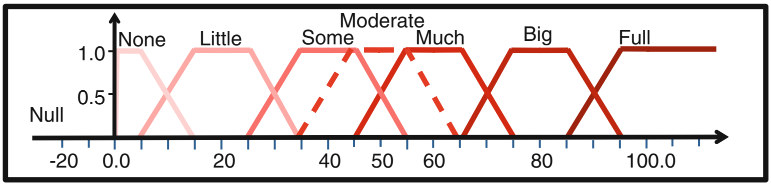


Fig. 51.2 Fuzzy sets dividing the satisfaction levels domain of the different goals' value properties (operationalization, achievement, and prevention)

lateral relations. Second, *Goal Prevention* describes the hindering (negative) effect that other goal's achievement has on G and denoted as Prvn(G). The Prevention property is especially important to track conflicting/hindering effect that may be hidden otherwise (if we have only achievement level indicators for goals). Third, *Goal Operationalization* describes the operationalization level of G, and denoted as Opr(G). This property will state whether the agent has committed itself to a set of plans that will ensure a degree of operationalization for the goal, or not. Higher goals in the trees have operationalization levels that reflect the degree of operationalization that is provided to each by the lower level goals, mainly the Intentions. It is important to track Operationalization, separate from Achievement, because the maximum level of achievability possible for any goal depends on the level of operationalization the agent commits to it. For each constraint node $C \in \mathscr{C}$, there are two value properties: First, *Constraint Achievement* value attached to C to reflect the true reality/strength of the constraint C as imposed by the enforcer, or as believed to be enforced/exist, and denoted as Achv (C). Second, *Constraint Prevention* value attached to C, denoted as Prvn (C), to reflect the prevention the constraint suffers from, stopping it fully or partially from having its effect on the goals it affects.

For the Constraint Rationality's qualitative reasoning framework (Al-Shawa 2011, Al-Shawa and Basir 2009), let us consider a limited number of satisfaction levels (instead of considering all the levels between 0 and 100 %) for these value properties' variables. And let these levels be defined as fuzzy sets; each is given a name which represents a meaningful linguistic label such as Full or Some. Each fuzzy sets to be defined by a fuzzy membership function mapping the actual satisfaction level of the property (0–100 %) to a set membership degree [0, 1]. While the fuzzy domain of any value property's satisfaction levels can be divided into any number of fuzzy sets, as deemed beneficial to the framework user, we introduce a simple but sufficient scheme to divide the satisfaction level of each value property to seven fuzzy sets: *Full*, *Big*, *Much*, *Moderate*, *Some*, *Little*, and *None*. These fuzzy sets cover all the value properties (Operationalization, Achievement, or Prevention) for goals/constraints, as shown in Fig. 51.2. Membership functions, in practice, should be defined as per the user needs.

Let us introduce $\mathscr{L}$ as a set of labels whose elements match in number and names the fuzzy sets chosen to divide the satisfaction levels of the operationalization, achievement, and prevention value properties. In our case, $\mathscr{L} = \{Full, Big, Much, Moderate, Some, Little, None\} = \{F, B, M, Mo, S, L, N\}$. And let $F > B > M > Mo > S > L > N$, matching the order of the fuzzy sets coverage over the satisfaction levels domain. Let the *Achievement* value property of a goal G_i is represented as $Achv(G_i) = L_{\text{achv}}$, where $L_{\text{achv}} \in \mathscr{L}$ and L_{achv} is a label that matches the name of the fuzzy set which the achievement level of G_i has the highest membership of. The same is assumed for both $Opr(G_i)$ and $Prvn(G_i)$. We also use the proposition *Null* to represent the Null trivially true statement that the status of the satisfaction level of the value property for a goal/constraint is unknown or negative. We also add the *Null* label to $\mathcal{L}$, where $F > B > M > Mo > S > L > N > Null$.

In Al-Shawa and Basir [2009] and Al-Shawa [2011], a comprehensive, flexible, and extendable set of G-G and C-G interrelationships were proposed, including (1) Goal Reduction/Refinement *and*/*or* Relations responsible for generating the tree-like structures found in goal tree/s and (2) Goal-Goal Lateral Relations, to represent *Supports*, *Hinders*, and *Conflicts with* among goal nodes in GCMs. The G-G lateral relations are named based on whether the cause/effect is positive (achievement or operationalization) or negative (prevention) on the goal at that end of the relation. For example, if G_1 is achieved fully and this will cause G to be fully achieved as well, then we call the relation a " $++$ " relation, and if having G_1 fully achieve will cause G to be fully prevented, then the relation is called a " $+-$ " relation. On the other hand, constraints are connected to goal nodes through Constraint-Goal (C-G) Lateral Relations, which are similar to the G-G Lateral ones, with similar propagation rules for value labels, with two exception: constraints do not have operationalization value and do not affect the target goal-nodes' operationalization and constraints set the upper and lower limits of the $Achv(G_i)$ not actual achievement level s for G_i to harness through the C-G lateral relations. Each G-G, or C-G, lateral relation has a *Modifier*. For example, the lateral relation of " $+$ $(Some+)$" represents a relation in which a full or partial satisfaction of the source node will make the satisfaction level of the target node be partial. This relation's *Modifier* is "*Some*". The relation's *ModifierM* is a label that belongs to the same set of labels $\mathscr{L}$ used for value properties, i.e., $M \in \mathscr{L}$. Note that an assignment of *Null* as a relation's *Modifier* makes the relation as if the relation does not exist.

In addition, a complete set of propagation rules for the goals' fuzzy value property labels to propagate through the G-G relations (reduction and lateral) and C-G lateral relations were proposed in Al-Shawa and Basir [2009] and Al-Shawa [2011]. Table 51.1 lists the G-G relations' propagation rules (Al-Shawa 2011 provides also the C-G ones). The final value labels of G_i at any time t are concluded by the following propagation rules (an algorithm to calculate the final value property labels for each G_i the agent has is given in Al-Shawa and Basir [2009] and Al-Shawa [2011]):

$$\mathrm{Opr}(G_i) = \left(\mathrm{Opr}(G_i) \vee \bigvee_{r_j \in \mathscr{R}_{G-G_i}} \mathrm{Opr}_{r_j}(G_i) \right) \tag{51.22}$$

Table 51.1 The propagation rules of value labels using goal-to-goal relations

G-G AND Reduction Relations	*G-G OR Reduction Relations:*
$(G_1, G_2) \xrightarrow{and} G:$	$(G_1, G_2) \xrightarrow{or} G:$
$Opr(G) = \min\{Opr(G_1), Opr(G_2)\}$ (1)	$Opr(G) = \max\{Opr(G_1), Opr(G_2)\}$ (4)
$Achv(G) = \min\{Achv(G_1), Achv(G_2)\}$ (2)	$Achv(G) = \max\{Achv(G_1), Achv(G_2)\}$ (5)
$Prvn(G) = \max\{Prvn(G_1), Prvn(G_2)\}$ (3)	$Prvn(G) = \min\{Prvn(G_1), Prvn(G_2)\}$ (6)
G-G Symmetric Consistent Lateral Rels	*G-G Symmetric Conflicting Lateral Rels:*
$G_1 \xrightarrow{(M=)} G:$	$G_1 \xrightarrow{(M\times)} G:$
$Opr(G) = \min\{Opr(G_1), M\}$ (7)	$Opr(G) = Null$ (10)
$Achv(G) = \min\{Achv(G_1), M\}$ (8)	$Achv(G) = \min\{Prvn(G_1), M\}$ (11)
$Prvn(G) = \min\{Prvn(G_1), M\}$ (9)	$Prvn(G) = \min\{Achv(G_1), M\}$ (12)
G-G Asymmetric Consistent Lateral Rels:	*G-G Asymmetric Conflicting Lateral Rels:*
$G_1 \xrightarrow{+(M+)} G:$	$G_1 \xrightarrow{+(M-)} G:$
$Opr(G) = \min\{Opr(G_1), M\}$ (13)	$Opr(G) = Achv(G) = Null$ (18)
$Achv(G) = \min\{Achv(G_1), M\}$ (14)	$Prvn(G) = \min\{Achv(G_1), M\}$ (19)
$Prvn(G) = Null$ (15)	
$G_1 \xrightarrow{-(M-)} G:$	$G_1 \xrightarrow{-(M+)} G:$
$Opr(G) = Achv(G) = Null$ (16)	$Achv(G) = \min\{Prvn(G_1), M\}$ (20)
$Prvn(G) = \min\{Prvn(G_1), M\}$ (17)	$Prvn(G) = Opr(G) = Null$ (21)

$$\mathrm{Achv}(G_i) = \left(\mathrm{Achv}(G_i) \vee \bigvee_{r_j \in \mathscr{R}_{G-G_i}} \mathrm{Achv}_{r_j}(G_i)\right) \wedge \left(\bigwedge_{r_k \in \mathscr{R}_{c-G_i}} \mathrm{Achv}_{r_k}(G_i)\right) \tag{51.23}$$

$$\mathrm{Prvn}(G_i) = \left(\mathrm{Prvn}(G_i) \wedge \bigwedge_{r_j \in \mathscr{R}_{G-G_i}} \mathrm{Prvn}_{r_j}(G_i)\right) \vee \left(\bigvee_{r_k \in \mathscr{R}_{c-G_i}} \mathrm{Prvn}_{r_k}(G_i)\right) \tag{51.24}$$

3 Integrating the Agents' GCMs and Modeling the Agent's Strategic Priorities

3.1 Integrating the Agents' GCMs and Identifying Their Strategic Goals and Shared Alternatives

Once the agents' individual GCMs are captured, their collective alternatives should be identified. The alternatives, which will be modeled as intention nodes as we said earlier, could be reduced from the refined desire-type goals at the agent's goals

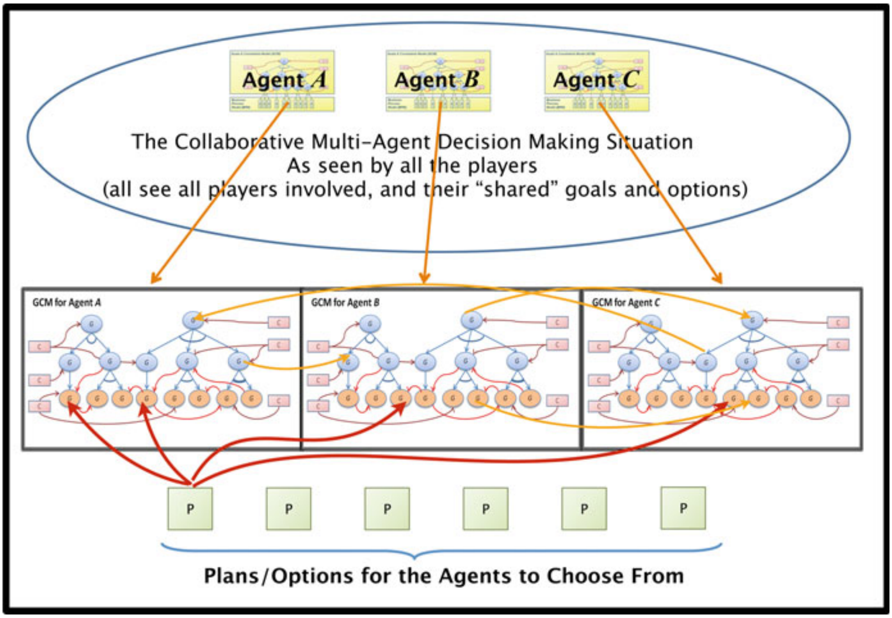


Fig. 51.3 Each shared alternative, the involved agents have in a multiagent collaborative decision-making situation, contribute positively/negatively to the agents' goals

tree/s, elicited directly from the agents themselves or identified through a process of brainstorming or creative thinking. The alternatives are then connected to the goals and constraints in the agent's GCM. Then, comes the step of integrating all the agents' GCMs. This is done by capturing the positive and negative effects that the different agents' goals and constraints have on each other's goals. The integration process is intended mainly to test and highlight the effect of each agent's GCM on the others' GCMs. At the of the integration process an integrated model, such as the one shown in Fig. 51.3 or the one shown for the example in Sect. 5 (Fig. 51.5), is produced.

Let each $DM_i \in \mathscr{DM}$, individually at time t of the decision-making situation, has a set of strategic goals $\mathscr{SG}_{D\,M_i,t}$ and $\mathscr{SG}_{DM_i,t} \subseteq \mathscr{G}_{DM_i,t}$, where $\mathscr{G}_{DM_i,t}$ is the set of all goals part of DM_i's GCM model $GCM\_Graph_{DM_i,t}\,\langle\, \mathscr{G}_{DM_i,t}\,, \mathscr{C}_{DM_i,t}\,, \mathscr{R}_{DM_i,t}\rangle$. What differentiate the goals in $\mathscr{SG}_{DM_i,t}$ from the rest of goals in $\mathscr{G}_{D\,M_i,t}$ is that the strategic goals are the aims/ends; DM_i is ultimately looking to achieve while the rest of the goals form the means. Also, the strategic goals in $\mathscr{SG}_{DM_i,t}$ are usually, but not necessarily, the top goals or root goals of the goal trees part of the GCM. And let for each $DM_i \in \mathscr{DM}$, where $\mathscr{DM}$ is the set of all decision makers in the collaborative decision-making situation, at time t ; let there be a set of strategic goals $\mathscr{SG}_{DM_i}$ (chosen by DM_i). Let the set of all strategic goals by all decision makers in $\mathcal{DM}$ is called the strategic goals of the collaborative decision-making situation and is denoted as $\mathscr{SG}$, where $\mathscr{SG} = \bigcup_{DM_i \in \mathscr{DM}} \mathscr{SG}_{DM_i}$. And, let the DMs in $\mathscr{DM}$

collectively decided on a set of shared alternatives $\mathcal{A}$ to choose one from, based on how much each alternative contributes to the achievement of all goals in $\mathcal{SG}$.

There are two important sets to be set up-front for each collaborative decision-making situation: (1) $\mathcal{SG}$: the set of all strategic goals, of all involved DMs and (2) $\mathcal{A}$: the set of all shared alternatives that DMs have.

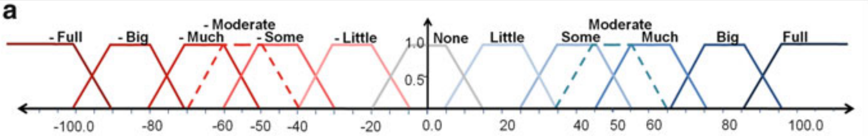
3.2 Modeling Final Achievement Levels of the Agent's Strategic Goals

The framework's forward propagation reasoning algorithm, which uses the above propagation rules to finalize the value labels for each goal node in GCM and proposed in Al-Shawa and Basir [2009] and Al-Shawa [2011], purposely keeps track of the achievement, operationalization, and prevention values of the goal nodes all separate from each other. It will not try to consolidate the value properties for each goal node to a single achievement value for the node, in order to highlight these values, and allow the DM/analyst to track what caused the values for each goal node in the model. But for the purpose of evaluating each alternative, a consolidated achievement level value is adopted to measure how good the alternative in helping the agent getting closer to his ultimate goals. Understandably, the final achievement value property for a strategic goal *SG*, denoted as FAchv(*SG*), should receive a value that represents subtracting its final prevention value (Prvn(*SG*)) from its final achievement one (Achv(*SG*)), taking in consideration the achievement upper limit that is set by both the constraints targeting *SG* and the level of operationalization *SG* managed to gain from the alternative the agent adopts. At time *t* and for $SG \in \mathcal{SG}_{D\ M_i,t}$, let the three final value labels that value propagation algorithm produces for *SG*'s operationalization, achievement, and prevention are denoted as Opr(*SG*), Achv(*SG*), and Prvn(*SG*), respectively.

$$F\ Achv(SG) = \begin{cases} Achv(SG) \ominus Prvn(SG) & \text{if } Achv(SG) \geq Opr(SG) \\ \\ Opr(SG) \ominus Prvn(SG) & \text{if } Achv(SG) < Opr(SG). \end{cases} \tag{51.25}$$

Let the fuzzy linguistic value label given to $F\ Achv(SG)$ based on the definition above is denoted as L_{FAchv} and is assigned by applying the " $\ominus$ " operator's table shown in Fig. 51.4b (implemented as reasoning rules). In other words, $F\ Achv(SG) = L_{\mathrm{F\ Achv}}$, where $L_{\mathrm{F\ Achv}} \in \mathcal{L} = \{Full, Big, Much, Moderate, Some, Little, None, -Little, -Some, -Moderate, -Much, -Big, -Full, Null\} = \{F, B, M, Mo, S, L, N, -L, -S, -Mo, -M, -B, -F, Null\}$. And, with the complete order of $F > B > M > Mo > L > N > -L > -S > -Mo > -M > -B > -F > Null$, where the labels range from representing *Full* goal achievement to *Full* goal prevention, covering the final achievement satisfaction level of 100 to $-$ 100 % and that the *Null* label represents an unknown achievement/prevention of the goal.

a



b

⊖	N	L	S	Mo	M	B	F	Null
N	N	−L	−S	−Mo	−M	−B	−F	Null
L	L	N	−L	−S	−S	−M	−B	Null
S	S	L	N	−L	−L	−S	−M	Null
Mo	Mo	S	L	N	−L	−S	−Mo	Null
M	M	S	L	L	N	−L	−S	Null
B	B	M	S	S	L	N	−L	Null
F	F	B	M	Mo	S	L	N	Null
Null	Null	Null	Null	Null	Null	Null	Null	Null

Fig. 51.4 (**a**) Fuzzy sets dividing the satisfaction levels of *FAchv* value property and (**b**) definition of $\ominus$ showing the resultant linguistic value label from the operation

The fuzzy membership functions defining these linguistic labels are given in Fig. 51.4a. As discussed in Al-Shawa and Basir [2009] and Al-Shawa [2011]), the fuzzy sets, shown in the figure to be trapezoidal in shape, should be defined based on the user needs.

