Marc Aiguier • Yves Caseau
Daniel Krob • Antoine Rauzy
Editors

Complex Systems Design & Management

Proceedings
of the Third International Conference
on Complex Systems Design
& Management
CSD&M 2012

CESAMES
Center of Excellence on Systems Architecture,
Management, Economy and Strategy

Springer

Complex Systems Design & Management

Marc Aiguier, Yves Caseau, Daniel Krob,
and Antoine Rauzy (Eds.)

Complex Systems Design & Management

Proceedings of the Third International Conference on Complex Systems Design & Management CSD&M 2012

Springer

Editors

Prof. Marc Aiguier
Professor at Ecole Centrale de Paris
Grande Voie des Vignes
Châtenay-Malabry
France

Dr. Yves Caseau
Bouygues Telecom
Issy-les-Moulineaux
France

Prof. Daniel Krob
Ecole Polytechnique
LIX/DIX
Palaiseau Cedex
France

Prof. Antoine Rauzy
Ecole Polytechnique
LIX/DIX
Palaiseau Cedex
France

ISBN 978-3-642-34403-9 e-ISBN 978-3-642-34404-6
DOI 10.1007/978-3-642-34404-6
Springer Heidelberg New York Dordrecht London

Library of Congress Control Number: 2012950366

Printed on acid-free paper

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

Preface

Introduction

This volume contains the proceedings of the Third International Conference on "Complex System Design & Management" (CSD&M 2012; see the conference website: http://www.csdm2012.csdm.fr for more details).

The CSD&M 2012 conference was jointly organized by the research & training Ecole Polytechnique - ENSTA ParisTech - Télécom ParisTech - Dassault Aviation - DCNS - DGA - Thales chair "Engineering of Complex Systems" and by the non profit organization C.E.S.A.M.E.S. (Center of Excellence on Systems Architecture, Management, Economy and Strategy) from December 12 to December 14 at the Cité Internationale Universitaire of Paris (France).

The conference benefited of the permanent support of many academic organizations such as Ecole Centrale de Paris, Ecole Nationale Supérieure des Techniques Avancées (ENSTA), Ecole Polytechnique, Ecole Supérieure d'Electricité (Supélec), Université Paris Sud 11 and Télécom ParisTech which were deeply involved in its organization. A special thank also goes to Dassault Aviation, DCNS, DGA, MEGA International and Thales companies which were the main industrial sponsors of the conference. All these institutions helped us a lot through their constant participation to the organizing committee during the one-year preparation of CSD&M 2012. Last, but not least, we would also like to point out the assistance of EADS in the same matter.

Why a CSD&M Conference?

Mastering complex systems requires an integrated understanding of industrial practices as well as sophisticated theoretical techniques and tools. This explains the creation of an annual *go-between* forum at European level (which did not existed yet) dedicated it both to academic researchers and industrial actors working on complex industrial systems architecture and engineering in order to facilitate their *meeting*. It was actually for us a *sine qua non* condition in order to nurture and develop in Europe this complex industrial systems science which is now emergent.

The purpose of the "Complex Systems Design & Management" (CSD&M) conference is exactly to be such a forum, in order to become, in time, *the* European academic-industrial conference of reference in the field of complex industrial systems architecture and engineering, which is a quite ambitious objective. The first CSD&M 2010 and CSD&M 2011 conferences – which were held in October 2010 and in December 2011 in Paris – were the first steps in this direction with more than

200 and 250 participants coming from 20 different countries with an almost perfect balance between academia and industry.

The CSD&M Academic–Industrial Integrated Dimension

To make the CSD&M conference this convergence point of the academic and industrial communities in complex industrial systems, we based our organization on a principle of *complete parity* between academics and industrialists (see the conference organization sections in the next pages). This principle was first implemented as follows:

- the Programme Committee was composed of 50% academics and 50% industrialists,
- the Invited Speakers came equally from academic and industrial environments.

The set of activities of the conference followed the same principle. They indeed consist of a mixture of research seminars and experience sharing, academic articles and industrial presentations, software and training offers presentations, etc. The conference topics cover in the same way the most recent trends in the emerging field of complex systems sciences and practices from an industrial and academic perspective, including the main industrial domains (transport, defense & security, electronics & robotics, energy & en- vironment, health & welfare services, media & communications, e-services), scientific and technical topics (systems fundamentals, systems architecture & engineering, systems metrics & quality, systemic tools) and system types (transportation systems, embedded systems, software & information systems, systems of systems, artificial ecosystems).

The CSD&M 2012 Edition

The CSD&M 2012 edition received 67 submitted papers, out of which the program committee selected 18 regular papers to be published in these proceedings and 3 complementary industrial full presentations, corresponding to a 28% acceptance ratio which is fundamental for us to guarantee the high quality of the presentations. The program committee also selected 28 papers for a collective presentation in the poster session of the conference.

Each submission was assigned to at least two program committee members, who carefully reviewed the papers, in many cases with the help of external referees. These reviews were discussed by the program committee during a physical meeting held in Ecole Nationale Supérieure des Techniques Avancées (ENSTA ParisTech, Paris) by June 19, 2012 and via the EasyChair conference management system.

We also chose 17 outstanding speakers with various industrial and scientific expertise who gave a series of invited talks covering all the spectrum of the conference during the two first days of CSD&M 2012, the last day being dedicated to a special EEC vision session and to the presentations of all accepted papers. The first day of the conference was especially organized around a common topic – Risk and Safety – that gave a coherence to all the initial invited talks.

Futhermore, we had a poster session, for encouraging presentation and discussion on interesting but "not-yet-polished" ideas, and a software tools presentation session, in order to provide to each participant a good vision on the present status of the engineering tools market offer.

Acknowledgements

We would like finally to thank all members of the program and organizing committees for their time, effort, and contributions to make CSD&M 2012 a top quality conference. A special thank is addressed to the team of CESAMES non profit organization which managed permanently with a huge efficiency all the administration, logistics and communication of the CSD&M 2012 conference (see http://www.cesames.net).