3.3 Modeling the Strategic Importance of Agent's Goals

To model the priorities a DM might give to his strategic goals, we introduce the *Strategic Importance* value property. A property attached to each strategic goal node and given a fuzzy linguistic label that represents qualitatively the importance/priority of the strategic goal. For $DM_i \in \mathcal{DM}$, and at time t of the decision-making situation, let the *Strategic Importance* value property for a strategic goal $SG \in \mathcal{SG}_{DM_i,t}$ be denoted as $SImprt(SG)$. And, let $SImprt(SG_i) = L_{\text{SImprt}}$, where L_{SImprt} is a fuzzy linguistic label that represents the name of the fuzzy set which the strategic importance of SG, as assigned to it by DM_i, has the highest membership of. Let us assume $L_{\text{SImprt}} \in \mathcal{L}$, where $\mathcal{L}$ is shown the same value labels set used before (Fig. 51.2). Note that the fuzzy sets should be defined based on the user needs, and the total of all goals' strategic importance values do not have to add up to be 1 or *Full* (as AHP requires Saaty [1980]).

4 Calculating Agents' Preferences Over Their Shared Alternatives

For DM_i, to calculate the effectiveness of adopting an alternative over another, we calculate how much each alternative contributes to the final achievement of the strategic goals in $\mathcal{SG}_{D\ M_i}$. We introduce here the *Weighted Final Achievement* value property, attached to each strategic goals in $\mathcal{SG}_{D\ M_i}$, and provide a weight-adjusted Final Achievement value based on the importance given to the strategic goal. This value property will be represented as a numerical value, capturing the relationship between value properties represented by fuzzy labels. Therefore, we adopt the simple but effective Centroid defuzzification scheme. For DM_i, at time t , let the Weighted Final Achievement of a strategic goal $SG \in \mathcal{SG}_{D\ M_i}$, as a result of having alternative $A \in \mathcal{A}$ been adopted, to be denoted as *WFAchv* (*SG*, DM_i, *A*, *t*), be:

$$WFAchv(SG, DM_i, A, t) = \begin{cases} SImprt^*(SG) \cdot FAchv^*(SG) & \text{if} \quad SImprt^*(SG) \geq 0 \\ 0 & \text{if} \quad SImprt^*(SG) < 0. \end{cases} \tag{51.26}$$

where *SImprt* * (*SG*) represent the defuzzified values of *SImprt*(*SG*) and where *SImprt*(*SG*) $\neq$ *Null*, and it reflects the state of mind and beliefs of DM_i at time *t*. And where *FAchv* * (*SG*) represents the defuzzified values of *FAchv*(*SG*) and *FAchv*(*SG*) $\neq$ *Null*, and where "*A* was fully applied at time $t-1$" means that the intention to apply A was fully achieved, i.e., $Achv(A) = F$, at $t-1$.

For all the decision maker in $\mathcal{D}\ \mathcal{M}$ collectively, at time t of the situation, let the effect of the full joint application of alternative $A \in \mathcal{A}$ on all strategic goals in the nonempty $\mathcal{SG}$ is represented by a *Total Weighted Final Achievement* value property, and let this property value be denoted as $T\ W\ F\ Achv(\mathcal{D}\ \mathcal{M}, A, t)$ and be calculated:

$$TWFAchv(\mathscr{DM}, A, t) = \frac{1}{|\mathscr{SG}|} \sum_{\substack{SG \in \mathscr{SG}_{DM_i} \\ DM_i \in \mathscr{DM}}} WFAchv(SG, DM_i, A, t) \tag{51.27}$$

For the *Total Weighted Final Achievement* $T\ W\ F\ Achv(\mathcal{D}\ \mathcal{M}, A, t)$ value to reflect the effect of alternative *A*, and only *A*, on all DMs in $\mathcal{D}\ \mathcal{M}$, then the situation's integrated viewpoint with all its constructs and value properties' values must stay the same, and only *A* is applied fully. The achievement value of the intention node representing the intention to implement/apply alternative *A* changes from $Achv(A) = N$ to $Achv(A) = F$. All other alternatives have their respective intentions' achievement values stay the same unchanged, preferably unselected and stay at the *None* level, i.e., $(\forall A_k \in \mathcal{A} : A_k \neq A)\ Achv(A_k) = N$. Then, after the values forward propagation algorithm finalized the value labels for all goals, at t, we calculate $TWFAchv(\mathcal{DM}, A, t)$. The value of $TWFAchv(\mathcal{DM}, A, t)$, now, reflects the effect of applying alternative A, and only A, on all collaborating DMs in $\mathcal{DM}$. Equation 51.27 requires that the set of strategic goals that DM_i have must not be empty, i.e., it requires that $|\mathcal{SG}_{DM_i}| \neq 0$.

For all DMs in $\mathcal{DM}$, collectively, in a multiagent collaborative decision-making situation, at time t, let the *Cardinal Preference* that $\mathcal{DM}$ has over alternative $A \in \mathcal{A}$ be represented as a *Weighted Payoff* value property attached to A and be denoted as $WP(A, \mathcal{DM}, t)$. Let $WP(A, \mathcal{DM}, t) = TWF\ Achv(\mathcal{DM}, A, t)$. Based on the cardinal preferences, weighted payoffs, calculated for $\mathcal{DM}$ over each shared alternatives in $\mathcal{A}$, $\mathcal{DM}$ will have a *Preference VectorPref*$(\mathcal{DM}, \mathcal{A})$ showing the order of the alternatives in $\mathcal{A}$ from the *most preferred* to the *least preferred*. The preference order of a specific shared alternative $A \in \mathcal{A}$, collectively, to $\mathcal{DM}$ at t, is given as an *Ordinal Preference* value property attached to A and is denoted by $OP(A, \mathcal{DM}, t)$.

5 Example: Modeling a Multi-stakeholder Collaborative System Design Decision Making

To validate the Constrained Rationality framework and its suitability for collaborative multiagent system development initiatives, we used the framework to model and analyze two complex strategic industrial system development cases. The two initiatives were very successful but contractually confidential to discuss here. We will use here a simpler heavily scaled-downed version of one case as an illustrative example to show how the framework is used.

First, define the context: The purpose is to decide on the best architecture for a software system. There are three architectures to be reviewed for best fit (most accommodating to all the stakeholders' needs). Second, relevant decision makers: The software architecture for such small system will be decided on by four agents–two represent the system's business users, one system architect/designer (also represents the rest of the technical team) and one project manager (responsible for the contractual obligations, delivery timing, and reporting requirements). Third, build a viewpoint model for each decision maker: The analyst will acquire the knowledge needed to build a viewpoint model for each agent. For collaborative decision-making situations, the agents' individual viewpoints include mainly their respective GCMs.

Fourth, integrate all viewpoint models and finalize the base mode: The analyst will integrate the individual agents' viewpoints (GCMs), adding cause-effect qualified fuzzy-labeled lateral relationships between the different goal and constraint nodes reside in different GCM models.The result will be similar to Fig. 51.5.

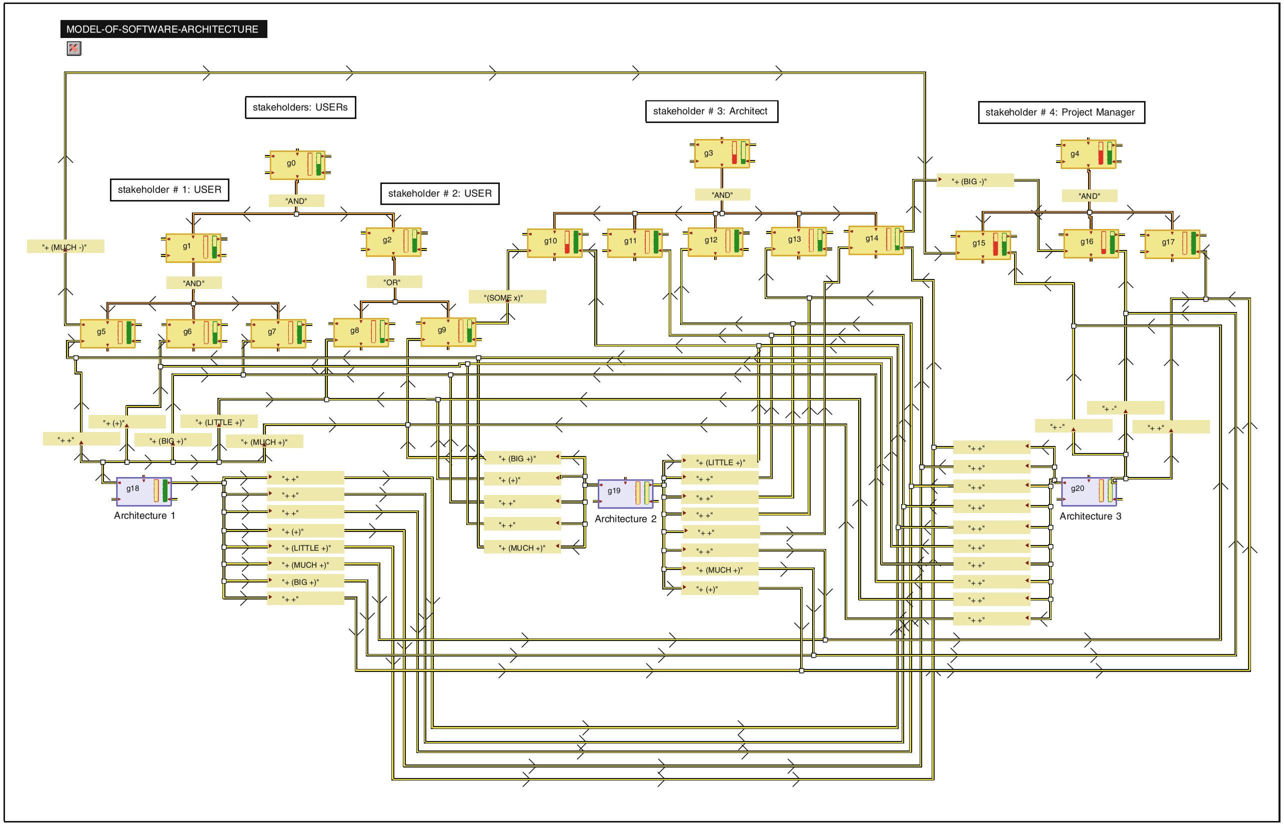


Fig. 51.5 Model of a collaborative system requirements decision making. Mobilizing an intention to implement architecture 1 will provide different achievement levels to the agents' goals

Note that the figure shows an integration of very simple GCMs, for a very simple system requirements example, with each agent has a GCM model with a two-layers goal tree, and that goals in some GCMs conflict with each other. In real-life requirements engineering initiatives, there are usually many more stakeholders, each with a complex GCM model that has multiple goal trees with goals conflict with others across the individual GCM boundaries. Also, Fig. 51.5 shows the three alternative architectures at the bottom, with each alternative connected to the agents' goals through lateral relations to show the positive/negative achievement/contribution that implementation of these alternatives have on the goals. In real-life requirements engineering initiatives though, there are many more alternatives.

Fifth, add priorities then generate agents' preferences over alternatives: In this step, the analyst will follow the same process used above to (1) capture the agents' individual strategic importance over each of their respective strategic goals and (2) elicit the agents' collective preferences over the shared alternatives. Figure 51.5 shows the effect (the achievement and prevention values) of implementing Architecture 1 on the agents' goals. Similar ones should be completed for the integrated model with each of alternatives 2 and 3 applied (Figures for alternatives 2 and 3 are not included for space constraint). After calculating the final achievement values for each strategic goal, for each architecture, the analyst can add the importance values for each of the agents' strategic goals and generate the group's preferences over these alternatives, as shown in Fig. 51.6. Finally, sensitivity and what-if analysis: The analyst will test variations to the value properties, the individual GCMs, or the integrated decision-making situation as a whole. In this simple example, one can notice that all three alternatives show some serious weakness. In such case, which is not far from what real-life requirements projects face, the group has to rethink their alternatives, come up with other creative alternatives, break the system/project to smaller systems/projects, and ask for a bigger budget and/or more resources.

6 Conclusion

In this chapter, we extended the Constrained Rationality's basic modeling and reasoning framework (Al-Shawa and Basir 2009). New modeling and reasoning facilities are proposed to effectively generate/verify the agents' collective cardinal and ordinal preferences (from their goals, constraints, and priorities) over their collective shared alternatives in collaborative decision-making situations. An illustrative requirements engineering application, a simplified version of an industrial case study we conducted, is used to demonstrate the concepts and methods proposed. The example shows the effectiveness and flexibility of the extended framework to model and reason about the system design options that stakeholders in a collaborative strategic system development decision-making initiatives have.

a

Strategic Goals of DMs	SGs:	Collaborative		
		$SG_{Collaborative}$		
		SG_{USERs}	$SG_{Architect}$	$SG_{ProjMangr}$
Architecture $A_{Arch\,1}$	$Achv(SG_k)$	Mo	L	M
	$Prvn(SG_k)$	N	S	M
	$FAchv(SG_k)$	Mo	-L	N
{ Achv($A_{Arch\,1}$)=F }	$FAchv^*(SG_k)$	0.50	-0.20	0.00
Architecture $A_{Arch\,2}$	$Achv(SG_k)$	M	L	Mo
	$Prvn(SG_k)$	N	S	B
	$FAchv(SG_k)$	M	-L	-S
{ Achv($A_{Arch\,2}$)=F }	$FAchv^*(SG_k)$	0.60	-0.20	-0.40
Architecture $A_{Arch\,3}$	$Achv(SG_k)$	F	F	N
	$Prvn(SG_k)$	N	S	B
	$FAchv(SG_k)$	F	M	-B
{ Achv($A_{Arch\,3}$)=F }	$FAchv^*(SG_k)$	1.00	0.60	-0.80

b

		decision making situation # 1			decision making situation # 2			decision making situation # 3		
		Collaborative			Collaborative			Collaborative		
Strategic Goals of DMs	SGs:	$SG_{Collaborative}$			$SG_{Collaborative}$			$SG_{Collaborative}$		
		SG_{USERs}	$SG_{Architect}$	$SG_{ProjMangr}$	SG_{USERs}	$SG_{Architect}$	$SG_{ProjMangr}$	SG_{USERs}	$SG_{Architect}$	$SG_{ProjMangr}$
Strategic Importance	$SImprt(SG_k)$	F	F	F	F	L	F	F	Mo	L
Architecture $A_{Arch\,1}$	$FAchv(SG_k, A_{Arch\,1}, t)$	0.50	-0.20	0.00	0.50	-0.20	0.00	0.50	-0.20	0.00
	$W(SGk,DM,t)$	1.00	1.00	1.00	1.00	0.20	1.00	1.00	0.50	0.20
	$WFAchv(SGk,DM,A_{Arch\,1},t)$	0.50	-0.20	0.00	0.50	-0.04	0.00	0.50	-0.10	0.00
	$WP(A_{Arch\,1}, DM, t)$	0.10			0.15			0.13		
{ Achv($A_{Arch\,1}$)=F }	$OP(A_{Arch\,1}, DM, t)$	2			1 (Best)			3(Worst)		
Architecture $A_{Arch\,2}$	$FAchv(SG_k, A_{Arch\,2}, t)$	0.60	-0.20	-0.40	0.60	-0.20	-0.40	0.60	-0.20	-0.40
	$W(SGk,DM,t)$	1.00	1.00	1.00	1.00	0.20	1.00	1.00	0.50	0.20
	$WFAchv(SGk,DM,A_{Arch\,2},t)$	0.60	-0.20	-0.40	0.60	-0.04	-0.40	0.60	-0.10	-0.08
	$WP(A_{Arch\,2}, DM, t)$	-0.00			0.05			0.14		
{ Achv($A_{Arch\,2}$)=F }	$OP(A_{Arch\,2}, DM, t)$	3(Worst)			3(Worst)			2		
Architecture $A_{Arch\,3}$	$FAchv(SG_k, A_{Arch\,3}, t)$	1.00	0.60	-0.80	1.00	0.60	-0.80	1.00	0.60	-0.80
	$W(SGk,DM,t)$	1.00	1.00	1.00	1.00	0.20	1.00	1.00	0.50	0.20
	$WFAchv(SGk,DM,A_{Arch\,3},t)$	1.00	0.60	-0.80	1.00	0.12	-0.80	1.00	0.30	-0.16
	$WP(A_{Arch\,3}, DM, t)$	0.27			0.11			0.38		
{ Achv($A_{Arch\,3}$)=F }	$OP(A_{Arch\,3}, DM, t)$	1 (Best)			2			1 (Best)		

Fig. 51.6 Results (**a**) final achievement levels for all architectures and (**b**) generating the preferences for the group

One drawback we experienced, using the framework in real-life complex initiatives, is that the models increase in their size and details as the number of agents increase. Therefore, as a future work, we are working on model management tools, templates, and metrics to help the decision makers and analysts be more effective in using the framework.

References

Al-Shawa M (2006a) Viewpoints-based value-driven—enterprise knowledge management (ViVD-EKM). MASc thesis, Electrical and Computer Engineering, University of Waterloo

Al-Shawa M (2006b) Using viewpoints to model and manage stakeholders' knowledge in enterprise domains such as healthcare. In: Proceedings of the eleventh international symposium on health information management research (iSHIMR 2006), Halifax, pp 352–368

Al-Shawa M (2011) Constrained rationality: formal value-driven enterprise knowledge management modelling and analysis framework for strategic business, technology and public policy decision making and conflict resolution. Ph.D. dissertation, Electrical and Computer Engineering, University of Waterloo

Al-Shawa M, Basir O (2009) Constrained rationality: formal goals-reasoning approach to strategic decision and conflict analysis. In: Proceedings of the 2009 I.E. international conference on systems, man, and cybernetics, Taipei

Dardenne A, van Lamsweerde A, Fickas S (1993) Goal-directed requirements acquisition. Sci Comput Program 20:3–50

Keeney R (1992) Value-focused thinking: a path to creative decisionmaking. Harvard University, Cambridge

Raiffa H, Richardson J, Metcalfe D (2002) Negotiation analysis: the science and art of collaborative decision making. Harvard University, Cambridge

Saaty TL (1980) The analytic hierarchy process: planning, priority setting, resource allocation. McGrawHill, New York/London

Shaw PH, Bordini RH (2007) Towards alternative approaches to reasoning about goals. In: Proceedings of 5th international workshop on declarative agent languages and technologies, Honolulu

Winikoff M, Harland J, Padgham L (2002) Linking agent concepts and methodology with CAN. In: Submitted to the first international joint conference on autonomous agents and multi-agent systems. ACM, New York

Wooldridge M (2000) Reasoning about rational agents. MIT, Cambridge

Chapter 52
CRM for SMEs

Ana Hol

1 SMEs in the Information Era

In the Information era, we depend greatly on information and communication technology (ICT). Day-to-day business operations rely heavily on how well data is stored, processed and retrieved. Furthermore, companies depend on electronic communication with their customers and business partners which often happens via e-mails, attachments, drawings and news bulletins. Therefore, to successfully communicate with customers, companies utilise Outlook contact details, e-mail exchange, spreadsheets, PDF files and associated data such as forms, drawings and images. To identify how best SMEs can communicate with their customers and to what extent existing CRM systems can assist SMEs as well as the reasons why CRM systems are not to a great extent used by SMEs, we conducted a case study analysis of five companies from various industry sectors. The investigation also included reviews of existing off-the-shelf CRM systems.

2 Investigations and Interview Analysis

To be able to identify what types of systems are required by the SMEs in order to successfully manage their relationships with the customers, we conducted a case study analysis with five SMEs ranging from 5 to 25 employees. The selected SMEs were from various industry sectors, namely, hospitality and tourism, manufacturing, sales, financing and service. Findings per company are listed below.

A. Hol (✉)
e-mail: a.hol@uws.edu.au

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_52,

– Boat Charter Company
  Currently, the company uses Outlook to store and manage communications with the customers. More detailed information about the customers is stored in spreadsheets. Contact details are recorded in paper diaries. The spreadsheets are not linked to the contacts, and therefore, it is very difficult to locate past customer data. It is also very difficult to search through existing records.
  The company would like to be able to better manage customer data. They would also like to be able to group customers and have the ability to send mass e-mails. In addition, the company would also like to have the opportunity to send their loyal customers special deals based on their past purchases.
– Financial Management Company
  The company uses Outlook, PDF files and electronic forms to keep data about the customers. Data appears to be disjointed, and it is difficult to keep a track of forms and PDF files. Company also conducts meetings face to face and via the phone and on occasions collates data electronically.
  The company feels that they would greatly benefit if they would be able to have one central location to store data for the customers. They would like to be able to store general data as well as files relevant to the particular customer within the same system. The company also states that they would like to be able to send customers' reminders, group customers and send individual customer greetings for their birthdays or special occasions. Moreover, they would like to have the ability to join data and records collected via phone meetings, e-mails and face-to-face meetings so that the data collected to date can be used in decision-making.
– Online Sales Company
  The company uses Outlook and reports from the online purchases to store data about the customers. Customer data is managed through a tree folder structure. Each customer has its own folder. All data that is linked to that customer resides within the customer-allocated folder. It is difficult to search through the recorded data as well as be able to identify the timeline when particular activities need to take place.
  The company would greatly benefit if they could store data about the customer within one location. The company also felt that it would be good if orders received could be allocated to staff so that each staff member within the company could know who is responsible for which customer. In addition, they also stated that they would prefer for the data to be centralised so that if the customer for their past purchases has been served by a different staff member, the current serving staff member would still be able to (if needed) access the data that is linked to that customer.
– Building and Design Company
  Data about the customers and their orders is recorded in multiple spreadsheets. Data for each customer is also associated with product drawings that are kept in a specialised drawing package format as well as a PDF file. There is no associated link between the files, and searching through files is difficult. It is also quite hard at present to keep a track of drawing's version changes and updates.

The company feels that it would benefit them if they had one central location where they could keep files for the customer. They also stated that they would like to be able to keep a track of discussions and changes to drawings. They also stated that it would be good if they could have a system where they could link various types of files for a customer with customer-specific requirements.

– Research Unit

  Data about the customers is saved in the spreadsheets. Associated documents about the customers are kept in hard copies. This makes it very difficult for searching.

  The company identified that they would greatly benefit if they could have an online system that would allow them to collect data about customers, fill out associated forms for the customers and have them saved under the customer records. They also stated that this would allow them to be able to search across a wide range of customer data records and once they are to find customers with a set of particular characteristics they would be able to identify a particular group so that they can undertake actions required for that particular customer group.

3 SME Requirements

Based on the data analysis, it was identified that a CRM system for SMEs would need to be able to:

– Store customer data – store customer contact data as well as specific data such as birthdays and customer interests.
– Send mass e-mails – allow companies to e-mail individuals as well as groups with special interests.
– Allow companies to create special deals which they would be able to send to their loyal customers.
– Collect customer documents – allow companies to collect various documents such as Word, Excel, images and PDF files for customers and search through them.
– Track communication with customers via e-mails, meetings and phone contacts.

4 Why Existing CRMs Are Not Sufficient

SMEs interact with customers via different mediums, including telephone, fax, company website and e-mail. Maintaining consistency in all customer interaction channels and areas, such as marketing, sales, after sales and servicing, ensures customer satisfaction (Pan and Lee 2003). However, CRM vendors have their own niches in different areas of the company and finding one CRM vendor to satisfy all customer touch points is proving to be extremely difficult. SMEs require a single view of the customer. Lack of coordination and separate customer interaction environments caused by using several CRMs make obtaining a single view of the customer and customer-centric data warehousing an enormous problem (Pan and Lee 2003). Therefore, existing CRMs need to be more flexible in data sharing or interfacing with other applications. They also need to realise that customer data should be made available in a common format rather than being in a proprietary database format where purchasing special software is required (Bsales 2006).

After considering company needs and their CRM vision, the SMEs need to select a best fit solution; however, the initial investment itself is expensive. The actual implementation for an on-site licensed CRM package can sum up to a prohibitive amount for SMEs (Business-Software 2010a). Most SMEs do not have dedicated IT personnel required to acquire necessary hardware, instal software, do adequate testing and carry on ongoing maintenance and periodical upgrades (Business-Software 2010a). For SMEs, continuous attempts at improving a particular set of business capabilities to minimise cost and increase its revenue are at the core of the company vision. It should be in line with the CRM vision or adoption of CRM best practices. Hence, it is a challenge for SMEs (Band 2008). Thus, whenever possible, CRM solution providers should offer SMEs to try out the solution themselves for a period of time to see if it's applicable for free or for a minimal fee. However, most current CRMs do not offer such minimum commitment, and SMEs find themselves legally committed to unsuitable CRM software that is a burden to the company.

After a successful implementation, some SMEs still face further issues. These issues are related to data security, data integrity, data loss, data availability and uptime (Business-Software 2010a). Data security is necessary for hosted or a web-based CRM solution. A data security breach may result from failure to protect customer data from threats like viruses or hackers by not using authentication, necessary firewalls, encryption or techniques such as SSL certification (CRM-Software-Guide 2010b). In between two updates of a particular stored data record, data alterations can take place, and consequently the inaccuracy and inconsistency of that stored data will lead to loss of data integrity (BusinessDictionary 2010). Moreover, even in the recent past, there had been incidents in hosted CRM solutions where unexpected outages or downtimes lead to a number of companies without access to their data for sometime (Bsales 2006). Thus, CRM solutions need to be tested more for data security vulnerabilities, and at all times, they should take necessary security measures.

In addition to the technical issues, there are times when trust issues come to the forefront. Many SME owners are not comfortable in having to store their valuable customer data off-site in a hosted server (Bsales 2006). When the hosting CRM provider does not provide stringent security measures, valuable customer data could easily fall on to competitors hands (Business-Software 2010a). Even in an on-site CRM system, security breaches, unexpected outages or threat of an employee walking out with customer data exist (Bsales 2006). Thus, small business CRM vendors should make more effort to protect client's customer data to build trust with their clients.

Often, CRM failures result due to inadequate training of the software product. As CRM relies heavily on data inputs by sales, after sales and IT teams, employees who are not versatile with CRM solutions damage the purpose of having one in the first place. Thus, CRMs should offer more free eTutorials as well as live customer support (CRM-Software-Guide 2010a). Additionally, if the CRM software is hard to comprehend or subsequent updates change system look and feel constantly, employees become more and more reluctant to use the system (CRM-Software-Guide 2010a).

A growing business should always consider future needs and determine how to support such growth through CRM software solution that grows with them. When a business grows, SMEs may want to port from a hosted solution to an on-site solution, or they may want to change from an inflexible solution to a more flexible one which may cater for their future needs. However, handling this transition for an SME is not easy and is not generally currently supported by CRMs (CRM-Software-Guide 2010a). Thus, existing CRMs need to realise the SME capabilities and be more scalable in their offerings.

According to Business-Software (2010b), a small business CRM solution should be less costly, easy to implement, scalable by being flexible to changing needs and be user-friendly enough to cut down on cost and time in training and administration. However, all of these requirements are not always met by CRM vendors. Nevertheless, small business CRM would be inadequate if they are unable to meet the above-mentioned requirements.

The following table shows a summary of currently popular small business CRM systems including their business model and prices (Table 52.1).

Even though the above-mentioned CRMs are in existence, according to the conducted study, most SMEs are not readily accepting existing CRMs as they do not seem to answer current SME requirements in an efficient, affordable and user-friendly manner.

5 SME–CRM Prototype

Based on the data analysis of the interviewed companies, it has been identified that the following would be essential for a CRM system for SMEs to have.

Firstly, it would be important for the CRM for SMEs to allow for the ability to save company and/or individual customer data. This would mean that the contacts can be individuals and/or companies depending with whom the organisation is doing the business with. The proposed screen for managing company records is shown in Fig. 52.1.

Therefore, it has been established that for each customer, an organisation would need to have the following: customer contact information (Fig. 52.2), the most appropriate way of communication, date of customer's birth (if required), customer interests and a person responsible for the customer as well as all queries a customer has made. Each query would represent a task that an organisation would need to

Table 52.1 A review of the existing SME–CRM systems

Name	CRM applications	Business model	Products	Customer focus	Price
Commence	Integrated suite	Hosted, on premise	Commence Desktop CRM	SMEs of 20 to 250 users and divisions of large corporations	Starting at $30/user/month with a minimum of five users
	Account and contact mgt, calendaring, marketing, sales automation, lead management, quoting, activity mgt, project mgt, customer service, reporting, e-mail and mobile device integration	Rapidly	deployable to fully integrated Commence CRM On-Demand		
Requires no hardware, no software and no IT	infrastructure cost				
Maximizer	Fully integrated, sales, marketing and customer service and support management solution, integration with Microsoft Office, mobile access	On premise	Maximizer CRM 10	SME and divisions of larger enterprises	Starting at $229
Microsoft dynamics	Customisable CRM application	Hosted and on premise	Microsoft Dynamics	Outlook users in SMEs	Professional Edition – $622 and $880 per user Small Business Edition – $440 and $499
NetSuite	Full CRM feature set Integration with Microsoft Outlook	Hosted	NetSuite, NetCRM NetERP, NetSuite Small Business NetCommerce	SMEs	NetSuite is $499/month base fee and $99/user/month
Workbooks	Suite of user-friendly business applications. Web-based CRM solution. SaaS, sales	Hosted	Workbooks CRM Workbooks Business	SMEs	From £35/user/month

	force automation, marketing automation, sales order management, customer service and support, purchase order management, invoicing and cash collection, company management				
Sage	Sales, internal sales, customer care and marketing information. Wireless and Internet-based CRM system	Hosted and on premise	Sage CRM SageCRM.com ACT	SMEs	Starting at $1,495 per server and $495 per user Hosted service $69/user/month
eSalesTrack	For a hosted CRM application offers, the best value in features and functionality	Hosted	eSalesTrack	SMEs	$40/user/month
Ivinex	Sales and customer service, sales forecasting	Hosted	Ivinex Ultimate Edition	SMEs and large enterprises	$60/user/month
SAP	Offers complete suite of CRM, integrated with other business application software in marketing, sales, service, e-commerce, channel management	Hosted and on premise	MySAP CRM	Midsize and large enterprise	Not published
ClaritySoft	Account and contact management, opportunity management and forecasting, quotations, task management, shared calendar, e-mail marketing, data segmentation – for target marketing, easy customisation, end user management	Hosted and on premise	ClaritySoft CRM	SMEs	$26/user/month (3 year average)
		Hosted	Salesnet CRM	Medium, enterprise	

(continued)

Table 52.1 (continued)

Name	CRM applications	Business model	Products	Customer focus	Price
Salesnet CRM system for the next decade	Intelligent pipeline management, smart forms, patented sales process workflow engine, colour-coded task list, Office and Outlook integration, query data				For 50 users or more, $99/user/month
ADAPT	Marketing campaign management, real-time accounting, advanced sales automation, complete customer services, e-mail methods	On premise	ADAPTcrm	Midsize	$995 per named user and $1,495 per concurrent user
Oracle	Sales, marketing and e-commerce, customer service and analytics	Hosted, on premise	Oracle E-Business Suite, Siebel CRM, PeopleSoft Enterprise CRM, JD Edwards EnterpriseOne CRM	Enterprise, SMEs	Starts at $70/user/month
Salesforce.com	Sales force automation, marketing automation and customer service and support automation	Hosted	Contact Manager Edition Group Edition Enterprise Edition Professional Edition Unlimited Edition	Enterprise, SMEs	Contact manager starts at $5/user/month, group starts at $25/user/month, professional starts at $65/user/month, enterprise starts at $125/user/month, unlimited starts at $250/user/month

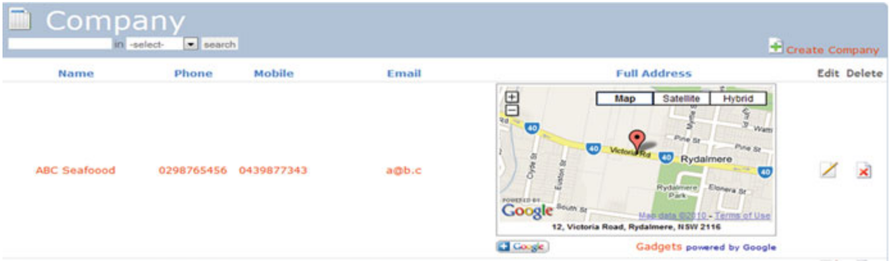


Fig. 52.1 Company details management screen

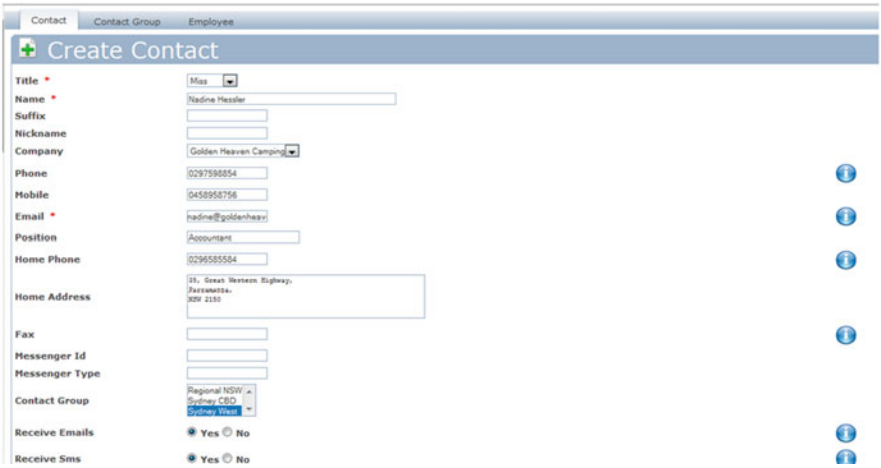


Fig. 52.2 Customer contact details creation screen

attend to. With each task, there would also be associated communications. The communications with the customer could consist of e-mails, face-to-face or online meetings and phone conversations. Each meeting could then be summarised and main actions and results of the meeting outlined and tasks allocated to the required individuals within the company. In this way, each employee would be able to see tasks they are responsible for. It can be noted from Fig. 52.3 that these communications are easily tracked based on cases, and in addition tabbed browsing allows for easy grouping.

Furthermore, for each customer, the company would be able to manage communications and, at each time, know exactly how the communication followed with each customer. In addition, this would also identify a timeline of conversations and actions that have been taken when dealing with the customer. Moreover, this would furthermore allow for the conversations to be linked with associated documents which could include PDF files, Word documents, spreadsheets and voice recordings as well as drawings and images.