The organizers of the conference are also greatly grateful to the following sponsors and partners without whom the CSD&M 2012 would not exist:

- **Academic sponsors**
  - Commissariat à l'Energie Atomique (CEA-LIST),
  - Ecole Centrale de Paris,
  - Ecole Polytechnique,
  - Ecole Supérieure d'Electricité (Supélec),
  - ENSTA PariTech,
  - Télécom ParisTech,
- **Industrial sponsors**
  - Bouygues Telecom,
  - Dassault Aviation,
  - DCNS,
  - DGA,
  - EADS,
  - Ecole Polytechnique - Microsoft Chair "Optimization for Sustainable Development",
  - EDF,
  - Thales,
  - Mega International
  - Orange – Ecole Polytechnique – Télécom ParisTech Chair "Innovation & Regulation"

- **Institutional sponsors**
  - Digiteo labs,
  - Région Ile-de-France,
  - Ministère de l'Enseignement Supérieur et de la Recherche,
- **Supporting partners**
  - Association Française d'Ingénierie Système (AFIS),
  - Institut du Maitrise du Risque (IMdR),
  - International Council on Systems Engineering (INCOSE),
  - Pôle de compétivité System@tic,
- **Participating partners**
  - Atego,
  - Dassault Systèmes,
  - Esterel Technologies,
  - IBM Rational Software,
  - Knowledge Inside,
  - MathWorks,
  - Obeo,
  - PragmaDev,
  - Project Performance International.
  - The CosMo Company

Paris, July 31, 2012

Marc Aiguier – Ecole Centrale de Paris
Yves Caseau – Bouygues Telecom
Daniel Krob – Ecole Polytechnique & CESAMES
Antoine Rauzy – Ecole Polytechnique

Conference Organization

Conference Chairs

- *General Chair*
  - Daniel Krob, institute professor, Ecole Polytechnique, France
- *Organizing Committee Chair*
  - Marc Aiguier, professor, Ecole Centrale de Paris, France
- *Program Committee Chairs*
  - Antoine Rauzy, associate professor, Ecole Polytechnique, France (academic co-chair of the Program Committee)
  - Yves Caseau, executive vice president "Technologies, Services & Innovation", Bouygues Telecom, France (industrial co-chair of the Program Committee)

Program Committee

The PC consists of 30 members (15 academic and 15 industrial): all are personalities of high international visibility. Their expertise spectrum covers all of the conference topics.

Academic Members

- *Co-Chair*
  - Antoine Rauzy, Ecole Polytechnique, France

- *Other Members*
  - Farhad Arbab, Leiden University, Netherlands
  - Sabine Buckl, TUM, Germany
  - Brian Cameron, Pennsylvania State University, USA
  - Olivier De Weck, MIT, USA
  - Wolter Fabrycky, Virginia Tech, USA
  - Patrick Godfrey, University of Bristol, Great Britain

  - Michaël Hinchey, University of Limerick, Ireland
  - Leon Kappelman, University of North Texas, USA
  - Dimitri Mavris, Georgia Tech, USA
  - Franck Ortmeier, Universität Madgeburg, Germany
  - Sylvain Peyronnet, Université Paris Sud, France
  - Marc Pouzet, Ecole Normale Supérieure, France
  - Bernhard Rumpe, Aachen Universität, Germany
  - Akira Yamaguchi, Osaka University, Japan

Industrial Members

- *Co-Chair*

  - Yves Caseau, Bouygues Telecom, France

- *Other Members*

  - Erik Aslaksen, Sinclair Knight Merz, Australia
  - Jean-Pierre Daniel, AREVA, France
  - Paul Davies, Thales, Great Britain
  - Aymeric Fillon, Safran, France
  - Hans-Georg Frischkorn, Verband der Automobilindustrie, Germany
  - Thierry Gueguen, Dassault Systèmes, France
  - Rudolf Haggenmüller, ITEA2, Germany
  - Cecilia Haskins, INCOSE, Norway
  - David Johnson, ABS Consulting, USA
  - Dominique Luzeaux, Direction Générale de l'Armement & AFIS, France
  - Andreas Mitschke, EADS, Germany
  - Thierry Pardessus, Airbus, France
  - Bran Selic, Malina Software, Canada
  - David Walden, Sysnovation & INCOSE, USA

Organizing Committee

- *Chair*

  - Marc Aiguier, professor, Ecole Centrale de Paris, France

- *Other Members*

  - Emmanuel Arbaretier, EADS, France
  - Karim Azoum, System@tic, France
  - Guy Boy, Florida Institute of Technology, USA
  - Isabelle Demeure, Télécoms ParisTech, France
  - Gilles Fleury, Supélec, France
  - Pascal Foix, Thales, France

- Vassilis Giakoumakis, University of Amiens, France
- Eric Goubault, CEA, France
- Omar Hammami, ENSTA ParisTech, France
- Clotilde Marchal, EADS, France
- Maximilien Nayaradou, Finance Innovation, France
- Isabelle Perseil, INSERM, France
- Srini Ramaswamy, University of Arkansas at Little Rock, USA
- Fatiha Zaïdi, University Paris Sud 11, France

Invited Speakers

Societal Challenges: Risk & Safety

- Akira Yamaguchi, professor, Osaka University - Japan
- Raphaël Douady, founder & head of research, Riskdata S.A. - France
- Pierre Bornard, vice president, Réseau de Transport d'Electricité (RTE) - France
- Bernard Laporte, head of R&D and Training, AXA - France

Industrial Challenges: Risk & Safety

- David Walden, chairman, Sysnovation - USA
- Olivier Flous, technical director of Transport & Safety, Thales Communications & Security S.A. - France
- Jean-Claude Visier, head of the Energy, Health and Environment Department, Centre Scientifique et Technique du Bâtiment - France
- Pascal Medal, head of Certification Experts Department, European Aviation Safety Agency - Germany

Scientific State-of-the-Art

- Wolter Fabrycky, professor, Virginia Tech - USA
- Manfred Broy, professor, Technische Universität München - Germany
- Dinesh Verma, professor, Stevens University - USA
- Yves Caseau, executive vice president "Technologies, Services & Innovation", Bouygues Telecom - France

Methodological State-of-the-Art

- Antoine Rauzy, associate professor, Ecole Polytechnique – France
- Patrick Godfrey, professor, University of Bristol - UK

- Anas Alfaris, co-director of the Center for Complex Engineering Systems (CCES) at KACST and MIT - Saudi Arabia/USA
- Pierre-Etienne Labeau, professor, Ecole Polytechnique de Bruxelles - Belgium

EEC Session

- Khalil Rouhana, director of “Components & Systems” in the Directorate – General for “Communications Networks, Content & Technology” – European Commission

Contents

Chapter 1
Engineering Cyber-Physical Systems: Challenges and Foundations

Manfred Broy

Abstract. Cyber-Physical Systems are the next step into globally integrated software systems. They are the result of the combination of embedded system with cyberspace. Cyber-Physical Systems support real world awareness in the Internet and the access to global data and services by embedded system. The engineering challenge for Cyber-Physical System is the combination of characteristic properties of embedded systems such a real time, functional safety, dependability, closedness with characteristic properties of the internet such ad openness, partial availability, restricted quality of service and reduced dependability.

1 Introduction – Cyber-Physical Systems

The term Cyber-Physical Systems addresses a new type of systems which is the result of an amalgamation of embedded software systems, connected on one hand to their physical environment by sensors and actuators and global networks such as the Internet with its data and services. By Cyber-Physical Systems the Internet gets real world aware and embedded systems become location independent, can be connected on a global scale, and get access to global data and services.

In line with Moore's Law, ongoing advances in very large-scale digital circuits integration enable electronic components ever smaller, more powerful, and cheaper. As a result, devices and objects are increasingly be equipped with "invisible" embedded systems connected directly to the physical world through a range of sensors and actuators, and that, consequently, can be deployed in a broad range of applications in ways allowing them to be controlled, monitored, and networked. Global networks such as the Internet connect embedded computers, their data, their services and their applications.