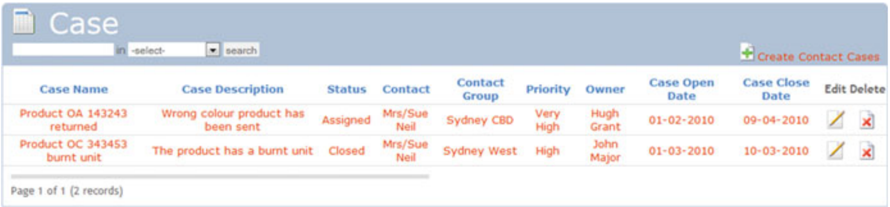


Fig. 52.3 Customer case management screen

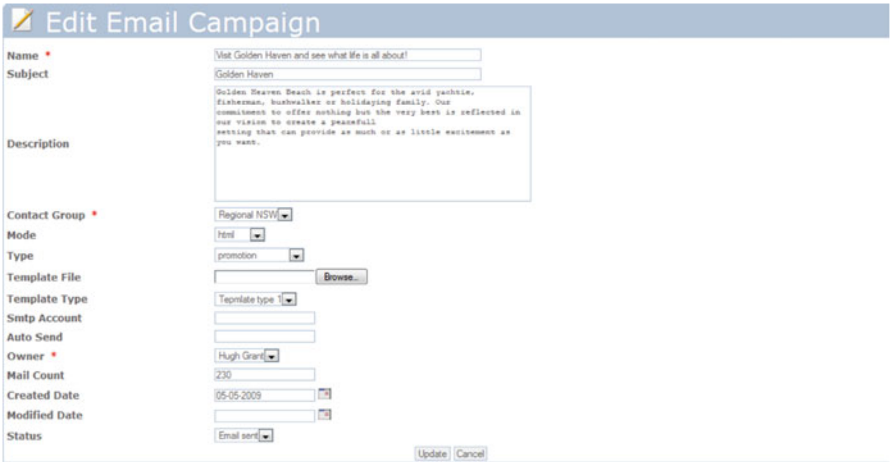


Fig. 52.4 E-mail campaign management screen

The e-mail campaign management function within the system allows the company to send e-mails via their mail server to the customers and keep records of communications linked to each contact (Fig. 52.4). The e-mailing function also allows the company to send e-mails to individual customers or to groups of customers. Furthermore, it allows for the customers to be grouped. Grouping can be done based on customer interests and or past purchases. The company can then perform mass e-mailing and keep in touch with particular groups or certain individuals.

To be able to streamline group e-mailing and campaigns management, it has also been identified that SMEs would benefit if they could create special deals. Therefore, within the system, a business can create a deal and, based on deal specifications, identify to whom this deal should be sent to.

Moreover, to be able to manage company effectiveness, it has been identified that it would be good to have the ability to collect information from customers and receive feedback. Therefore, it has been decided that a function should be created so that generic feedback forms can be uploaded and via mass e-mailing function sent to specific customer groups. This function can be implemented by the screen shown in the Fig. 52.5.

Fig. 52.5 Feedback form management screen

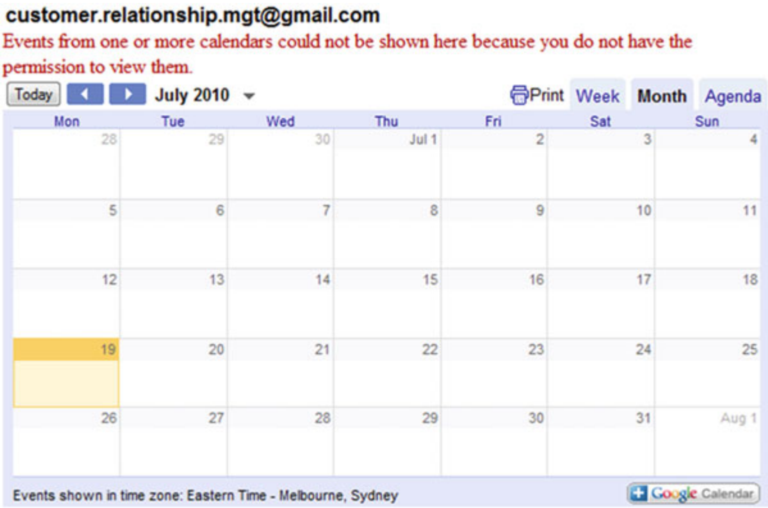


Fig. 52.6 Calendar management screen

Thus, each customer communication conducted through all communication channels can be followed by the proposed system. Customer data can easily be searched through and customers with particular interests can be identified. To further manage communications and identify activities that need to be undertaken for each customer, a calendar management function embedded into a Google account has been created so that the company can holistically keep a track of all activities that need to be done within the whole company by the use of a shared calendar (Fig. 52.6).

6 Conclusion

In summary, it can be seen that existing CRM systems are too large and cumbersome for SMEs. They have too much functionality and do not provide straightforward interfaces that SMEs could easily adopt. Most require in-depth studies, and SMEs do not have the time nor the resources which they can allocate to study CRM packages. Even when these do happen on rare occasions, it appears that SMEs do not adapt CRMs as they have many functionalities that are not used by the SMEs. Detailed studies of five SMEs helped identify exact functionalities SMEs from various industry sectors required to be able to keep in touch with their customers. This also enables companies to provide up-to-date information and assure customers are attended to and are satisfied with the provided service. Therefore, based on the research conducted, we developed an SME–CRM prototype that can now be used by SMEs from various industry sectors. Currently, the system is being tested, and once testing is completed, the system will be available for use. In the future, we expect to conduct a detailed study of SME–CRM effectiveness across industry sectors.

References

Band W (2008) CRM best practices adoption – for business process & applications professionals

Brown S, Gulycz M (2006) Performance-driven CRM: how to make your customer relationship management vision a reality. Wiley, New Delhi

Bsales J (2006) The CRM Conundrum. Retrieved 11 July 2010, from http://itmanagement.earthweb.com/entdev/article.php/3601431/The-CRM-Conundrum.htm

BusinessDictionary (2010) Data integrity. Retrieved 10 July 2010, from http://www.business-dictionary.com/definition/data-integrity.html

Business-Software (2010a) Top 10 hosted CRM vendors revealed – 2010

Business-Software (2010b) Top 10 small business CRM vendors revealed – 2010

Caretsky L (2010) Don't make a CRM buying mistake – 7 points to consider before selecting your CRM solution

CRM-Software-Guide (2010a) Top 7 CRM software mistakes to avoid – the pitfalls of CRM best practices. Retrieved 14 July 2010, from http://www.crm-software-guide.com/crm-software-mistakes.htm

CRM-Software-Guide (2010b) Web-based CRM solutions and customer data security management. Retrieved 2 June 2010, from http://www.crm-software-guide.com/web-based-crm-solutions.htm

Pan SL, Lee J.-N (2003) Using E-CRM for a unified view of the customer

Chapter 53
Perceptions of Low-Cost Carriers' Compliance with EU Legislation on Optional Extras

Chris Barry, Mairéad Hogan, and Ann Torres

1 Introduction

As the base price of flights has reduced, airlines are improving their revenue through ancillary services. At the same time, the European Union and other governments are regulating the industry in order to protect the consumer from inappropriate commercial practices. This regulation has resulted in some airlines finding innovative ways to ensure the consumer will at least consider purchasing some of these ancillary services while still remaining compliant with the legislation. This chapter examines user perceptions of the level of compliance of two Irish airlines with the relevant European legislation.

2 Literature Review

2.1 Low-Cost Carriers' Influence Within the Airline Industry

The airline industry has evolved through three 'waves of dramatic change', which not only restructured the industry but also radically affected consumers' travel behaviour (Rubin and Joy 2005:215). The first wave was deregulation from the late 1970s through to 2000s; the second wave was consolidation during the late 1980s; and the third wave of change is evident by the market power wielded by low-cost carriers (LCCs), which has instituted changes to the industry's ticket

C. Barry (✉) • M. Hogan
Department of Business Information Systems, National University of Ireland, Galway, Ireland
e-mail: chris.barry@nuigalway.ie; mairead.hogan@nuigalway.ie

A. Torres
Department of Marketing, National University of Ireland, Galway, Ireland
e-mail: ann.torres@nuigalway.ie

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_53,

purchasing, route patterns and competitive structure (Graham and Vowles 2006; Rubin and Joy 2005).

Advances in technology have revolutionised the marketing, selling and procurement of tickets through the Internet. The prevalence of LCCs has been a catalyst for the development of ‘low-cost airports and of low-cost airport facilities in general’ (de Neufville 2008:37). This collection of changes to long-standing aviation practices has resulted in a sharp focus on operation costs. The LCCs healthy financial performance is largely attributed to improved cost savings rather than differences in revenue management practices (Shumsky 2006). The LCCs’ impact on altering consumers’ price expectations is significant. Consumers ‘have widely changed their buying criteria, preferring price and convenience over extensive connectivity and seamless travel’ (Franke 2007:24). Thus, cost reduction is believed to have become a permanent requirement for profitable airlines (Doganis 2006).

2.2 Ancillary Revenues and Regulation

LCCs’ commitment to cost reduction means they examine every function to eliminate those considered superfluous or to charge for them separately as ancillary services. To bolster profit, LCCs have become adept at generating ancillary revenues such as baggage fees, now the largest single source of extra fees (Harteveldt and Stark 2010). Airlines have also partnered with third-party vendors add ancillary revenue generated through car rental and hotel bookings (Sorensen 2011). During 2009, ancillary revenues accounted for 22.2 % of Ryanair’s total revenue and 14.4 % of Aer Lingus’s total revenue (Amadeus and IdeaWorks 2010). Global airline ancillary revenues for 2010 were estimated at €18.5 billion, which represents a 68 % increase from 2009. It is anticipated that airlines will grow their sales of ancillaries appreciably, which are expected to increase by 300 % to €74.8 billion in the next few years (Amadeus and IdeaWorks 2010).

In an effort towards greater price transparency, the US Transportation Department is requiring airlines to release more detailed information on ancillary fees. In 2011, the US Bureau of Transportation Statistics will publish these ‘airline-imposed fees’ to address concerns of ‘corporate travel managers and consumer groups [who] have been complaining that all the new fees – including those for checked bags, priority boarding and in-flight food and drinks – have made it increasingly difficult to calculate the real costs of travel’ (Sharkey 2011:1). Similarly, hidden costs and the exclusion of charges that are unavoidable are becoming an increasingly contentious issue that has attracted the attention of EU bodies. The European Consumer Centres Network (ECC-Net 2010) recognises the information airlines provide on different price elements is incomplete.

The EU Commission coordinated the airline ticket selling investigation under the auspices of Consumer Protection Cooperation Regulation, which had came into force in 2006 (ECC-Net 2007). The investigation identified the most common unfair practices related to price indications, availability of special offers and contract terms. With respect to clear pricing, the commissioner directed airlines to give a clear indication of the total price, including taxes and booking/credit card fees in the headline price first advertised on a Website, rather than at a late stage in the booking process. Accessibility of special offers was of particular concern, as in many cases, these offers were not available or extremely limited. Other unfair practices were found to include mandatory insurance attached to an offer, or where consumers were required to explicitly opt-out of insurance or other optional services (e.g. priority boarding and seat selection). Failure to comply with the report's recommendations may result in legal action or closure, as well as being 'named and shamed' for failing to bring their Websites in line with EU law. It is this last recommendation on optional services that is the subject of this study.

The European legislation governing airlines' price information are Articles 5–7 of the Unfair Commercial Practices Directive and Article 23(1) of Regulation 1008/2008. The key provisions on how optional services should be presented to the user frame the research objectives of this study. The implications to the consumer of these pieces of legislation in relation to optional services are as follows:

- All optional extras must be communicated in a clear, transparent and unambiguous way at the start of the booking process.
- All optional extras should only be accepted by the consumer on an 'opt-in' basis.
- An airline may not engage in aggressive commercial practices (e.g. using harassment or coercion) that significantly cause a consumer to make a transactional decision that he would not have taken otherwise.

Thus, we derive three research questions to test.

- RQ1: At the start of the booking process, were all optional extras communicated in a clear, transparent and unambiguous way?
- RQ2: Are all optional extras presented as opt-in options?
- RQ3: Did users feel harassed or coerced into choosing optional extras during the booking process?

2.3 Opaque Web Design

With respect to the operational management of LCCs, securing resources and developing competences in managing e-business tools have become crucial (Nucciarelli and Gastaldi 2008). The LCCs' adoption of technology in areas such as electronic ticketing and dynamic pricing has become an important component in offering consumers more efficient flight options. Yet despite these advances, it appears a number of LCCs use their information systems in a conflicting manner when managing customer interactions. The Websites for many LCCs smoothly engage and facilitate customers through the self-service process to commit users to purchase tickets. However, once users move beyond the 'committal' point (i.e. after they have chosen where and when they wish to travel and have received an initial quote), the Websites appear more opaque. Research on this phenomenon has found significant disquiet amongst users (Torres et al. 2009).

2.4 Defining Opt-in and Opt-out

Rather than the more usual opt-in/opt-out mechanisms used to offer ancillary services, airlines are using a new approach, referred to here as a 'must-opt' selection. In this format, the user must explicitly accept or reject the service before continuing with the interaction. According to Wiktionary, opt-in means 'Of a selection, the property of having to choose explicitly to join or permit something; a decision having the default option being exclusion or avoidance' whereas opt-out means 'Of a selection, the property of having to choose explicitly to avoid or forbid something; a decision having the default option being inclusion or permission'. Based on the first part of these definitions, the airlines' must-opt optional extras are both opt-in and opt-out as the user must choose to participate if they wish to do so and if they do not. However, based on the second part of the definitions, the must-opt can be seen as neither opt-in nor opt-out, since the default option is to prevent the user from continuing until they either accept or refuse the option.

3 Research Methodology

Verbal protocols involve a typical user thinking out loud while carrying out representative tasks on a system. While carrying out the tasks, the participants explain what they are doing and why (Monk et al. 1993). This verbalisation aids in understanding the user's attitudes towards the system to identify aspects of the design that are problematic for the user (Benbunan-Fich 2001; Holzinger 2005). One of the main strengths of this technique is 'to show what the users are doing and why they are doing it while they are doing it in order to avoid later rationalizations' (Nielsen 1993:196). The role of the evaluator is to support the participants by prompting and listening for clues rather than taking over or giving instructions (Shneiderman and Plaisant 2010). Therefore, it is important to ensure the prompts used do not distort or invalidate the user's dialogue. Developing prompts that facilitate in the collection of the type of data needed and using them at appropriate times will help to ensure this (Cotton and Gresty 2006). In this study, verbal protocols were used to determine the opinions of 20 typical users on how compliant two airlines, Aer Lingus and Ryanair, are with the European legislation regarding the sale of flights using the Internet. In a previous study (Torres et al. 2009:306), a participant stated 'there will always be regulation, but [LCCs] will always find a way around it'. This study is interested in determining the truth or otherwise of this statement.

4 Findings

4.1 RQ1: At the Start of the Booking Process, Were All Optional Extras Communicated in a Clear, Transparent and Unambiguous Way?

There was unanimity that neither airline was in compliance with the provision that at the start of the booking process all optional extras should be communicated in a clear, transparent and unambiguous way. At the flight selection screen with Ryanair, the message 'Optional charges such as administration and checked baggage fees are not included' appears immediately beneath the total flight cost. Despite its proximity to the flight cost, most did not see the message. Participants broadly found this message unhelpful in clarifying specific optional charges. Nine other optional charges or services are not mentioned. A 'detail' link invokes a pop-up window where fees (not optional extras) are explained. Some participants thought the pop up acceptable in presenting detail of optional charges, others felt the information value was lost, so extensive and superfluous was the content. With Aer Lingus, no indication other than a handling charge was displayed on the initial selection page. Even that was at the bottom of the screen and greyed out, making it difficult to see. On the next screen, many participants expressed severe annoyance that the total flight price had changed with the addition of a handling fee that was wholly unexplained. Many felt optional extras were introduced incrementally so that the price would change gradually and users would be less likely to back out of the process. In this regard, participants felt such design to be deliberate.

4.2 RQ2: Are All Optional Extras Presented as Opt-in Options?

4.2.1 Case 1: Ryanair

On Ryanair, this question dealt with the 'Services' pages and generated enormous confusion, ignominy and anger. Ten decisions on optional extras (see Table 53.1) had to be negotiated (travel insurance being presented for decision twice). Seven of them are what the authors described earlier as 'must-opt' because while neither an opt-in nor opt-out decision, a forced interaction is required. On Ryanair, the mechanism for enforcing a must-opt is a pop-up error window when the 'Continue' button is pressed detailing the option that has not been selected. Barring terms and conditions, no indication is given that any must-opt demands an interaction on the part of the user. The remaining three options (i.e. baggage, sport equipment and special assistance) are opt-in and can be bypassed without any required interaction. In general, participants were unclear and wary about the nature of options and spent some time reading them in order to avoid choosing them. At the outset, participants

Table 53.1 Ancillary services categorised

	Airline			
	Ryanair		Aer Lingus	
Option presented as	Option	No.	Option	No.
Opt-in	Baggage Sports equipment Special assistance	3	Flex fare (1) Flex fare (2) SMS confirmation Special assistance Voucher Baggage Extra baggage weight Sports equipment Lounge	10
Opt-out	–	0	Mailing list	1
Must-opt	Priority boarding Travel insurance (1) SMS confirmation Ryanair-approved cabin bag Terms and conditions Travel insurance (2) Hertz rent-a-car	7	Terms and conditions Travel insurance Parking	3

were asked what they understood by an opt-in and opt-out selection. Nearly all defined opt-in and opt-out in a way that was consistent with the definitions in the literature review. There was clarity and satisfaction amongst participants that the bag option was actually opt-in. Priority boarding presents the first must-opt decision point. Five participants felt it was actually an opt-in decision, but as their comment was queried, they began to contradict their initial view. More felt it was either opt-out or at least forcing them to make a decision, coming close to the definition earlier that opting-out was explicitly choosing to avoid or forbid something. Others felt confusion about the exact nature of the interaction with comments like 'it's making me choose', 'it's forcing me to make an option', and 'it's making me read through it.' In summary, there was unease at the design of priority boarding, and the predominant view was that the option was either not in compliance with the law or not in compliance with the spirit of the law.