Manfred Broy
Technische Universität München
Institut fuer Informatik
Boltzmannstrasse 3
D-85748 Garching
Germany
e-mail: broy@in.tum.de

2 Key Capabilities of Cyber-Physical Systems

Cyber-Physical Systems lead into increasing openness, complexity, autonomy, “smartness” and evolution of the systems (with disruptive effects in the fields of application). To realize and master the development of Cyber-Physical Systems we need a number of capabilities both in engineering and within the systems as described in detail in the following.

2.1 Cyber-Physical, Sensors/Actuators, Networked (Local-Global), Virtual, Real-Time Management

Key issue for Cyber-Physical Systems (CPS) is the collection and acquisition of data such as parallel data collection (via sensors), data fusion, processing of physical data from the environment, locally, globally and in real time. Data from sensors combined with data fusion, data mining and interpretation enable physical awareness of systems. The interpretation aims at regarding achievements of objectives and tasks of Cyber-Physical Systems. In the acquisition and interpretation techniques of deduction are useful combined with the prediction of faults, obstacles, and risks. A key property of Cyber-Physical Systems is the interaction between Cyber-Physical Systems and their context consisting of users, the physical environment and systems and services from the cloud. This requires interoperation, integration, rules for and control of CPS-components and functions in globally distributed, networked real-time control and regulation.

2.2 Systems of Systems (sos), Controlled Networks with Dynamic Boundaries

At a higher, more domain specific level, the interpretation of context and situation evaluating data over several levels, depending on different application situations becomes essential. To offer comprehensive functionality a systematic selection, incorporation, coordination, and use of services – depending on specific situations, local and global objectives, and behavior is indispensable. Using services in the cloud, service discovery, composition, and integration is needed. Due to the many sub-systems working in parallel patterns of decentralized control have to be developed controlling the recognition of missing services, data, functions and active search and dynamic integration.

Finally a high degree of autonomic behavior and self-organization is to be achieved.

For the evaluation of benefit and quality required for the application concepts of Quality of Service (QoS) and overall system quality of components and services is to be incorporated – also regarding possible risks to guarantee dependability including reliability and compliance with respect to guaranteed QoS. Security issues, such as the controlled access to system's own data and services, have to be solved.

2.3 Context-Adaptive and (Partially) Autonomous Systems

Context awareness is to result in a comprehensive, continuous context awareness on the basis of a continual collection, observation, selection, processing, evaluation, decision-making, communication of context data, situation and application data. Context awareness enables a systematic adaptation of the interaction, coordination, control with/of other systems and services.

A more advanced adaptive behavior requires the recognition, analysis, and interpretation of plans and intentions of objects, systems and participating users. This is mastered in engineering by model creation for application field and domain, for participants, including their roles, objectives and requirements, available services and tasks. This has to include the assessment of objectives, taking into consideration alternatives with regard to costs and risks.

A further step into Cyber-Physical Systems leads to self-awareness in terms of knowledge about the systems' own situation, status and options for action. This requires the learning of, for example, modified work processes, logistics processes, habits, interaction, etc. and corresponding behavior adaption finally providing capacity for self-organization.

2.4 Cooperative Systems with Distributed, Changing Control

The integration of a high number of systems requires distributed, cooperative and interactive perception and evaluation of the situation and the distributed, cooperative and interactive determination of the steps to be carried out – depending on the evaluation of the situation, on the objectives of individual participants and on the objectives of the community these participants belong to (local vs. global objectives). This cannot be done without subsequent coordinated assessment and negotiation of the decision ultimately taken, i.e. self and shared control and decision-making autonomy.

To achieve such a level of autonomy requires decision-making on the basis of uncertain knowledge, cooperative learning and adaption to situations and requirements estimatiing the quality of own and external services and abilities.

A special goal is the coordinated processing of mass data as they are produced by billions of embedded systems.

2.5 Extensive Human/System Cooperation

The future does not lie so much in completely autonomous. Humans will interact with such systems in various ways. Thus, in general, Cyber-Physical Systems will offer several user interfaces. This requires intuitive, multimodal, active and passive HMI – support (with simplified control) and support of a broader (space, time) perception and capacity to act for individuals and groups. The systems need to be able to recognize and interpret of human behavior including emotions, needs and intentions. For adaptive behavior acquisition and evaluation of data concerning state and context of human and system (extension of perception and evaluation skills) is required. A capability difficult to achieve is the integrated and interactive decisions and actions between systems and individuals or groups.

On the long run the ability to learn will lead to more flexible user interfaces as they are required for mobile access to Cyber-Physical Systems.

2.6 Key Capabilities and Non-functional Requirements Quality in Use, Quality of Service (qos)

On the long run the significant capabilities of Cyber-Physical Systems are context awareness (correct perception and interpretation of situation and context). To some extent self-awareness in reflecting the systems' role and situation in an operational context, third party-awareness, human awareness (status, objectives, intentions, ability to act) is needed. Key capabilities are learning and adaption (behavior), self-organization, flexible cooperation, negotiation and decision-making (within defined boundaries following some specified compliance).

Partial autonomy requires decision-making on the basis of uncertain knowledge.

Key quality requirements are provision and maintenance of QoS guarantees as well as comprehensive policies for safety and security - proactive, strategic and reliable actions and privacy protection.

The key area of human-centric systems requires transparent (HMI), shared control, integrated situation evaluation and predictable actions.

Finally the systems are operating in a number of critical areas. This needs careful risk management.

3 Engineering Challenges – Foundations of CPS Engineering

Cyber Physical Systems include concepts from embedded systems, from engineering HMIs, and from global networks, such as the Internet or the cloud. Therefore it is necessary to find engineering foundations for all of these, which is more then just putting together existing and integrate engineering foundations for these areas. In fact, it is necessary to unify and harmonise these foundations and to deal with additional effects and concepts that arise from the merger.

3.1 Modeling Cyber-Physical Systems – Semantic Foundations

Cyber-Physical Systems incorporate a large variety of concepts. Characteristic examples are real world awareness and adaptivity that make it necessary to capture all kinds of physical issues. Moreover, the connectivity leads to additional challenges.

There are a number of semantic models around, in particular, for distributed interactive systems that work for real time like embedded systems. These models, in principle, can be extended to communication systems and connectivity.

3.1.1 Distribution and Interaction

Interaction is a first class concept in Cyber-Physical Systems due to their extensive connectivity. This comprises both interactions between systems and their physical environment as well as the interaction between systems and their users, and also the interaction between different context systems and sub-systems. Talking about systems of systems the interaction and interoperability between the systems is most essential.

Distribution is also something, which is usually found inside systems. However, for Cyber-Physical Systems we do not only have to consider just a logical distribution concept as found in a number of approaches and computer science, we need a real world aware kind of distribution in space, where we can relate distribution to the geographical view on the physical world.