Travel insurance caused a great deal of resentment and anger amongst participants. The design is at first curious and ultimately devious. The user is invited to buy travel insurance but not with Yes/No radio buttons or a check box, rather a drop-down list with the default option to 'Please select a country of residence', a supposition that the service has already been chosen. Users were drawn to the alphabetically ordered drop-down list to look for an avoidance mechanism. About halfway down between Latvia and Lithuania was a 'country' called 'No Travel Insurance Required.' Most did not notice the line beneath the drop-down list that informed participants that 'If you do not wish to buy insurance, select No Travel Insurance in the drop down menu.' This feature was described in trenchant terms by participants such as 'it's buried into the drop down list' and 'this should be illegal!' It drew a torrent of adjectives such as underhand, sneaky, aggressive, extremely dodgy, tricky and deceptive. Participants wholly agreed the design was intentional and was not in compliance with the legislation. For the SMS optional extra, presented with two unchecked radio buttons, there is no indication that it bears any cost. Identified by several participants as a useful feature, none, however, felt the charge appropriate. While in fact a must-opt, several felt it was opt-in, some that it was opt-out and others that it was neither opt-in nor opt-out. Participants did not express strong emotions on compliance – although more felt it was not compliant.

The approved Ryanair cabin bag elicited a great deal of emotional reaction. Participants felt most charitably that it was marketing gimmickry and at worst pushy and aggressive marketing. What seemed to annoy them most was that it was delaying and distracting marketing in the middle of the booking process. A few felt it gave the impression it was an official cabin bag. On sports equipment and special assistance, general comments were benign and participants quickly realised they were genuine opt-in decisions. Ryanair use a must-opt check box for terms and conditions, which is a widely used convention in transactional activity and for registration purposes.

There was surprise and annoyance when a pop-up page reminds the user to make selections on the must-opt decisions that were overlooked: 'it is making me read all the options', '[it] didn't say it was compulsory to answer travel insurance', 'you should not be forced to decide.' A further inconvenience for users is that going back a screen involves re-entering all decisions except the passenger's name. On declining the must-opts and choosing 'Continue' on the main services page, a pop-up must-opt decision for travel insurance appears. Users are implored to 'Wait!' and asked if they are prepared to take the risk of not taking out travel insurance. Most felt the reminder was pushy and not in compliance as an opt-in decision. Two felt it helpful to be reminded to check they have travel insurance. Once the much smaller 'no thanks' button is clicked, participants may proceed.

The second service page is devoted wholly to selling a Hertz rent-a-car. The first reaction of participants was about the presumption of having the first named traveller preselected as the main driver and one of three cars highlighted, suggesting it has already been chosen. One commented 'Oh my God – it looks like they rented a car for me!' At this point, participants were becoming irritated and exhausted; one wearily commenting that there's 'a lot of reading to make sure I don't miss something.' Participants felt such heavy selling was inappropriate in the middle of the process of booking a flight and that it would be better placed after the flights were reserved. The general view was that it was non-compliant.

4.2.2 Case 2: Aer Lingus

On Aer Lingus, users navigate through four screens that deal with the booking process up to card payment. Fourteen decisions on optional extras had to be negotiated (see Table 53.1). Of these, three are must-opt, one opt-out and the remainder opt-in. Participants defined the concepts of out-in and opt-out consistently with the definitions in the literature review.

The ‘flex fare’ optional extra (i.e. offering free date changes and lounge access) on the first page was agreed by participants to be opt-in. Some felt it a bit pushy as it was directly embedded within the booking process. On the next page, the user encountered another invitation to choose a flex fare for each leg of the journey. It was not immediately obvious to participants that this was the same flex option from the previous screen. Many felt irritated they were expected to consider the option twice. At the end of the page, Aer Lingus also uses a must-opt check box for terms and conditions to confirm they are read.

Participants all agreed that SMS was opt-in, and there were no significant issues. They felt it consistent with their definition at the outset. One reported ‘it’s opt-in, that’s the default. You have to knowingly choose it.’ It was also positively noted by several that the price of an SMS was shown before it was selected. Participants felt the mailing list opt-out option to be very unclear and ‘tricky’ since previous optional extras were opt-in. Several expressed dissatisfaction with the way it was designed, immediately beneath the SMS option in lighter grey text, and header-less. Three opt-in options (frequent flyer, special assistance and voucher submission) follow that are easily bypassed by participants. These drew no comment.

Most participants felt baggage was compliant as it was a clear opt-in decision. However, in order to see the price, participants had to click a drop-down list. It was mentioned that this was still preferable to Ryanair’s design whereby the bag needed to be selected before a price appeared. In this regard, it was deemed compliant. Travel insurance is presented in a most confusing manner and invoked some annoyance amongst participants. The amount appears preselected in a right-hand column but not included in the total price, further down the page. In fact, two Yes/No options appear with unchecked radio buttons (thus a must-opt) There was a lot of disagreement on how to actually describe the decision. While few believed it was opt-in, participants were fairly evenly split as to whether it was opt-out, neither or that they remained unsure. Many changed their minds during the interaction. The confusion was even more exaggerated than in the case of Ryanair because the cost is preselected alongside other charges to date. Participants dealt with opt-in lounge access speedily. Parking was presented as a must-opt selection, displayed identically to the travel insurance. Thus, the cost was shown in the right-hand side of the page as if it had already been selected. By the time participants dealt with the task of contemplating parking, many were weary. They felt once again they had to read the text carefully to avoid the charge and then had to tell Aer Lingus they did not want it. Some felt it was presumptuous to ask them about parking. When asked why it and travel insurance were designed in this way, one participant cynically observed ‘they prioritise what they want to sell and then choose different technology [to sell it].’ On compliance, participants were generally of two views on Aer Lingus’s travel insurance and parking options: that they either wholly contravene the legislation or it is designed to get around it.

4.3 RQ3: Did Users Feel Harassed or Coerced into Choosing Optional Extras During the Booking Process?

4.3.1 Case 1: Ryanair

Decisions on baggage, SMS, sports equipment and special assistance presented no real problems. There were, however, a few concerns over priority boarding, namely, the way in which it was phrased and presented. The inclusion of the option to purchase a Ryanair-approved cabin bag was far more problematic. Participants felt it inappropriate and out of place, describing it as pushy and annoying. A couple felt some coercion in having to ensure that their bag would be acceptable to Ryanair. With travel insurance, when presented with the design described earlier where the option to avoid insurance is halfway down a drop-down list, many of the participants were annoyed and felt that it was not an acceptable way to design the interaction. Several felt it harassing and deceptive. On declining travel insurance, the pop-up reminder was widely cited by participants as aggressive and by many as coercive and harassing. Strong emotions were also evinced on the design and colour choices that suggested danger or risk for users declining travel insurance. Participants agreed Ryanair was non-compliant on this question.

The Hertz rental page was deemed distasteful by most participants. Views ranged from considering it pushy but compliant to aggressive and non-compliant, to extremely aggressive and non-compliant. Comments included 'a little bit aggressive and harassing at this stage', 'this is aggressive selling. It's moving towards coercion. [They] pre-selected my name!', '[it's] aggressive marketing. It's definitely non-compliant', and 'it's pushy, aggressive, in my face.'

4.3.2 Case 2: Aer Lingus

Generally, the participants felt Aer Lingus was more compliant than Ryanair. Most felt Aer Lingus's site was not usually harassing or coercive. However, there were real areas of concern. The first appearance of the flex fare was thought acceptable. However, when it appeared for the second time, most thought it was pushing the boundaries of compliance, rather than being non-compliant '(it's) pushy. A bit aggressive, borderline compliant', 'it's designed to be close to the border of coercive. Still it is pushing it.' Some felt the must-opts (insurance and parking) with Aer Lingus were borderline compliant, while others that they were outright non-compliant. The preselected nature of the cost was thought to be pushy at least and at worst aggressive: 'they are hoping people will take it. It's a little aggressive'; 'it's designed to just meet the terms of the legislation.' Others complained about the use of the must-opt design of these optional extras, one declaring 'it's aggressive. I shouldn't have to tell them I don't want it.'

5 Discussion and Conclusions

No participant believed either of the airlines to be compliant with the requirement to communicate all optional extras in a clear, transparent and unambiguous way at the beginning of the process. It would appear that airlines have blatantly ignored this legal obligation as outlined in the EU legislation (European Union 2005, 2008) governing airlines price information in order to maintain the perception that flights are low cost for as long as possible into the booking process. By preventing the user from realising the true cost of the flight until late in the interaction making comparisons across airlines is more difficult. It is therefore less likely that users will reverse out of the process having invested heavily in time and emotional capital.

The findings are conclusive that all optional extras are not presented as opt-in decisions to users. A key finding in this study is that a novel device, ‘must-opts’, has been deployed and that it may circumvent the legislation. As noted earlier, must-opt decisions are both opt-in and opt-out and are also neither opt-in nor opt-out. Strictly speaking, a user does not need to read opt-in or opt-out decisions. This is most decidedly not the case with a must-opt decision. A user may not proceed (e.g. to the next screen) without having made a selection. Failure to do so means you enter an endless loop until you comply. The burden of a must-opt is that you are forced to engage intellectually and mechanically with the Web page. Must-opt decisions apply pressure on users to take seriously the option placed before them. In some cases, this approach is justifiable, such as the requirement that users read terms and conditions of a ‘contract’. The must-opt is clever – it does not preselect an option for a user, so ostensibly, it is not an opt-out optional extra. More importantly for the airline, it does not allow the user to lazily scroll through unwanted services seeking the continue button and an exit from the page. It presents a juncture in the workflow of the transaction where a user is forced to select either service inclusion or service exclusion. At the outset, participants in this study were clear about their definition of opt-in. However, this understanding often changed during the verbal protocol session, sometimes shifting towards an acceptance of must-opt as a characteristic of the opt-in process. This uncertainty surrounding users understanding of choice led to confusion, wariness and frustration. In addition, Ryanair presenting the must-opts in a variety of design forms added to the confusion. The reason for this design is unclear, but poor systems development practices would be the most innocent. Given that both LCCs are aware of certain industry conventions, such as enforcing users to agree that terms and conditions have been read, means the LCCs fully understand what a must-opt construct is designed to do. However, as ancillaries increase their contributions to the airlines’ bottom line (Amadeus and IdeaWorks 2010), we can expect them to continue to find creative ways to circumvent the ethos of the legislation.

Participants were mixed in their opinions as to whether the airlines were using harassment or coercion in order to convince them to purchase optional extras. Most issues arose around the must-opt selections. Participant views ranged from deeming them pushy to harassing and non-compliant. While insurance on both Websites is presented as a must-opt, participants responded more negatively to Ryanair's handling of the option. Re-presenting the option again after users have clearly made a decision questions the merit of their action rather than pointing out something they have overlooked. The offer of parking on Aer Lingus can be seen as similar to the offer of car hire on Ryanair in that both are tangential to the task of booking a flight. However, even though Aer Lingus use the must-opt feature for parking, the reactions of the respondents were not as negative as they were for car hire on Ryanair. This reaction is presumably because, although both use a must-opt selection, Aer Lingus present it in a more muted way. It is simply presented as an option in the flow of the purchase rather than a full-blown, separate window on Ryanair in bold black and yellow.

This study set out to explore user views on whether two Irish low-cost carriers were acting in good faith in implementing consumer protection legislation. The European Commission had recognised new technologies were being used to nudge consumers to behave in a way that airlines wished. While ancillaries have become an essential and growing component of airlines' revenues, it should be expected they are implemented in such a way that consumers do not perceive them as a barrier to securing flights. Although regulations specifically recommend optional charges be accepted on an opt-in basis, the airlines in this study may have found a technological bypass of the regulations – the must-opt construct. This approach and the ambiguity of the definition of opt-in and opt-out decisions allow airlines to exploit the legislation to serve their own ends. The airlines' lack of full compliance with EU legislation suggests forthcoming ECC-Net reports are likely to reiterate the same recommendations (ECC-Net 2010). The game of catch up between regulation and technology continues – back to the drawing board for researchers and legislators.

References

Barry C, Lang M (2001) A survey of multimedia and Web development techniques and methodology usage. IEEE Multimed 8(3):52–60

Benbunan-Fich R (2001) Using protocol analysis to evaluate the usability of a commercial Web site. Inf Manage 39(2):151–163

Cotton D, Gresty K (2006) Reflecting on the think-aloud method for evaluating e-learning. Br J Educ Technol 37(1):45–54

de Neufville R (2008) Low-cost airports for low-cost airlines: flexible design to manage the risks. Transp Plan Technol 31(1):35–68

Doganis R (2006) The airline business. Routledge, London

European Union 1008/2008, Regulation of the European Parliament and of the Council on Common rules for the Operation of Air Services in the Community (recast), 2008

European Union 2005/29/EC, Unfair Commercial Practices Directive, 2005

European Consumer Centre Network (ECC-Net) (2007) Report on air passenger rights: consumer complaints 2006, Brussels, Belgium, European Consumer Centre Network

European Consumer Centre Network (ECC-Net) (2010) Fifth anniversary report 2005–2009, Brussels, Belgium, European Consumer Centre Network
Franke M (2007) Innovation: the winning formula to regain profitability in aviation? J Air Transp Manage 13(1):23–30
Graham B, Vowles TM (2006) Carriers within carriers: a strategic response to low-cost airline competition. Transp Rev 26(1):105–126
Harteveldt H, Stark E (2010) Airlines need to convince passengers to use digital channels to buy ancillary products. Forrester Research, Cambridge, MA
Holzinger A (2005) Usability engineering methods for software developers. Commun ACM 48 (1):71–74
Hunter L (2006) Low-cost airlines: business model and employment relations. Eur Manag J 24 (5):315–321
Monk A, Wright P, Haber J, Davenport L (1993) Improving your human-computer interface: a practical technique. Prentice Hall, New York
Nielsen J (1993) Usability engineering. Morgan Kaufmann, San Francisco
Nielsen J, Clemmensen T, Yssing C (2002) Getting access to what goes on in people's heads? – reflections on the think-aloud technique. NordiCHI, Aarhus
Nucciarelli A, Gastaldi M (2008) Information technology and collaboration tools within the e-supply chain management of the aviation industry. Technol Anal Strateg Manage 20 (2):169–184
Rubin RM, Joy JN (2005) Where are the airlines headed? Implications of airline industry structure and change for consumers. J Consum Aff 39(1):215–228
Sharkey J (2011) The real cost of airline travel remains a mystery, for now. The New York Times, May 9, 2011. URL: http://www.nytimes.com/2011/05/10/business/10road.html. Accessed 10 May 2011
Shneiderman B, Plaisant C (2010) Designing the user interface, 5th edn. Addison Wesley, Upper Saddle River/London
Shumsky R (2006) The southwest effect, airline alliances and revenue management. J Revenue Pricing Manage 5(1):83–89
Sorensen J (2011) Planes, cars, and ancillary revenues. IdeaWorksCompany.com, 11 Jan 2011
Teichert T, Shehu E, von Wartburb I (2008) Customer segmentation revisited: the case of the airline industry. Transp Res A Policy Prac 42(1):227–242
Torres A, Barry C, Hogan M (2009) Opaque Web practices among low-cost carriers. J Air Transp Manage 15:299–307
Wiktionary. http://en.wiktionary.org/wiki/opt-in

Chapter 54
Building Healthcare Service Navigation System for a Local Health Integration Network: A Requirements Elicitation Model

Gokul Bhandari

1 Introduction

Local health integration networks (LHINs) were created in March 2006 to transform the delivery of healthcare services in Ontario by simplifying the process of healthcare service access when patients move from one service provider to another. The LHINs are mandated to plan, integrate, and fund local health services and are guided by the belief that "a community's health needs and priorities are best understood by people familiar with the needs of our communities and the people who live there, not from those in offices hundreds of miles away[1]". There are 14 LHINs in Ontario and this chapter discusses information systems development initiative in one of the LHINs called Erie St. Clair LHIN (ESC LHIN hereafter) serving the population of over 649,000 people residing in the regions of Chatham-Kent, Sarnia/Lambton, and Windsor/Essex with an annual budget of over $900 million. There are 88 agencies consisting of hospitals, long-term centres, assisted living services, community support services, community care access centres, mental health agencies, addiction agencies, and community health centres, which are funded by the ESC LHIN. The ESC LHIN needs different types of information from these member agencies in a periodic manner in order to implement its healthcare service integration strategy in the region.

An earlier work (Bhandari and Snowdon 2011) suggested service-oriented architecture (SOA) (Fig. 54.1) for managing the seamless flow of data and information between this meta-organisation (i.e. ESC LHIN) and its member agencies, system architects and designers, and clients. Table 54.1 outlines the agencies and their services. In this chapter, we adopt the SOA framework to develop a requirements elicitation model for building a navigation tool for accessing

[1] http://www.lhins.on.ca/aboutlhin.aspx?ekmensel=e2f22c9a_72_184_btnlink

G. Bhandari
Odette School of Business, University of Windsor, Ontario, Canada
e-mail: gokul@uwindsor.ca

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_54,

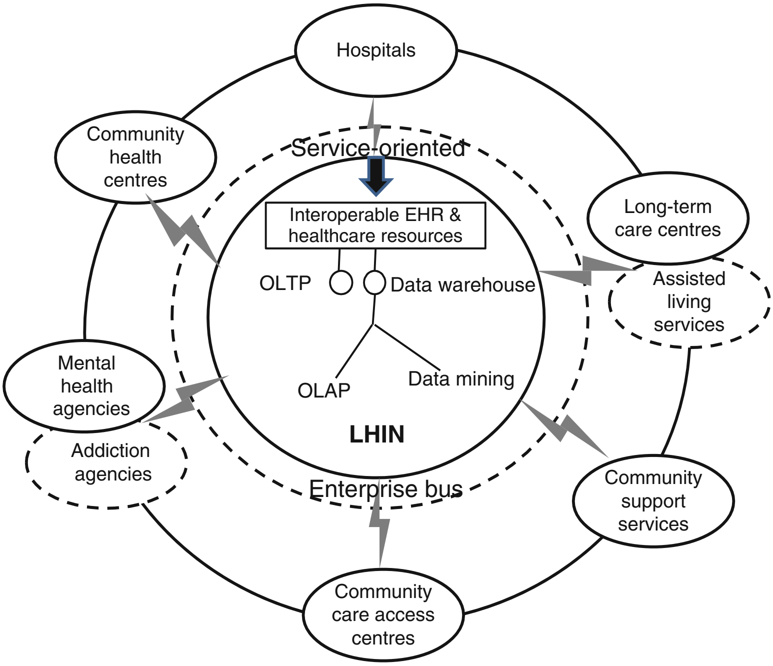


Fig. 54.1 Service-oriented architecture for the LHIN (Adapted from Bhandari and Snowdon 2011)

Table 54.1 LHIN member agencies and their brief service description

Member agencies	Services
Hospitals	Primary care, emergency, outpatient
Long-term care centres	24-h nursing care and supervision
Assisted living services	Assistance with activities of daily living
Community support services	Independence while living at home
Community care access centres	Service co-ordinator and planner
Mental health agencies	Dealing with mental illness
Addiction agencies	Dealing with drug, alcohol, and other abuse
Community health centres	Primary health and health promotion

healthcare services available in the region. The remainder of this chapter is organised as follows. Section 2 introduces the current information systems development initiative (ISD). Section 3 summarises various approaches to formulating the process of requirements elicitation based on our literature review. Section 4 discusses our ontology-based model for developing an automated and interactive tool capable of supporting automatic service discovery, automatic service composition, and dynamic composition. Section 5 concludes this chapter.