3.1.2 Statically Adaptive Systems and Dynamically Adaptive Systems

As already pointed out, one of the important concepts for Cyber-Physical Systems is that they are adaptive. There are differences in statically adaptivity and dynamically adaptivity.

Static adaptivity addresses adaptivity at the user interface. This means that a system may change its behavior from the viewpoint of the user depending on additional information not exchanged directly by the user, but on different channels and that are dealt with the help of context models.

If we look at the distribution of systems and their connectivity and the number of components, then for Cyber-Physical Systems it seems necessary that these structures permit that new components are introduces, new channels are generated, and channels find different connections. This means a highly dynamic structural adaptivity that has to be captured and supported in the models.

Models available so far such as π-calculus or ambient calculus are only very specific models not very integrated with further modeling concepts. We need generalized models for capturing all kinds of dynamics and adaptivity of Cyber-Physical Systems.

3.1.3 Provisioning/Deployment

Although we believe that most of the modeling issues have to be addressed at an abstract virtual level, it is also necessary to relate virtual functions, services, and processes to concrete technical processes. This means that we have to model deployment and scheduling but in contrast to embedded Systems we assume that a dynamic deployment is also necessary and needed.

3.1.4 Modeling Time and Space

Time is a fact of the physical world and real life. Therefore it has to be captured by models for embedded systems. Anyhow time is also an issue in global networks, however, there we usually talk about soft real time properties while in embedded systems often hard real time properties have to be addressed.

Therefore a flexible timing model has to be introduced for Cyber-Physical Systems.

Space distribution is a fact for Cyber-Physical Systems. Space and distribution has to be captured including concepts of mobility.

3.1.5 Modularity and Compositionality

Modularity means that abstraction techniques such as interface abstraction are compatible with refinement.

Modeling techniques for Cyber-Physical Systems have to be highly modular. This means that they have to be modular with respect to their structure and composition. Thus we need a notion of an interface which is good enough to carry out composition at the interface in the sense that from the interface description of the systems and their behavior we can construct interface description of composed systems and their behavior.

Modularity has to support the possibilities to select a number of different scopes for a Cyber-Physical System to concentrate on a variety of views and crosscutting concerns.

3.1.5.1 Extraction of Cross-Cutting Behaviors

Cyber-Physical Systems typically show cross cutting behaviors. Therefore apart from the interface behaviors, it is interesting to develop a number of views on the behavior, in particular to capture cross cutting behavior.

The modularity of the models has to support this extraction of the views.

3.1.6 Rely/Guarantee

It has to be possible to model all the relevant properties of the context of the systems. That means we have to understand the interaction between the system and its context. This is of high interest, both at the level of physicality and at the level of connectivity. To do that we need specification techniques that explicitly support the formulations of assumptions about the context.

3.1.7 Operational "Modes"

Cyber-Physical Systems run in different operational modes. A service has to determine the current operational "mode". This is a form of adaptivity.

Modes are a good way to capture feature interaction between the different functions offered by a Cyber-Physical System.

3.2 Modeling Techniques and Methods in the Development Process

What is needed for engineering Cyber-Physical Systems is a set of modeling techniques, which are strong enough to capture all the notions that have been treated above. In addition, we have to be able to use these models in the classical important steps of system development, such as requirements capture, and representation of architecture views as well as refinement.

3.2.1 Requirements Capture

The models have to be able to structure and reflect all important requirements. This includes functional, behavioral, and non-functional requirements. Only a structuring along the lines of reference models may allow us to deal with the enormous number of requirements for Cyber-Physical Systems.

3.2.2 Architecture

The models in addition have to be able to model a number of views at the level of architecture. This includes levels of abstraction such as a functional view, a logical subsystem view and a technical view. These different views have again to be related to each other and to the requirements.

3.3 Programming Techniques and Technologies

So far the programming technologies that are around today are not good enough to deal with Cyber-Physical Systems as required. The properties that have been described also with respect to modeling, showing concepts and capabilities to be captured at the level programming also.

Therefore we need programming languages that reflect physicality and real world awareness in their concepts such as time and geographical distribution as well as issues of system dynamics. They have to scale, they have to be able to be location aware and time aware and they have to be able to express certainly specific quality concerns. On the long run new programming languages are required that reflect all these issues.

3.4 Quality

Software quality is a multi-faceted topic even for conventional software systems. Moving towards Cyber-Physical Systems we have to bring together issues of quality of software with issues of quality for systems for communication networks and for man-machine-interfaces. This needs an extensive quality model capturing all the different quality aspects one can find for systems of that type.

3.5 Techniques and Technologies

Cyber-Physical Systems show a number of important properties that have to be captured by the modeling techniques for Cyber-Physical System. This includes complex event processing, time trigger systems and enterprise service collections. These are parts of the architectures that have also to be reflected at a level of modeling and at the level of programming languages.

4 System Modeling

There is a wide variety of subjects to be modeled. In this section we study the modeling of systems.

4.1 What Is a System?

The term system is widespread and is used with many different meanings. Below, we focus on the conceptual notion of system, as used in software and systems engineering.

Definition: System
A system (Greek σύστημα, systematic - literally, the structure, prepackaged, associated) is a set of related elements that interact in an organized way, to achieve and purposefully conduct a common goal.

The resulting characteristics of a system are its

- system boundary separating the system more or less clearly from its environment,
- internal structure,
- clear and usually dedicated interaction with its environment via its system boundary.

The system boundary is determined by the system interfaces through which the system interacts with the environment. A system shows in its interior and via its system boundary some behavior that can be captured by the change of characteristic system attributes over time. At any given time a system and its characteristic attributes have a state. The change of state is continuous or discrete.

- The state of a system in its interior and its system boundary can be characterized by system attributes.
- The temporal behavior of the system as a transition between system states can be described according to a continuous or discrete change of its attributes.
- The interaction of the system with its environment through the system interfaces and the temporal interaction behavior can be captured by the flow of information, energy, and matter as well as the attribute changes.

The internal structure of a system can be described by

- a set of sub-systems ("components"), which interact with each other and via the system boundaries with the environment, and/or
- its states and state transitions.

A system may be a subsystem of an overall system, since systems can be composed with other systems if the systems fit together at their boundaries (interfaces).

4.2 System Models

System models are used to describe, design and analyze systems. A system model is an abstraction of a system, which usually serves a purpose. Each system model has the purpose to reflect some conclusions about the modeled system.

System models are a basis for the specification of systems. A system model is

- *valid* for a system if the statements that result from the model are valid for the system,
- *modular* if the interface behavior of the system can be composed from the interface behavior of its subsystems,
- *abstract*, if they do not contain any details that do not serve the purpose.

By definition, system behavior has to incorporate notions of time. System models can be based on discrete or continuous time. In the following a model for systems is introduced, which is a general, but specifically mathematical representation for systems behavior [Broy, Stølen 01], [Broy 07b].

A system interacts with its environment through a set of channels. The channels are directed communication devices and serve the information exchange between system and environment. The channels are divided into input and output channels.

On each channel a stream of messages is sent during a system's execution. We distinguish the following types of streams (let $\mathbb{N}_+ = \mathbb{N}\backslash\{0\}$, $\mathbb{R}_+ = \{ t \in \mathbb{R}: t > 0 \}$ be discrete and continuous models of time):

Time discrete channels are channels that operate with discrete time (values from a discrete set of values or a set of continuous values M). Values from a given set M of values are transmitted in a discrete stream

$s: \mathbb{N}_+ \to \mathtt{M}$	exactly one message per discrete time slot
$s: \mathbb{N}_+ \to \mathtt{M} \cup \{-\}$	at most one message per discrete time slot
$s: \mathbb{N}_+ \to \mathtt{M}^*$	a sequence of messages at each discrete time slot

Channels with continuous time work with streams based on continuous time. M denotes a set of values, where the set can be discrete or continuous (for instance real numbers). Values from M are transmitted at discrete time points, let set $D \subseteq \mathbb{R}_+$ discrete; we get streams

$$s: D \to \mathtt{M}$$

Channels with continuous time, where values are continuously transmitted operate with discrete time. The values of M may be discrete or non discrete, for example, given by the real numbers (in compliance with streams that represent continuous functions); we get streams

$$s: \mathbb{R}_+ \to \mathtt{M}$$

By `Stream` we denote the set of all streams, discrete or continuous. In a typed channel set C for each channel a set of values M and a time concept is fixed. For a set of channels C a function

$$x: \mathtt{C} \to \mathtt{Stream}$$

denotes a channel history, if the streams assigned to the channels match the defined sets of values for the channels and their time concepts.

With H[C] we denote the set of channel histories for channel set C.

The behavior of a system S with a set E of input channels and a set A of output channels via its interface is described in the deterministic case as follows:

$$f_S : H[E] \to H[A]$$

It is required that f_S has a directional flow of time. f_S has a directional time flow, if for some $\delta \geq 0$ we have (for all $t \in \mathbb{R}_+$, $x, x' \in H[E]$)

$$x{\downarrow}t = x'{\downarrow}t \Rightarrow f_S(x){\downarrow}t+\delta = f_S(x'){\downarrow}t+\delta$$

where $x{\downarrow}t$ denotes the prefix of the stream x until (including) time t.

If this formula holds for $\delta > 0$, then the interface behavior f_S is called strongly causal, otherwise weakly causal. Causality models the cause/effect principle for systems following the progress of time.

The internal behavior of a system can be described both as a state machine and as a family of interacting subsystems. More precisely, there exists for each system, both

- a state view in which the system is assigned a state at any time, and
- a structural view, where the system is decomposed (from a logical technical, or architectural point of view) into a number of subsystems (called components).

The architectural view may be trivial if the system contains no real subsystems. If the system contains subsystems, which in turn contain sub-subsystems, we speak of a hierarchical structure of the system architecture.

We compose systems, creating a composite system that contains the systems as subsystems. The state view is then obtained from composition the state perspective of the subsystems. If the interface of the composite system is obtained from the interfaces in the subsystems, then the system model is called modular.

For models in software and systems engineering a number of principles have been worked out that are not found in models of mechanical engineering or only in a very simplistic variant. These principles are hierarchical decomposition, modular modeling and levels of abstractions. Although abstraction is used in engineering a lot, there does not exist an explicit concept of levels of abstractions in traditional engineering.

The system model we have introduced is quite general. It comprises time and concurrency and can be rather seen as an abstract model of behavior of real world systems in terms of a continuous and discrete meaning.

4.3 Probabilistic Interface View

For the logical model of system behavior we introduce a probabilistic model (see [Neubeck 12]). Given a set of typed channels C we define a probability distribution for a set $G \subseteq H[C]$ by the function

$$\mu: G \to [0:1]$$

Let $\mathcal{M}[\mathtt{H[C]}]$ denote the set of all probability distributions over sets $\mathtt{G} \subseteq \mathtt{H[C]}$.

Given a behavior

$$\mathtt{F}: \mathtt{H[E]} \rightarrow \wp(\mathtt{H[A]})$$

its probabilistic behavior is defined by a function

$$\mathtt{D_F}: \mathtt{H[E]} \rightarrow \mathcal{M}(\mathtt{H[A]})$$

where for every input history $\mathtt{x} \in \mathtt{H[E]}$ by

$$\mathtt{D_F}(\mathtt{x})$$

we get a probability distribution for every input history $\mathtt{x} \in \mathtt{H[E]}$

$$\mu_{\mathtt{x}}: \mathtt{F}(\mathtt{x}) \rightarrow [0 :1]$$

We get a probability $\mu_{\mathtt{x}}(\mathtt{Y})$ by the function μ for every measurable set $Y \subseteq F(x)$ of output histories. This shows that μ defines a probability distribution μ_x for every input history $\mathtt{x} \in \mathtt{H[E]}$on the set F(x) of possible output histories.

4.4 Integration of Modeling Concepts

Cyber-Physical Systems need modeling concepts from following fields:

- Discrete system modeling
- Communication system modeling
- Control theory aiming at continuous system modeling
- Modeling human-machine interaction
- Dynamic and mobile systems
- Probabilistic systems

These modeling techniques have to be combined and integrated into a comprehensive system modeling approach. In addition, application domains have to be modeled as well as various specific system properties such as adaption systems and context awareness.

4.5 What and How to Model?

In systems engineering we basically aim at modeling a variety of system views and properties. We model behavior and we model non-behavioral properties such as structure and quality. In particular, we model specific aspects of systems such as data, using data models, processes, using process models, interaction, using interaction models and finally modeling the behavior of systems in interface models as well as their structure in architectural models and their state views in state transition models.

For software, which usually is written in some kind of programming languages, similar system models can be used, in principle. However, often software systems have more specific restricted concepts such as for instance the nowadays broadly used object-oriented programming languages. In contrast to the system model we have introduced, object-oriented programming concepts are closer to the von Neumann programming style. Object-oriented languages contain a number of concepts, which are related to modeling. Examples are inheritance and object creation, which reflect principles from modeling such as taxonomy and ontology, while object creation and the identification of dynamically created objects by identifiers is a step into modeling of concepts that we can observe in a number of real world systems where we can identify objects where over time objects may appear and disappear. Nevertheless, models for programming languages are much more specific than the system model we have introduced above. For programs logical techniques are used, in particular, based on assertions. A typical example for that is the contract concept proposed by Bertrand Meyer (see [Meyer 88], [Meyer 92]).

5 Conclusion

Cyber-Physical Systems open up the door for a whole world of new applications based on software. Today we see first examples of these applications in cars connected to the Internet, in smart phones using via apps several services including sensors and actuators. In the future we will see a large scale of highly connected Cyber-Physical Systems which are systems of systems containing special-purpose embedded systems connected to larger systems of coordination control and monitoring. Humans will be integrated into these systems by adaptive man-machine-interfaces finally leading to Human-centric Cyber-Physical Systems.