2 Information Systems Development in ESC LHIN

In order to understand the need for a navigation system for accessing healthcare services available in the region under the mandate of the ESC LHIN, consider the following scenario:

> An elderly woman at home becomes ill with a fever. She is unable to eat and only drinks the occasional sip of water. When she gets up in the evening she is very light headed, dizzy and then falls while going to the bathroom. She crawls to the phone and calls her daughter, who dials 911 to get her to the hospital. She arrives in the emergency room and waits several hours to be seen. She has blood tests, a CT scan, a urinalysis and a chest x-ray, and she is found to dehydrated, has a low blood pressure due to poor fluid intake and a mild flu. She did not have any serious injuries as a result of the fall. Social work is consulted for home support for this woman. (Bhandari and Snowdon 2011)

This is a case involving a senior citizen living independently at home and becoming ill who needed primary care services and home support to prevent dehydration, low blood pressure, and subsequent fall. This case illustrates how the inability to access appropriate services (i.e. primary care and supportive services) results in patients ending up at the wrong location (emergency room) for the wrong service. This is a common scenario in Ontario and is one of the primary reasons for overcrowding in emergency rooms. In order to facilitate the access of healthcare services available in the region, the ESC LHIN has embarked on a project called the *Integrated Health Service Plan* for the period of 2010–2013 with the following five strategic objectives: improved outcomes in alternate level of care, emergency department care, diabetes management, mental health addiction, and rehabilitation care and interventions. This approach is reflective of the trend that organisations are increasingly defining their solutions as services and their interconnections (Sheth et al. 2006). Such an approach is referred to as a service architecture which is considered successful when it closes the information gap among key actors (Jones 2005).

Earlier efforts in making healthcare services accessible include *ConnexOntario* (http://www.connexontario.ca) and a care coordination programme called *Facilitated Access and Coordinated Teamwork* (FACT) (Bertoni 2009). However, both of them are limited in the sense that *ConnexOntario* provides basic information about alcohol and drug, gambling, and mental health services only and the FACT is a noncomputerised, group participation-based review system only. It should be evident from Fig. 54.1 that for a meta-organisation like ESC LHIN, the information system development is a critical undertaking because it involves understanding user requirements for all of its member agencies. In the following section, we provide an overview of prior approaches to requirements elicitation in ISD projects.

3 An Overview of Requirements Elicitation in ISD

Prior studies find that many ISD projects fail because of the poor understanding of user requirements during the requirements elicitation (RE) phase (Mathiassen et al. 2007) resulting in significant costs to organisations (Brown and Rogich 2001). While the importance of RE in ISD is unequivocal among IS scholars, the lack of uniformity in conducting RE is critical. For example, Sommerville (2007) views the RE process as a staged sequence of activities and/or task objectives while others view it as "chaotic and non-linear" (Davidson 2002), non-deterministic (Rolland 1993), and dialectical (Thanasankit 2002). There is also a disagreement between researchers as to whether RE processes should be characterised as normative or nonnormative ones. As normative processes, RE have been described as a multi-stage process characterised by input-task objective-output (Browne and Ramesh 2002), iterative process typified by evolving knowledge about the system requirements (Hickey and Davis 2004), and knowledge acquisition process (Byrd et al. 1992). While Hickey and Davis (2004) have proposed a unified model of RE by emphasising the role of situational knowledge, the abundance of proposed RE techniques for ISD has created a confusion among systems developers, which Mathiassen et al. (2007) rightly term as a "methodology jungle".

Given that the ESL LHIN is a meta-organisation mandated to integrate a variety of healthcare services offered by diverse agencies (see Fig. 54.1), the challenge of RE for the navigation project increases exponentially because just one methodology or technique cannot possibly meet all requirements of different organisations (Davis and Hickey 2002; Maiden and Rugg 1996). In order to manage this complexity of RE in the current ISD initiative, we suggest an ontology-based RE model for developing an automated and interactive tool capable of supporting automatic service discovery, automatic service composition, and dynamic composition.

4 Ontology-Based RE Model

Ontology refers to a formal, explicit specification of a shared conceptualisation to represent common knowledge within a domain (Gruber 1993). While the use of ontology in RE can be traced back to the early 1980s (Dobson and Sawyer 2006), it still remains a popular methodology until these days. Kaiya and Saeki (2005) proposed an ontology-based RE model with the objective of identifying incompleteness and inconsistency of requirements artefacts, measuring requirement engineering quality, and forecasting possible future changes.

In the current ISD, we use Semantic Web Service (OWL-S) for automating implementation activities for the SOA framework depicted in Fig. 54.1. OWL-S, as an ontology of services, makes automatic service discovery and composition possible (Martin et al. 2004). Before software with SOA methods can be built, all relevant services must be discovered as the output from the RE process. In OWL-S, an instance of *ServiceProfile* is used to discover a service with the instantiation of *Input*, *Output*, *Precondition*, and *Result*, which respectively represent the following: the information required by the service to work, the message returned by the service, the condition required by the service for its proper execution, and the impact of the service execution. Each elicited requirement is associated with an aspect of service discovery along with the functionalities represented with *Input*, *Output*, *Precondition*, and *Result*.

Our proposed interactive RE process can be divided into two broad stages: (1) machine-directed RE for identifying implicit knowledge (indirect relationships) with reasoning and then performing RE and (2) user-directed RE for customising requirements. The interactive dialogue system is being developed in Java for Windows OS using Eclipse IDE for coding and debugging. Service ontology is being developed with Protégé ontology editor and Pellet for ontology reasoning.

The proposed interactive dialogue system consists of four components: dialogue interface, I/O controller, dialogue manager, and ontology knowledge base. The dialogue interface is a text-based system displaying generated statements such as "Would you like to select the requirement…" to which the user is expected to answer "Yes" or "No". The user's response is then passed on to the I/O controller which will attempt to match the input with a set of prespecified information. When no match is found, the user is asked to modify her response. When a match is found, the input is formatted and given to the dialogue manager, thereby knowing the user's requirement decision currently being elicited. The dialogue manager then consults the ontology knowledge base with the decision and customises the requirements if necessary. Once this process is complete, the dialogue manager generates output which is passed onto the I/O controller. The I/O controller then converts the output in natural language which is displayed in the dialogue interface initiating the next interaction cycle.

In order to understand the proposed ontology-based RE model, let us consider a service ontology for locating a healthcare service (as illustrated in the scenario in Sect. 2). The first task would involve function decomposition. As shown in Fig. 54.2 (adapted from Zhang 2011), *Locate a service* can be decomposed into two functions: *Get reference to a service* and *Get detailed info of a service*. Getting detailed information about a healthcare service may involve either getting information about the service provider (such as hospital or community care access centre) or getting description about the service itself (such as primary care and supportive services for elderly). In order to get the reference to a service, search method is followed resulting in a list of relevant services which the user can select by pointing to it. Sorting of services (in alphabetical order) could also be done to facilitate locating the relevant service from the list. Searching of services can be implemented in two ways: predefined keyword search and advanced search allowing different combinations. For the search involving keyword matching, there are two possibilities: broad match and exact match. Since broad match and exact match exhibit two different levels of search quality constraints, they are considered mutually exclusive (indicated as contradiction in Fig. 54.2).

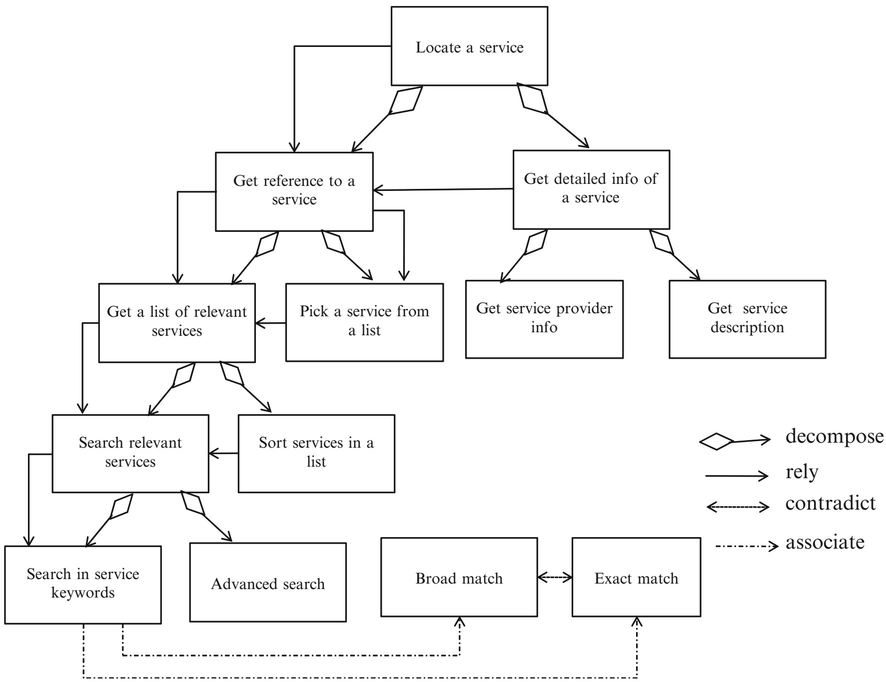


Fig. 54.2 Instantiation of requirement model for locating a service

Requirements elicitation can be performed once the ontology model is instantiated and completed through the interactive dialogue system (as discussed earlier). When the dialogue system encounters a user response that it cannot handle, it asks the user to correct it. When all the functions depicted in Fig. 54.2 are handled by the dialogue manager, the requirements elicitation is considered complete resulting in the generation of output in the OWL-S document format. When services are described in the OWL-S format, they can be discovered by semantic capability matching.

The proposed system, which could be hosted by the ESC LHIN, would be used by both organisations (member agencies) and the general public. The example shown in Fig. 54.2 is intended to illustrate how the public could access and navigate healthcare services available in the region. Descriptions (profiles) of the services in the organisational level can be generated in a similar manner.

5 Conclusion

In this chapter, we discussed ontology-based requirements elicitation model for developing a service navigation system for a meta-organisation such as the ESC LHIN. A major advantage of the proposed model is that it demonstrates the feasibility of conducting automated RE which is critical for a complex ISD initiative such as healthcare service navigation system. Although this chapter discusses a study done in the Canadian health sector, we believe the methodology followed by us is general enough to be of value in other contexts as well. One major limitation of the proposed automated RE model is that it is not applicable to a dynamic environment. Furthermore, the support for customising requirements is inherently limited by the knowledge the machine owns. However, in the future, the proposed model can be extended by improving its expressiveness and domain characteristics. Finally, we conclude by underscoring the significance of eliciting requirements from the public and regional healthcare service providers for developing information systems following software product line engineering approach.

Acknowledgments Financial support for this study has been provided by the Odette School of Business, University of Windsor, under Research, Teaching, and Innovation Fund.

References

Bhandari G, Snowdon A (2011) Design of a patient-centric, service-oriented health care navigation system for a local health integration network. Behav Inf Technol:1–11, First published on: 04 May 2011 (iFirst)

Browne GJ, Ramesh V (2002) Improving information requirements determination: a cognitive perspective. Inf Manage 39:625–645

Byrd TA, Cossick KL, Zmud R (1992) A synthesis of research on requirements analysis and knowledge acquisition techniques. MIS Q 16(1):117–128

Davis A, Hickey A (2002) Requirements researchers: do we practice what we teach. Requir Eng J 7:107–111

Dobson G, Sawyer P (2006) Revisiting ontology-based requirements engineering in the age of the semantic web. In: Proceedings of the international seminar on dependable requirements engineering of computerized systems at NPPs. Available at http://citeseerx.ist.psu.edu

Gruber TR (1993) A translation approach to portable ontology specifications. Knowl Acquis 5 (2):199–220

Hickey AM, Davis AM (2004) A unified model of requirements elicitation. J Manage Inf Syst 20 (4):65–84

Jones S (2005) Toward an acceptable definition of service. IEEE Softw (May/June): 87–93

Kaiya H, Saeki M (2005) Ontology based requirements analysis: lightweight semantic processing approach. In: Proceedings of the 5th international conference on quality software, 223–230

Maiden N, Rugg G (1996) ACRE: selecting methods for requirements acquisition. Softw Eng J 11 (5):183–192

Martin D et al (2004) OWL-S: semantic markup for web services. Available at http://www.w3.org/submission/OWL-S

Mathiassen L, Tuunanen T, Saarinen T, Rossi M (2007) A contingency model for requirements development. J Assoc Inf Syst 8(11):569–597

Sheth A, Verma K, Gomadam K (2006) Semantics to energize the full services spectrum. Commun ACM 49(7):55–61

Thanasankit T (2002) Requirements engineering-exploring the influence of power and Thai values. Eur J Inf Syst 11(2):128

Zhang X (2011) An interactive approach of ontology-based requirement elicitation for software customization. Unpublished Master's Thesis, University of Windsor, ON

Chapter 55
Cost Estimation in Agile Software Development Projects

Michael Lang, Kieran Conboy, and Siobhán Keaveney

1 Introduction

A decade has now passed since the chief proponents of what were then called "lightweight" software development methods, – including eXtreme Programming (XP), Scrum, Crystal, dynamic systems development method (DSDM), feature-driven development (FDD), adaptive software development (ASD), and pragmatic programming, – famously convened in Utah to form the Agile Alliance. The outcome of that meeting was the proclamation of a "Manifesto for Agile Software Development" which called for a profound shift in the underlying philosophy of traditional ISD approaches (Highsmith 2001). The Agile Manifesto embodies 12 guiding principles and a declaration of values which places individuals and interactions over processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation, and responding to change over following a plan. Fundamentally, the agile movement is based on a new paradigm which argues for a departure from so-called plan-driven or heavyweight ISD methods on the basis that they are not appropriate in the modern era of rapid change, a viewpoint that increasingly was gaining support in the academic literature through the 1980s and 1990s (McCracken and Jackson 1982; Baskerville et al. 1992). At the turn of the millennium, practitioners and academics alike were calling into question the underlying assumptions upon which traditional ISD

M. Lang (✉) • S. Keaveney
Business Information Systems, J. E. Cairnes School of Business & Economics, National University of Ireland, Galway, Ireland
e-mail: Michael.Lang@nuigalway.ie

K. Conboy
Business Information Systems, J. E. Cairnes School of Business & Economics, National University of Ireland, Galway, Ireland

School of Information Systems, Technology and Management, Australian School of Business, The University of New South Wales, Sydney, Australia

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5_55,

methods were based (Highsmith 2001; Russo and Stolterman 2000), and it was broadly acknowledged that there was a need to move from the past imperfect to a better future way of building software (Fitzgerald 2000).

Over the course of the past decade, the notion of "agile" information systems development has found tremendous favour, as evidenced by the increasing number of practitioner and academic conferences, the high rate of uptake of agile methods within industry, and a rapidly growing body of research activity. However, few studies of agile methods in actual use are based on strong theoretical or conceptual foundations (Abrahamsson et al. 2009; Conboy 2009). In the absence of systematic research, there are few lessons learned across studies, and thus the existing body of knowledge is fragmented and inconclusive. This is particularly problematic for agile project managers who have trained and worked with traditional, plan-driven development approaches (Dybå and Dingsøyr 2008; Tan and Teo 2007).

Regardless of the methodology adopted, the ISD process requires effective management and planning. A large part of this planning is the creation of estimates so that resources can be appropriately allocated during projects. Numerous cost-estimation techniques and models (e.g. COCOMO, SLIM, ESTIMACS, COBRA, Checkpoint) have been proposed over the years, an extensive taxonomy of which can be found in Boehm et al. (2000). Across their systematic literature review of 304 software cost-estimation papers, Jørgensen and Shepperd (2007) identified regression, function point, expert judgement, theory-based, and analogy as the main cost-estimation approaches. However, notwithstanding the vast body of cost-estimation literature, the chronic problem of cost and schedule overruns on ISD projects indicates that accurate estimation remains elusive.

One of the main principles of agile methods is to "welcome changing requirements", but changing requirements is a major cause of software cost-estimation problems (Jones 2003; Conboy 2010). Alford and Lawson (1979) pointedly remark that "in nearly every software project that fails to meet performance and cost goals, requirements inadequacies play a major and expensive role in project failure". As yet, the issue of cost estimation in agile software development projects has received very little attention in the academic literature, the only previous empirical studies that we discovered in our literature search being those of Cao (2008), who conducted an in-depth longitudinal study on an agile project in which estimates were compared versus actuals, and Ramesh et al. (2007), who investigated agile requirements engineering practices within 16 US software development organisations. Interestingly, whereas Ramesh et al. (2007) observed that "the agile approach towards RE makes the estimation of costs and schedules more difficult than with traditional methods", Cao (2008) found that "estimation in agile development is more accurate than that in traditional development even though agile developers still underestimate the effort". The research discussed in this chapter was part of a broader study, but in view of the aforementioned gap in the literature, the aspect that we have chosen to concentrate on here is how do agile ISD approaches cope with the problems that have traditionally plagued software cost estimation.

2 Literature Review

Estimating the cost of an ISD project is one of the most crucial tasks for project managers, but unfortunately it is a persistent weak link. ISD projects have a long history of being delivered over time, over budget, and failing to satisfy requirements. As early as 1958, concerns about information systems project failure were expressed in the inaugural edition of *The Computer Journal* (Caminer 1958). By the time of the 1968 NATO Conference on Software Engineering, the high incidence of failure had reached such proportions that the now infamous phrase “software crisis” was first uttered (Naur and Randell 1969). Brooks (1987) uses the metaphor of “a monster of missed schedules, blown budgets, and flawed products” to convey the essence of this problem. In a US study conducted by The Standish Group (1995), it was found that only 16 % of software projects were completed on time and on budget, with 53 % of projects costing approximately double their original estimates. Somewhat more positive findings were reported by Lang and Fitzgerald (2007), whose survey of 164 Web development companies in Ireland revealed that 67 % of projects were delivered within the agreed budget and 33 % were delivered on time. However, even though those figures are more favourable, they still indicate that two-thirds of projects for whatever reason are delivered late, and a third are over budget, which might even be more if the real cost of fixed-price contracts was considered.

Lederer and Prasad (1995) conducted a survey of 112 ISD project estimators and implementers, and based on an exploratory factor analysis, they classified the principal causes of inaccurate cost estimates in traditional ISD projects into four categories, namely, *methodology*, *politics*, *user communication,* and *management control*. We use these classifications here to present the main strands of literature and the corresponding findings.

2.1 Methodology Issues

Cost-estimation problems attributable to methodology issues include the techniques and guidelines employed to produce the estimate, the means by which estimates relating to past projects are examined and reviewed, the setting of standard estimation durations, insufficient analysis when developing estimates, and lack of co-ordination of systems development activities (Lederer and Prasad 1995).

Over-reliance on intuition and personal memory is a concern for project members trying to increase estimation accuracy. Estimation inaccuracy can also be caused as a result of a lack of policies on how to learn from past experiences and properly deal with failures and mistakes (Ewusi-Mensah and Przasnyski 1995). In agile software development, estimates are normally produced on an iterative basis for each “sprint” of activity, typically 2–4 weeks turnaround. One very commonly used agile requirements specification technique is a “user story”, a feature of both the eXtreme Programming and Scrum methodologies. This technique, which has been referred to as "just-in-time analysis", asks users to very succinctly communicate requirements in the form of short, simple task descriptions. Each user story therefore represents a distinct piece of functionality that can be plugged into a system. An overall expected time for each of these stories is estimated by the developers, and the customers then prioritise the stories based on these initial estimates and on the business value of each one (Lovaasen 2001).

The frequency with which estimation is performed, typically at the beginning of every iteration, leads to progressively more accurate estimation by the developers as they become more and more skilled at estimating the tasks (Abrahamsson 2003). According to Highsmith (2003), the nature of agile methods often results in fixed budgets and a fixed schedule, and it is the scope of the project that remains flexible throughout. On the other hand, Ceschi et al. (2005) report that companies using agile methods usually lean towards "flexible contracts instead of fixed ones that predefine functionalities, price, and time".

Agile methods "welcome changing requirements, even late in development"; however, in terms of estimation, the requirements are finalised to a certain extent at the start of each iteration, and so developers can devise their estimates being reasonably safe in the knowledge that the scope for the iteration has been broadly agreed (Taber and Fowler 2000). The impact of changes in scope and requirements within ISD projects can vary greatly depending on the stage at which the change is introduced. The cost of change rises phenomenally throughout traditional development (Boehm 1981), while in agile projects the impact of change levels off (Neill 2003). Agile methods aim to reduce the cost of changes throughout the development of a system, but not necessarily to reduce the occurrence of changes (Highsmith and Cockburn 2001).

2.2 Political Factors

According to Jørgensen and Moløkken (2003), estimation is typically fraught with "tug of wars" and "political games"; therefore, high accuracy may not be the only goal or perhaps not even the principal goal of the actors involved. Chapman and Ward (2002) refer to a "conspiracy of optimism" whereby political pressures from within the organisation can lead to unrealistic estimates or reluctance to report the actual outcome. Moløkken and Jørgensen (2003) suggest that software managers may over-report causes of inaccuracy that lie outside their responsibility, such as customer-related causes. Project managers therefore have to be aware of the implications that political factors can have on ISD estimation (Winklhofer 2002).

Lederer and Prasad (1995) identified pressures from managers, users, or others to increase or reduce the estimate, or removal of padding from the estimate by management, as political factors that can negatively impact the accuracy of software cost estimation. It is quite common for software developers to experience stakeholder pressure to stay within the original base estimate, but if those estimates were initially pitched or subsequently manipulated in order to satisfy managers or customers, they will usually lead to overruns and shortfalls (Lang 2009). Within "self-organising" agile teams, the delegation of responsibility to developers to estimate their own tasks can cause inaccuracies if a developer feels pressurised into underestimating his workload in order to gratify managers or customers. This agile practice can also lead to reluctance by developers to expose themselves to the risk of developing a reputation as having poor estimation/time management skills or perhaps even seen amongst peers as having limited technical capabilities (Elssamadisy and Schalliol 2002).

Lederer and Prasad (1995) also identified reduction of project scope or quality to stay within the cost estimate resulting in extra work later and "red tape" as two other political factors. The agile principle of simplicity, which is defined in the Manifesto as "the art of maximizing the amount of work not done", would seem to directly relate to these issues. Additionally, the Manifesto's emphasis on "customer collaboration over contract negotiation", which resonates with Jones' (1988) call that "both parties (designers and clients) have to give up the use of the requirements as a semi-legal basis of control and measurement and agree to work together", means that at least in principle agile ISD cost estimation aspires to be less prone to the ills of adversarial politics. Indeed, one of the criticisms of the traditional heavyweight methodologies was that they could descend into political "rituals which enable actors to remain overtly rational while negotiating to achieve private interests" (Robey and Markus 1984). In their study of Web development practices, Lang and Fitzgerald (2007) found that methods could serve a variety of covert political motives, such as being seen to have followed a process in the event of a "blame game" arising between stakeholders about overruns. While one would hope that the spirit of the Agile Manifesto would precipitate a change of culture in this regard, it is perhaps too optimistic to expect that political factors and their potentially disruptive impacts can be entirely eradicated.

2.3 User Communication

Brooks (1987) famously declared that "the hardest single part of building a software system is deciding precisely what to build . . . no other part of the conceptual work is as difficult as establishing the detailed technical requirements . . . no other part of the work so cripples the resulting system if done wrong". Lederer and Prasad (1995) identify users' lack of understanding of their own requirements, frequent requests for changes by users, users' lack of IT knowledge, and poor or imprecise problem definition as major contributory factors to inaccurate cost estimates.

Poor communications with users is one of the most prominent reasons why project estimates tend to be inaccurate (Jørgensen 2003). It is quite normal for users not to fully understand what they want and to be unable to clearly articulate their needs (Brooks 1987; Walz et al. 1993). Boehm (2000) refers to the fickle and rapidly changing nature of user requirements as the "I'll know it when I see it"

(IKIWISI) phenomenon. If customers have a limited awareness of the potential and limiting factors of information technology, as is typical, they cannot be expected to be in a position to clearly state their requirements at the outset of an ISD project (Orr 2004; Stamelos and Angelis 2001). This leads to difficulty in producing a complete set of requirements and thus estimation inaccuracy is inevitable.

Keil and Carmel (1995) therefore recommend that a substantial portion of the time assigned to systems development activities should be given over to learning and knowledge exchange between customers and developers. Moreover, they call for direct links between customers and developers because where communication passes indirectly through intermediaries, it is likely to be less effective because they can filter and distort messages (Keil and Carmel 1995). Similarly, Grudin (1991) makes the point that "go-betweens or mediators often discourage direct developer-user contact . . . and are often ineffective conduits". Bringing this point forwards, the Agile Manifesto states amongst its principles that "business people and developers must work together daily throughout the project" and that "the most efficient and effective method of conveying information to and within a development team is face-to-face conversation". The eXtreme Programming methodology recommends that an on-site customer should be attached to the development team, though in practice this often does not happen or perhaps the role is filled by a customer proxy. This close working relationship between the project team and the customer in agile software development approaches mitigates the traditional problems arising from poor user communication, but on the other hand, the effectiveness and success of agile methods is very dependent on customer co-operation and availability, which if not forthcoming threatens to unravel the whole process (Paulk 2002). For example, if a customer was not available to clarify and elaborate on confusing user stories, development activities might have to proceed misguided or stall altogether.

2.4 Management Control

Problems caused by management control include management reviews and comparison between estimates and actuals. When management fails to participate in the preparation of the estimate and does not monitor the accuracy of the estimate, this can contribute to the estimate being inaccurate. Inaccuracy also occurs when management does not refer to the estimate when conducting performance reviews of estimators and other project personnel (Lederer and Prasad 1995).

In order for an estimate to be accepted and adhered to, it must consider and include all members of the development team and in particular the project manager (Agarwal et al. 2001). It also must be communicated clearly to the project team before the development begins. Research has shown that if the estimator is somebody who will be involved in the development, the estimation accuracy is likely to be higher than if an estimate is produced by a senior executive or a staff member from a different department (Jurison 1999).

Within the agile paradigm, each developer takes responsibility and ownership for the stories that he estimates, and so management involvement is less of an issue in agile ISD as it is in traditional development (Schalliol 2001). Management involvement in agile projects tends to be less "hands-on" than on traditional projects and their involvement is at a higher level, enabling them to oversee the estimation process from one iteration to the next (Abrahamsson 2003). Evaluation of team members based on their ability to meet the estimates is less appropriate for agile projects because it is the developers themselves who estimate their own tasks (Schalliol 2001).

3 Research Approach

The notion of cost estimation in agile ISD projects combines an important and much researched project management issue with the relatively new topic of agile development, where comparatively little empirical research exists. In order to gain a deeper understanding of the core issues, we therefore chose to follow an investigative approach based on semi-structured qualitative interviews. For feasibility reasons, the four companies that participated in our study were drawn from a convenience sample, but they were purposefully selected so as to obtain breadth and diversity, e.g. indigenous small-to-medium businesses versus larger multinational organisations and well-established companies versus recent start-ups. A comparative profile of the companies is shown in Table 55.1.

At each company, interviews were held with project managers/team leaders. A list of interview questions and topics for discussion was emailed to each interviewee in advance. All interviews took place on-site within the companies' premises and lasted between 1 and 2 h. Conversations were audio-recorded by agreement of interviewees, and observational notes were also made during and immediately after the interviews. The interviews followed the general course of the

Table 55.1 Case study company profiles

Company name	Year founded	No. of employees	No. of concurrent projects	Typical project length	Team size	Development methodology	Estimation techniques
Travtech	1999	70	15–20	2–3 years	2–5	Tailored version of XP	Expert judgement; regression
BrightSoft	1995	12–15	4–5	4–6 months	12–15	MSF for agile software development (MSF4ASD)	Analogy
MobilApp	2002	13	1–5	1–2 months	1–10	Variant of XP	Expert judgement
HPG	1971	500	2–3	4–8 months	6–7	Tailored version of XP	Expert judgement

pre-planned questionnaire schedule but, where appropriate, elaboration was sought on points that were of high relevance to the research. Upon conclusion of each interview, provision was made for follow-up meetings, phone calls, or emails on points that required clarification or further investigation. Immediately after the interviews, the recordings were fully transcribed. All interviewees were subsequently contacted via email requesting clarification or elaboration on points made during the interview. The analytical procedures employed followed the general principles espoused by Miles and Huberman (1994) and concentrated on the transcribed interview conversations, notes made during the interviews, email correspondence from before and after the interviews, and any available secondary information about each company's activities.

4 Findings

The experience of the companies in our study was that reasonably accurate estimates for agile projects are easier to produce because of the frequency with which estimates are required. Typical agile iterations spanned 2 weeks, with estimates being produced at the beginning of each iteration. This not only helped to keep a high degree of accuracy but also honed the estimation skills of the team members and developers involved. Notably, estimation inaccuracy was not a substantial problem for any of the companies and where it was; they typically saw it as an opportunity to learn and inform their future estimation activities.

Fixed-price projects where a budget is agreed at the beginning seemed to be the most common project type. In some cases, the schedule was movable and in others it was the functionality that could be revised. Typically when the cost is determined, a number of developers can be assigned and the delivery date calculated from this. On the other hand, if the schedule is set by the customer, then the cost can be calculated from the number of people available to work on the project. Either way this enables the project to be run in a manner that delivers increasingly more features as time progresses until the scheduled delivery date has been reached.

4.1 Methodology Issues

The main estimation techniques used across the four projects were analogy and expert knowledge with varying degrees of formality and structure between the companies. In some cases project data was stored, and in others it was simply assigned to the developers own memories. Estimation models, despite their popularity in the literature, were not used by the companies and for the most part were not even recognised. "User stories" were very commonly used as the unit of work for which developers were asked to return estimates, and all four companies followed the "planning game" practice.

4.1.1 Procedural Flaws and Shortcomings

As regards causes of inaccurate estimates related to flaws within the execution of estimation processes, Travtech's experience is that the requirement for appropriate expertise, and in particular domain and technical expertise, as part of their estimation process is something that has given rise to estimation errors. For example, it has occasionally happened that somebody might be asked to produce estimates in an area that they are not particularly familiar with or with a technology or development language with which they have limited experience. Travtech's approach in situations where they find themselves in unfamiliar territory as regards application domain or development platform is to build a risk factor into the estimate to compensate for the amount of time and effort that will be required to come up to speed with the intricacies of new technologies. Interestingly, Travtech also commented that eXtreme Programming, if indeed taken to extreme limits and "applied rigorously with little up front documentation" could lead to costly situations further on where "there is a lot of refactoring which has to be done and this can create real inefficiencies when having to rewrite software".

The estimation method that HPG use on their projects is quite an informal process, and they feel that this lack of formality may contribute somewhat to the discrepancies in their estimates (of the order of 10 %), although another possible reason for the variance is not necessarily that the original estimates were wrong, but rather that subsequent change requests were not properly tracked and the initial estimate would therefore appear to be out of line. The fact that they are using agile development practices has also led them to focus less on formality overall. With regard to the development process changes that HPG have experienced, there have been significant repercussions from the adoption of agile processes. The team members have had to adapt to the new practices and learn new skills such as pair programming and test-driven development. This has impacted cost estimates, particularly in the early stages of the project, because they now have to take other factors into account such as refactoring and acceptance testing.

4.1.2 Use of Guidelines to Counteract Underestimation

Consistent with the findings of Cao (2008), we discovered that although cost estimates are reasonably accurate, developers continue to have a tendency to underestimate. Both HPG and BrightSoft spoke of how they compensate for this by means of the concept of the "perfect engineering day". Interestingly, although agile methods have a strong focus on productivity issues such as "maximizing the amount of work not done" and "maintaining a constant pace indefinitely", HPG and BrightSoft both have a policy of treating a developer's estimate of "8 hours" of work as the equivalent of two working days. As the HPG project manager explained, "team members tend to think that they have spent a full day at a task but in reality will have only spent 3 or 4 hours because of interruptions". BrightSoft combine this “perfect engineering day” rule of thumb with their “relative size table”, which is a reference guide to the effort taken to complete similar work assignments in the past. This enables them to produce very accurate estimates. For example, if a developer estimates that a task will take 1 day (i.e. 4 h), but the relative size table suggests that it will take 2 days (i.e. 8 h), the average of the two is usually taken as the estimate.

4.2 Political Factors

Neither HPG nor MobilApp raised any cost-estimation issues related to political factors. BrightSoft explained that they are a small closely knit crew with a strong collegiate culture and as a result have never experienced any internal political divisions. For example, there is never any suggestion of apportioning blame on individuals for estimates going astray, and though estimates are audited during post-implementation reviews, that is emphatically not for the purposes of evaluating the personnel involved in either the estimation process itself or the development effort.

As regards “playing it safe” with customers, if BrightSoft find themselves placed under pressure to come up with a fixed deadline, they generally react by building 2 weeks of slack into the estimate as a risk buffer. It may turn out that only a certain number of features are required and the project can be delivered before the risk buffer has been expended, so they recognise that “it is important politically to get the balance right because over-estimating can cause problems as well”.

4.2.1 Pressure to Reduce Estimates

Of the four companies interviewed, only Travtech appear to be experiencing estimation problems arising out of the types of political factors identified by Lederer and Prasad (1995). Pressures from Travtech managers and customers can cause unrealistic estimates to be produced in order to keep a customer on track or prevent a manager from pulling the project altogether. There could also be a combination of both the management and the customers adding to the pressure for lower estimates.

It can be very difficult if the team leader calculates an estimate that represents the capabilities of their team and they know that the customer will not be willing to accept the length of time or the cost required for completing the project. This problem arises more frequently with customers that are not IT savvy because they are not as appreciative of the effort required to implement certain requirements. This can sometimes lead to underestimation by team members as they may be conscious in the back of their minds that the customer wants better value.