References

[ABB+09] Achatz, R., Beetz, K., Broy, M., Dämbkes, H., Damm, W., Grimm, K., Liggesmeyer, P.: Nationale Roadmap Embedded Systems. In: ZVEI (Zentralverband Elektrotechnik und Elektronikindustrie e. V.) Kompetenzzentrum Embedded Software & Systems, Frankfurt/Main (December 2009), https://www.zvei.org/fileadmin/user_upload/Forschung_Bildung/NRMES.pdf

[BGC+12] Broy, M., Geisberger, E., Cengarle, M.V., Keil, P., Niehaus, J., Thiel, C., Thönnißen-Fries, H.-J.: Cyber-Physical Systems: Innovationsmotor für Mobilität, Gesundheit, Energie und Produktion. Acatech BEZIEHT POSITION, vol. 8. Springer, Berlin (2012), http://www.springer.com/computer/book/978-3-642-27566-1

[Broy, Stølen 01] Broy, M., Stølen, K.: Specification and Development of Interactive Systems: Focus on Streams, Interfaces, and Refinement. Springer (2001)

[Broy 07] Broy, M.: Model-driven architecture-centric engineering of (embedded) software intensive systems: modeling theories and architectural milestones. Innovations Syst. Softw. Eng. 3(1), 75–102 (2007)

[Broy 12] Broy, M.: System Behaviour Models with Discrete and Dense TimeSamarjit Chakraborty. In: Eberspächer, J. (ed.) Advances in Real-Time Systems (to Georg Färber on the Occasion of his Appointment as Professor Emeritus at TU München After Leading the Lehrstuhl für Realzeit-Computersysteme for 34 Illustrious Years). Springer (2012)

[CPS08] Cyber-Physical Systems Summit "Holistic Approaches to Cyber-Physical Integration". Report, CPS Week (April 2008), `http://iccps2012.csewustl.edu/_doc/CPS_Summit_Report.pdf`

[GBC+12] Geisberger, E., Broy, M., Cengarle, M.V., Keil, P., Niehaus, J., Thiel, C., Thönnißen-Fries, H.-J.: agendaCPS: Integrierte Forschungsagenda Cyber-Physical Systems. Springer, Berlin (2012), `http://www.fortiss.org/fileadmin/user_upload/downloads/agendaCPS_Studie.pdf`

[IEC10] Functional safety of electrical/electronic/programmable electronic safety-related systems. Technical Report IEC/TR 61508 Part 0-7, International Electrotechnical Commission, IEC (2010)

[Lev95] Leveson, N.: Safeware: System Safety and Computers. Addison-Wesley (September 1995)

[Meyer 88] Meyer, B.: Object-Oriented Software Construction. Prentice Hall (1988)

[Meyer 92] Meyer, B.: Applying "Design by Contract ". Computer (IEEE) 25, 40–51 (1992)

[Neubeck 12] Neubeck, P.: A Probabilitistic Theory of Interactive Systems. PH. D. Dissertation, Technische Universität München, Fakultät für Informatik (2012)

[SC01] Sage, A., Cuppan, C.: On the Systems Engineering and Management of Systems of Systems and Federations of Systems. Information-Knowledge-Systems Management 2(4), 325–345 (2001)

Chapter 2
Game-Theoretical and Evolutionary Simulation: A Toolbox for Complex Enterprise Problems

Yves Caseau

Abstract. Complex systems resist analysis and require experimenting or simulation. Many enterprise settings, for instance with cases of competition in an open market or "co-opetition" with partners, are complex and difficult to analyze, especially to accurately figure the behaviors of other companies. This paper describes an approach towards modeling a system of actors which is well suited to enterprise strategic simulation. This approach is based upon game theory and machine learning, applied to the behavior of a set of competing actors. Our intent is not to use simulation as forecasting – which is out of reach precisely because of the complexity of these problems – but rather as a tool to develop skills through what is commonly referred as "serious games", in the tradition of military war-games. Our approach, dubbed GTES, is built upon the combination of three techniques: Monte-Carlo sampling, searching for equilibriums from game theory, and local search meta-heuristics for machine learning. We illustrate this approach with "Systemic Simulation of Smart Grids", as well as a few examples drawn for the mobile telecommunication industry.

Keywords: simulation, machine learning, game theory, evolutionary game theory, evolutionary algorithms, Monte-Carlo, dynamic games, serious games.

1 Introduction

Forecasting is an essential part of any enterprise's activities. It is the essence of budgeting, marketing, sales management or strategic planning. Although forecasting is notoriously difficult and turns out to be wrong most of the time, it remains a necessary management tool. It is used to define a path and to align the company resources along a common objective, even if this objective needs to be

Yves Caseau
Bouygues Telecom,
82 rue Henri Farman, Issy-les-Moulineaux
France
e-mail: yves@caseau.com

constantly re-evaluated to take unforeseen events into account. Models which are used implicitly or explicitly to perform those forecasts are simplified to fit existing tools. Further-developed models would be necessary to capture interactions between competing companies. The most commonly-used tool is the spreadsheet. Spreadsheets make it difficult to capture two common traits of complex models: uncertainty and feedback loops. Hence the usual approach is to identify loops and uncertainty, make a few hypotheses to get rid of them, build a simpler deterministic model and rapidly forget about the simplifying assumptions.

Our proposed contribution is GTES, a tool to "play with" a model that is burdened with too much indetermination. This means learning a few insights from a repeated and organized set of simulations, mostly through exchanges and collaboration. GTES stands from *game-theoretical and evolutionary simulation* [1] and will be presented as a hybrid extension of evolutionary game theory [2]. This approach is based upon two key insights:

1. A number of classical techniques may be combined to get rid of a significant part of indetermination, such as sampling and machine learning.
2. Such "weak" models should be used to "play", not to forecast. A global-scale model with too many unknown parameters is not useful to make a decision, but rather to help building a keener intuition about the situation at hand, through the repeated practice of scenarios.

This paper is organized as follows. Section 2 provides more detailed motivations for this work. We explain the limits of current forecasting methods and show why they need to be extended with a different approach. We present a few concrete examples which advocate for building a "model workbench", that is, a tool to experiment with complex models of competition. For instance, we propose a "systemic model" for smart grids, where the "complex system" nature of electricity networks is made explicit, through multiple actors with conflicting goals.

Section 3 describes our proposed GTES approach. Formally, we need to solve a repeated non-cooperative game with uncertainty. The key idea is to separate the unknown parameters, from the model that is being studied, into three groups: the parameters that describe the goals of the actors – parameters that define the objective function, those parameters which are bound to an actor but dominated by the strategic choice expressed by the first group, and a third group of parameters which are independent from any actor – they describe the market in which the actors operate. GTES deals with each group of parameters differently: the first group is the control group, which defines a repeated game for which we look for Nash equilibriums [3]. The second group is managed with machine learning (the algorithm looks for parameter values that maximize the objective function) and the third group is managed with Monte-Carlo sampling [4].