As regards pressures from managers giving rise to poor estimates, this can occur when there are certain tasks included as part of the estimate and a team member ends up cutting corners to produce a lower estimate than what can be realistically expected. For example, if the testing phase is left entirely until the end of the project, it is often the part that will be omitted in the formal estimates in an effort to bring the estimate down. This will render the estimates inaccurate because effort will always need to be expended on testing regardless.

4.2.2 "Red Tape" Issues

Travtech have also occasionally ran into "red tape" political problems, such as where trade unions within the customer organisation can dictate work practices and distribution of tasks or where the customer organisation has its own zealously protective internal IT department acting as a gatekeeper. In such scenarios, the political "tug of wars" spoken of by Jørgensen and Moløkken (2003) can arise as stakeholders attempt to wrest control of certain aspects of the project. This presents a serious risk to the integrity of time and cost estimates.

4.3 User Communication

Instances of all four of the main types of user communication problems identified by Lederer and Prasad (1995) were evident in our study. As mentioned in the previous section, users' lack of IT knowledge was not just a communication problem, but it also gave rise in some cases to political pressures being heaped on developers by technologically naïve customers. The other three issues are described in the following subsections.

4.3.1 Poor or Imprecise Problem Definition

The Travtech project manager felt that poorly documented requirements and insufficient management of the relationship with and involvement of the customer can leave too much room for misinterpretation, resulting directly in inaccurate estimates. Expectations may be based on an impression that the customer got from a meeting, and their methodology might fail to ensure that clarification was sought. Similarly at HPG, estimation problems have arisen as a result of poor problem definition from the customers or where the customer comes to the team with a change request and the scope of a particular story needs to be revised.

Agile methods aspire to address poor problem definition by placing developers and an "on-site customer" into direct daily communication. However, of the four companies that we visited, only one (HPG) had an on-site customer. As explained by the project manager, "our customer wasn't really on-site up until a few weeks ago when there were a few issues that were coming up in the retrospective regarding communication problems between the development team and the customer, so we decided to set up a machine for the customer in our lab area and now she's on-site probably 80 % of her time, which has helped". Both Travtech and BrightSoft mainly supply to the export market so it is not feasible for them to maintain a customer on-site. In order to stay close to their customer, which they feel is important from the point of view of gaining good feedback and getting to know how to handle key individuals, BrightSoft have set up an international office in California close to their main customer base.

4.3.2 Users' Lack of Understanding of Their Own Requirements

Travtech have experienced inaccurate estimates in situations where the customer knows their own business so well but cannot articulate it in a form that the development team can understand. Customers can often find it difficult to explain or even remember some of the intricate details that can be required in order to produce a concrete set of requirements. They typically know what they want the system to do, but they may not be capable of getting this information across to the developers at the early stages of the project, and it will often take a prototype version before they can provide a decent specification of the requirements. Of course, there is nothing new about this type of situation (Brooks 1987; Walz et al. 1993; Boehm 2000), but what it clearly demonstrates, lest we forget, is that the benefits provided by the agile development paradigm are negated unless we continue to use good old-fashioned user interface design principles and techniques.

4.3.3 Frequent Requests for Changes by Users

Travtech's approach to handling change requests depends on whatever agreement is in place with the customer. In a fixed-price contract the scope will have been signed off, and anything after that will probably be charged separately in addition, except if the change is very small. If the requested change is substantial, the additional cost will be negotiated with the customer. Revision of the estimate therefore is ad hoc and depends on a number of factors. The stage of the project at which the change is requested is a major factor in deciding whether or not to refine the estimate, as is the effort required and the relationship and contract agreed with the customer.

When asked if BrightSoft adhere to the agile principle of "welcome changing requirements", the project manager assuredly responded "Yes, fact of life for software engineering!", but their experience however is that constant feedback and streams of minor change requests can be overwhelming to the point where initial estimates get completely thrown out and the profitability of a project can be seriously threatened. In one such case, they have conceded that a project has effectively become a "loss leader".

MobilApp's approach is that if the customer submits a "must-have" requirement midway through the project, they will typically do their utmost to include it without affecting the schedule. Changes such as these are not caused by problems in the estimation process but rather because of problems in the requirements specification and poor definition of needs on the part of the customer. The schedule will typically have to be extended and the customer will be informed that this is due to their late change request. If there is an instance whereby the deadline cannot be extended, then the functionality will have to be revised and the time will be made up by omitting some other feature from the original set of requirements. This will all happen with agreement from the customer because it was their lack of understanding of their own requirements that drove the change.

This is one of the advantages that MobilApp find with agile methods, i.e. that the customer can see the mid results and at any stage have an input and reassess their requirements. If the customer does want to change the requirements, they can go back to the team who would then thrash out new requirements for that particular area and give an estimate for it. MobilApp find that once they have a project plan in place and have signed off with the customer on requirements, then these can be changed once the customer is made aware that the schedule for release will be pushed out as a result of the late change in requirements. In this way, costs can be controlled effectively.

4.4 Management Control

In addition to expert judgement, Travtech also use regression-based estimates where they look at previous data to compare the actuals with forecasted values and then conduct a variance analysis. This approach is especially useful on projects that have a number of iterations because data from earlier iterations can inform estimates for subsequent phases. Travtech have managed to improve their estimation capabilities not just by comparing whole projects to one another but also by comparing different phases within the same project. With agile iterations, it is possible to quickly identify through post-iteration reviews if the initial estimates have slipped, and subsequent estimates can then be revised accordingly to absorb any overruns. However, few customers are prepared to agree to an open-ended budget, and most want to sign fixed-price contracts up front. In those circumstances, the development company takes all the risk but Travtech find that the combination of experiences gained from previous projects, coupled with the greater control that comes from using atomic use cases or user stories as the basis of estimates, places them in a position where they can confidently price contracts.

Inaccurate estimates caused by IS management approval and control do not affect MobilApp to any significant extent. Generally, if a team member comes up with an estimate, this will be accepted by the management because they feel that everybody in the company has the experience and skills necessary to estimate fairly accurately without having it checked or validated by management. However, if a post-implementation audit reveals that the estimates for a project were 15 % or more out of line with actuals, management regards it as very important to use this information to try to rectify the inaccuracies. Whether it is early or late, they always try to find out where the issues were, what problems arose, why they got ahead or fell behind so much, and what are appropriate reference points for future project estimation.

Similarly, BrightSoft use estimates to audit projects afterwards to determine how successful the estimation techniques were. They conduct a postmortem analysis on all of the estimates for each project, and this helps them to review the accuracy of the projects that have just been finished, and it also enables them to create a new set of figures for future estimation.

At HPG, the estimates that are produced for projects are continuously evolving, and they are monitored closely at all times. When new pieces of work emerge in later iterations, the original estimate is re-examined, and this helps to keep the team members focused on the bottom line. In the past when projects were developed using traditional development methods, they often found that by the end of the project, the original estimates were redundant because of the changes that had occurred during the project. On projects that are using agile approaches, the estimate is continuously checked, and although inaccuracies still occur, the project team are aware of this as it is happening and can react accordingly.

4.5 Other Factors

Regarding the personnel who are involved in the projects, Travtech would be conscious of the skills and experience of those assigned to various tasks as this is likely to have an impact on their estimates. The estimate could change if customers were not willing to pay for more experienced people at a higher rate, in which case there would be a need to use people with less experience. In the preparation of the estimate, it is therefore important to have a good sense of who is actually going to be doing the work or at least the skill level or expertise that they will have.

Similarly, BrightSoft see new people and new technologies as a principal threat to their ability to produce accurate estimates. When people are learning on the job or learning a new technology, it is very difficult to know what to expect in terms of the time and effort required for them to become familiar and comfortable with the technology. A major factor that has contributed to the accuracy of BrightSoft's estimates is the maturity of the programmer, not only in terms of their skills and experience but also their own estimation capabilities. The accuracy of the entire estimate can depend on how well each individual estimates the work that they have to do themselves. If the person does not know their own capabilities or if they have not used the particular type of development process before, they are very likely to come up with an unrealistic estimate.

Table 55.2 Summary of findings: "agile" responses to traditional ISD cost-estimation problems

Traditional ISD cost-estimation problem	Response of companies in this study
Lack of an adequate methodology or guidelines for estimating	Short iterations, expert judgement
Inability to tell where past estimates failed	Post-iteration audits/retrospectives
Lack of setting and review of standard durations for use in estimating	Normal iteration = 2 weeks
Insufficient analysis when developing estimates	Planning game
Lack of co-ordination of systems development activities	Lean, lightweight processes, small teams
Pressures from managers, users, or others to change estimate	Reduce scope, swap in cheaper labour
Removal of padding from the estimate by management	No evidence
Reduction of project scope/quality to stay within estimate, leading to extra work later	No evidence
Red tape	Intractable, requires patience and diplomacy
Users' lack of understanding of their own requirements	Rapid prototyping, on-site customer
Frequent requests for changes by users	Change request process, renegotiate estimates
Users' lack of IT knowledge	Closer working relationship with customer
Poor or imprecise problem definition	Rapid prototyping, on-site customer
Performance reviews do not consider whether estimates were met	Post-iteration audits/retrospectives, stand-ups
Lack of project control comparing estimates and actuals	Post-iteration audits/retrospectives, stand-ups
Lack of careful examination of estimate by ISD management	Know your staff's capabilities, need to tweak unrealistic estimates, build in risk buffer

5 Summary and Conclusion

In their study of inaccurate estimates in traditional ISD projects, Lederer and Prasad (1995) identified 16 items, grouped under four factors, that were problematic. These items and the corresponding solutions advanced by the agile paradigm, as evidenced in our study, are presented in Table 55.2.

While agile methods have come some way to mitigating these cost-estimation problems, there is little evidence to suggest that the agile paradigm is any less prone to falling victim of shoddy analysis, stakeholder politics, or disengaged end users than traditional ISD cost-estimation approaches. Although Travtech were the only company to have experienced political pressures from customers or managers to reduce estimates, that is not of course to say that similar issues could not foreseeably arise in the other companies. Management control factors were not found to be a major cause of inaccuracies, although the need to keep a watchful eye on over-optimistic estimates by new staff, or staff moving into unfamiliar territory, was noted. All of the companies experienced user communication difficulties at some stage or another, and this is potentially a very serious threat to accurate estimates.

Travtech find that when inaccuracies do occur, it is typically due to some lack of understanding between the customers and developers regarding the requirements. It can also be due to a lack of technical expertise in a particular area which would prevent the accurate estimation of certain tasks. BrightSoft seem to be the most confident in their estimation abilities. Typically the estimates produced are relatively on target and if not, the discrepancy is usually negligible. They have found the major potential threats to accurate estimates to have been the introduction of new people, new technologies, and too much feedback from their customers. MobilApp find that change requests from customers and lack of estimation expertise can cause problems on some projects, particularly if a new development language is being used. HPG have found that their estimates are typically accurate to within 10 % of the actual figures; however, they feel that their inaccuracies may be due to their lack of formality in the estimation process.

To conclude, research on project cost estimation has been conducted for decades with a vast number of models and tools in existence. This study has looked at the estimation process in the emerging field of agile development and examined causes of inaccurate estimates and steps to improve the process. From the four case studies, a number of recommendations can be summarised as follows: estimation models are not a necessary component of the process, fixed-price budgets may be the best option for both developers and customers, and a critical success factor for agile cost estimation is that experience and past project data must be documented and used to guide the estimation of subsequent projects.

References

Abrahamsson P (2003) Extreme programming: first results from a controlled case study. In: Proceedings of the 29th Euromicro conference, Belek-Antalya

Abrahamsson P, Conboy K, Wang X (2009) "Lots Done, More To Do": the current state of agile systems development research. Eur J Inf Syst 18(4):1–7

Agarwal R, Kumar M, Yogesh, Mallick S, Bharadwaj RM, Anantwar D (2001) Estimating software projects. ACM SIGSOFT Softw Eng Notes 26(4):60–67

Alford MW, Lawson JT (1979) Software requirements engineering methodology (Development), RADC-TR-79-168, U.S. Air Force Rome Air Development Center, June 1979

Baskerville R, Travis J, Truex D (1992) Systems without method: the impact of new technologies on information systems development projects. In: Kendall KE, Lyytinen K, DeGross J (eds) The impact of computer supported technologies on information systems development. Elsevier Science Publishers, Amsterdam, pp 241–269

Boehm BW (1981) Software engineering economics. Prentice Hall, Englewood Cliffs

Boehm B (2000) Requirements that handle IKIWISI, COTS, and rapid change. IEEE Comput 33 (7):99–102

Boehm BW, Abts C, Chulani S (2000) Software development cost estimation approaches: a survey. Ann Softw Eng 10(1–4):177–205

Brooks FP (1987) No silver bullet/essence and accidents of software engineering. IEEE Comput 20(4):10–18

Caminer DT (1958) And how to avoid them. Comput J 1(1):11–14

Cao L (2008) Estimating agile software project effort: an empirical study. In: Proceedings of Americas conference on information systems (AMCIS)
Ceschi M, Sillitti A, Succi G, De Panfilis S (2005) Project management in plan-based and agile companies. IEEE Softw 22(3):21–27
Chapman C, Ward S (2002) Managing project risk and uncertainty: a constructively simple approach to decision making. Wiley, Chichester
Conboy K (2009) Agility from first principles: reconstructing the concept of agility in information systems development. Inf Syst Res 20(3):329–354
Conboy K (2010) Project failure en mass: a study of loose budgetary control in ISD projects. Eur J Inf Syst 19(3):273–287
Dybå T, Dingsøyr T (2008) Empirical studies of agile software development: a systematic review. Inf Softw Technol 50(9–10):833–859
Elssamadisy A, Schalliol G (2002) Recognizing and responding to "Bad Smells" in extreme programming. In: Proceedings of the 24th international conference on software engineering, Orlando, Florida
Ewusi-Mensah K, Przasnyski ZH (1995) Learning from abandoned information systems development projects. J Inf Technol 10(1):3–14
Fitzgerald B (2000) Systems development methodologies: the problem of tenses. Inf Technol People 13(3):174–185
Grudin J (1991) Interactive systems: bridging the gaps between developers and users. IEEE Comput 24(4):59–69
Highsmith J (2001) History: the Agile Manifesto. http://agilemanifesto.org/history.html
Highsmith J (2003) Agile project management: principles and tools. Cutter consortium
Highsmith J, Cockburn A (2001) Agile software development: the business of innovation. IEEE Comput 34(9):120–127
Jørgensen M (2003) How much does a vacation cost? Or what is a software cost estimate? ACM SIGSOFT Softw Eng Notes 28(6):1–4
Jørgensen M, Moløkken K (2003) A preliminary checklist for software cost management. In: Proceedings of the 3rd international conference on quality software
Jørgensen M, Shepperd M (2007) A systematic review of software development cost estimation studies. IEEE Trans Softw Eng 33(1):33–53
Jones JC (1988) Softechnica. In: Thackara J (ed) Design after modernism: beyond the object. Thames & Hudson, London, pp 216–226
Jones C (2003) Why flawed software projects are not cancelled in time. Cutter IT J 16(12):12–17
Jurison J (1999) Software project management: the manager's view. Commun AIS 2(3):1–50
Keil M, Carmel E (1995) Customer-developer links in software development. Commun ACM 38 (5):33–44
Lang M (2009) The influence of short project timeframes on web development practices: a field study. In: Proceedings of 18th international conference on information systems development, Nanchang, China
Lang M, Fitzgerald B (2007) An empirically-grounded conceptual framework of situated web design practices. Requir Eng J 12(4):203–220
Lederer AL, Prasad J (1995) Perceptual congruence and systems development cost estimation. Inf Resour Manage J 8(4):16–27
Lovaasen G (2001) Brokering with eXtreme programming. In: XP Universe 2001. Raleigh, North Carolina
McCracken DD, Jackson MA (1982) Lifecycle concept considered harmful. Softw Eng Notes 7 (2):29–32
Miles MB, Huberman AM (1994) Qualitative data analysis: an expanded sourcebook, 2nd edn. Sage, Thousand Oaks
Moløkken K, Jørgensen M (2003) A review of software surveys on software effort estimation. In: Proceedings of the 2003 international symposium on empirical software engineering

Naur P, Randell B (1969) Software engineering: report on a conference sponsored by the NATO Science Committee, Garmisch, Germany, 7–11 Oct 1968. Scientific Affairs Division, NATO, Brussels

Neill CJ (2003) The extreme programming bandwagon: revolution or just revolting? IT Prof 5 (5):62–64

Orr K (2004) Agile requirements: opportunity or oxymoron? IEEE Softw 21(3):71–73

Paulk MC (2002) Agile methodologies and process discipline. CrossTalk, J Defense Softw Eng (October):15–18

Ramesh B, Cao L, Baskerville R (2007) Agile requirements engineering practices and challenges: an empirical study. Inf Syst J 20(5):449–480

Robey D, Markus ML (1984) Rituals in information system design. MIS Q 8(1):5–15

Russo NL, Stolterman E (2000) Exploring the assumptions underlying information systems methodologies. Inf Technol People 13(4):313–327

Schalliol G (2001) Challenges for analysts on a large XP project. In: XP Universe 2001. Raleigh, North Carolina

Stamelos I, Angelis L (2001) Managing uncertainty in project portfolio cost estimation. Inf Softw Technol 43(13):759–768

Taber C, Fowler M (2000) An iteration in the life of an XP project. Cutter IT J 13(11):13–21

Tan C, Teo H (2007) Training future software developers to acquire agile development skills. Commun ACM 50(12):97–98

The Standish Group (1995) The CHAOS report

Walz DB, Elam JJ, Curtis B (1993) Inside a software design team: knowledge acquisition, sharing, and integration. Commun ACM 36(10):63–77

Winklhofer H (2002) Information systems project management during organizational change. Eng Manage J 14(2):33–38

Index

R. Pooley et al. (eds.), *Information Systems Development: Reflections, Challenges and New Directions*, DOI 10.1007/978-1-4614-4951-5,

C

D