Section 4 shows the application of GTES to our "Smart Grid System", as well as a few other models taken from the author's experience in the Telecom industry. Section 5 discusses different perspectives about GTES, from stability and performance point of views.

2 Motivations

2.1 Complex Models and Enterprise Simulation

As stated in the introduction, the ubiquitous use of spreadsheets has a profound impact on the way people see and think about their environment. Although many sophisticated tools have been designed for decision aid – from operations research to statistical and datamining tools –, spreadsheets still represent the vast majority of the modeling effort that takes place in an enterprise. It is not because that the vast majority of business activity is simple, it is because the simplicity of the underlying model is crucial to the appropriation that is required in most business situations. Our dual experience – ten years as an operation research scientist and ten years as an executive manager – has convinced us that a great decision tool that relies on a sophisticated model faces a lot of skepticism and distrust from business managers. Hence uncertainty is tamed with scenarios and ranges of values (proverbial *min* and *max* values), although scientific evidence shows that uncertainty about extreme values is even higher. The complex feedback loop that describe the fierce competition in an open market is reduced to making a few assumptions about "what our competitors will do" – most often, the same as they did before – which are treated as input parameters for the spreadsheet model. The result is a practice that is well suited to "complicated situations" (spreadsheets with many sheets and huge amount of data) but not to complex situations.

To take the complexity of a business situation into account, one would, on the contrary, expect to develop the following traits:

- Take into account the interaction loops between the company and its various competitors. This is one of the most obvious sources of complexity, but there usually are other feedback looks as we shall see later on.
- Keep within the model what is understood qualitatively, even if the quantitative formulas are unknown.
- Refrain from abusing linear formulas for elasticity or other forms of price sensitivity. Extrapolating trends that were seen in the past few years, which is quite easy with a spreadsheet, is another way of oversimplifying a business situation.

The intent here is nicely summarized by Carveth Read's famous quote, often misattributed to Keynes: "*It is better to be vaguely right than exactly wrong*". The work that is presented in this paper was born twelve years ago, while trying to evaluate the best strategies during the UMTS license bids. The dominant thinking at the time was that bidding for a new license was unavoidable because of the promises of 3G revenues. All market analysts had come up with similar spreadsheets models that described a truly significant increase of value creation. However, these models shared two short-comings: they did not factor the necessary price war that occurs when new actors enter a fixed-cost activity and need to gain market share, and the competition between the different motives for spending money in European households was not taken into serious consideration, although a wealth of accumulated knowledge existed. A first, simplified, game-theoretical simulation model was built to look into the dynamics of competition with a potential new entrant. With hindsight, these first simple simulations were quite relevant and provided a useful, though modest, contribution to the decision process.

2.2 Systemic Simulation of Smart Grids

Smart Grids are receiving a lot of attention from many scientific communities. The need for smart grids is emerging from the multiplication of intermittent, decentralized energy sources, such as wind or solar plants. The possibility of smart grids comes from the combination of price de-regulation and the lower cost of IT, which makes reaching dynamic demand-response equilibriums possible through price signaling. The necessity of smart grids, as explained by its advocates, is the urgency of global warming and depletion of fossil energy sources, which requires a behavioral change as far as energy consumption is required. Smart grids (with their components, such as smart cities, smart neighborhoods and smart homes) are meant to provide consumers with dynamic and constant feedback. One of the promises of smart grids is to optimize the amount of “shaved” energy through the right price incentive - “shaving” is the reduction of peak consumption when the price gets too high (demand-response adjustment).

The idea to build a “serious game” model for smart grids came from the following list of questions. Without being an expert on energy production or distribution, there are a few puzzling reasons to be doubtful about “distributed smart grids”. Centralization dampens part of the variation through aggregation (the “independent” variation, according to the “law of large numbers”). It also leverages the “economy of scale”. Large-scale energy plants are more efficient (mostly because of the cost advantage of nuclear power plants), even when transportation costs are factored in. On the other hand, there are a few factors that are favorable to local operators (as opposed to national suppliers):

- Distribution means that production & pricing policies are better suited to each city.
- A distributed approach yields a “system of system” (network of smart grids), which may prove to be more resilient and better able to manage peak situations.
- CO2 taxes will encourage renewable energies which are easier to manage in a distributed way. A key question is the availability of storage at a reasonable price, since most renewable energies are intermittent.

- Feedbacks works better at a "community level", which means that dynamic pricing and the "demand-response" behavior that produces shaving is likely to be more effective if managed locally. This is a key principle for smart grids in Japan and South Korea: focusing on communities and behavioral changes.

To get a better understanding of this complex system and its associated issues, we decided to build a simple model (S3G: *Systemic Simulation of Smart Grids*) with the four kinds of players:

- The "**regulator**" (political power) whose goal is to reduce CO2 emissions while preserving economic output and keeping a balanced budget (between taxes and incentives). Its tactical play includes setting up a CO2 tax, regulating the wholesale price for the suppliers and creating a discount incentive for renewable energies.
- The existing energy companies, here called "**suppliers**", whose goal is to maintain their market-share against newcomers, maintain EBITDA (revenue) and reduce exposure to consumption peaks. Their tactical play is mostly through pricing (dynamic), but they also control investment into new production facilities on a yearly basis.
- The new local energy **operators**, who see "smart grids" as a differentiating technology to compete against incumbents. Their goal is to grow turnover, EBITDA and market-share. Their real-time tactical play is dynamic pricing, and they may invest into renewable and fossil energy production units, as well as storage units. Another tactical choice is how to best use storage capacity, both as a "buffer" (for their own production) and as a "reserve" (buy energy when it is cheap and resell it when it is more expensive).
- The consumers are grouped into **cities**, whose goal is to procure electricity at the lowest average price, while avoiding peak prices and preserving their comfort. The cities' tactical play is mostly to switch its energy supplier (on a yearly basis) and to invest into "negaWatts", which are energy saving investments (more energy-efficient homes, etc.).

The objective of S3G is to simulate the production and consumption of electricity throughout a long period of time (15 years). The following figure is an overview of the S3G model. The set of equations that describes each player's behavior does not fit into this paper, but may be summarized with four parts:

- Energy demand: for each city, energy demand is generated from an hour-by-hour and day-by-day template, adding some random variation (the extent of which is a model parameter) together with a city-specific variation. This number is then reduced by the amount of "negaWatts", computed from the total amount invested by the city. The model uses a ratio obtained from a concave-increasing function of the investment.
- Dynamic Pricing: both suppliers and operators use a simple affine pricing model, with a constant price when the demand is less than a "base power", and a linear formula when the demand is higher.

- Production: suppliers use nuclear power according to planned schedule and adjust to resulting demand with fossil plants. Operators always use their green power (store it in the “buffer” or resell it when there is too much of it). They adjust to the city demand with their own fossil plant and wholesale electricity from suppliers, at the lowest marginal cost.
- Consumption: The actual electricity consumption for each city is the demand, minus “shaving”, which is obtained by applying an S-curve to the sale price.
- Market-share: for each city, the market balance between the national supplier and the local operator is determined yearly using another S-curve.

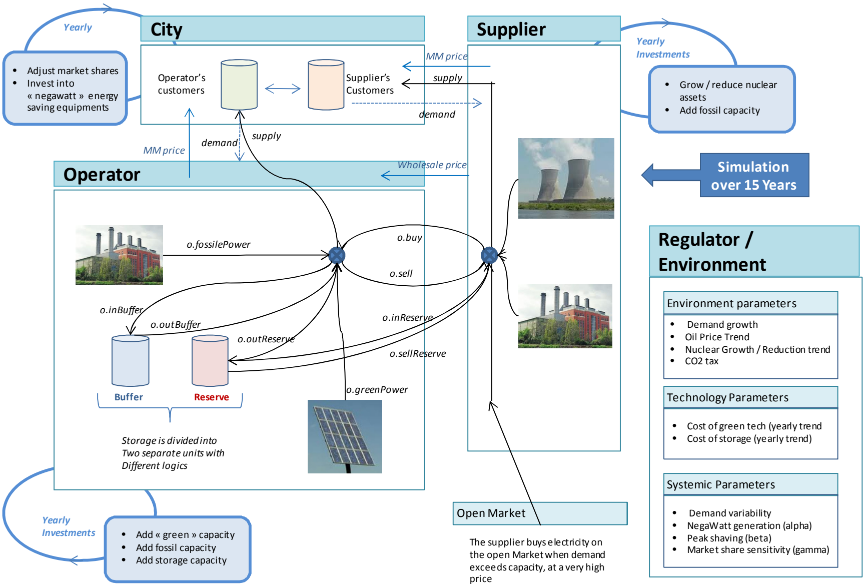


Fig. 1 Systemic Simulation of Smart Grids

This S3G model is both simple and complex. It is simple because it is based on a handful of equations, resulting in a simulation code that is 500 lines long. On the other hand, it is a complex model for two reasons. On the one hand, there are multiple feedback and interaction loops that make it difficult to analyze how the system will react to perturbations. On the other hand, there are many unknown parameters in this model (such as market sensitivity, demand-response behavior, negaWatt capabilities, etc.).

2.3 *Cellular Game Simulation*

Our second example (CGS: *Cellular Game Simulation*) was co-developed with GTES over the last ten years. Its ancestor is the simple “competition game” that was mentioned in Section 2.1. The heart of the CGS model is a simplified vision of how an operator works, based on a few public figures that may be found in yearly financial reports:

- Turnover (sales), EBITDA (*Earning Before Interest, Taxes, Depreciation and Amortization),* ARPU *(Average Revenue Per User),*
- Number of customers (size of its "base"), yearly acquisitions (addition to the base), "churn" (removal from the base),
- Acquisition and renewal costs, interconnection costs (the other costs are labeled as "operational expenses").

It is indeed truly simplified since all kind of lines of business (corporate vs. mass-market, MVNO, voice vs. data, prepaid vs. post-paid) are aggregated. This yields the concept of an "average telecom package", sold for an "average price". Operational expenses are supposed to be constant, or to vary according to a yearly trend. This simple model is interesting in itself because of its simplicity and the availability of indicators for competing operators.

However, the true interest of such a simple model is that it is possible to instantiate the three or four operators of the French market and see how they interact. The major tool that is used to implement this coupling is the S-curve function that describes market-share distribution according to prices. Each operator has three "control parameters" that describe its strategy on the open market: what is the price of the "average package", how much is spent on renewal and on acquisition. A "tactical play" can be seen, as a first approximation, as the values of these parameters for the three coming years (what is commonly referred to as a 3YP – three year plan).

The concept of an "average telecommunication package" supports some form of elasticity, which has varied over the years and will continue to change as we enter a world of "access price with unlimited use". Ten years ago the consumption unit was the minute of voice, and there was a fair level of elasticity. Today, the "average package" aggregates voices and data, from plans which are both unlimited (fixed-price) and usage-based. Hence the parameter that represents the price-to-volume elasticity in our model is both approximate and uncertain.

The overall CGS model is described by Figure 2. It may be decomposed into five steps, which are run consecutively for each year of the simulation (i.e., 3YP requires 15 steps). The **first step** is to compute the value of churn (the flow of customers who leave their current operator) as well as the flow of renewals (customers who obtain a new handset from their operator). In both case, the flow value is derived by applying the S-curve to the price variation. The **second step** distributes the combined flow of churn customers and new customers (market growth) into distribution channels. Based on the three-year plan (from which new prices are derived), the S-curve function produces the new market shares (for each channel). The **third step** is quite similar since it distributes the flow of customers assigned to each channel to all operators, using the same mechanism. The **fourth step** applies the previously mentioned price-to-volume elasticity to generate the turnover obtained by each operator. The **fifth step** computes the revenue by subtracting the costs.

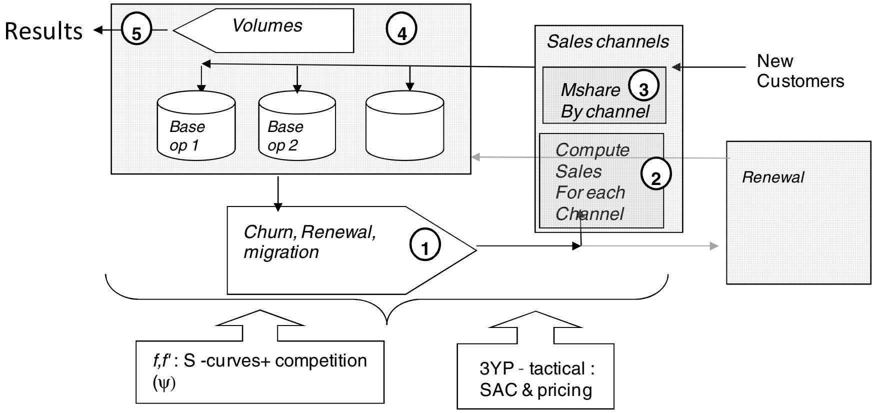

Results
5
Volumes
4
Base op 1
Base op 2
Sales channels
Mshare By channel
3
New Customers
Compute Sales For each Channel
2
Renewal
Churn, Renewal, migration
1
f,f': S -curves+ competition (ψ)
3YP - tactical : SAC & pricing

Fig. 2 CGS simulation architecture

There are a number of common traits to these types of models: S3G, CGS and the other telecom models which are mentioned in section 4.3. They are all "simulation model" that produce financial results (costs and revenue). They are "generic" simple models, but with a lot of "indetermination" (many of the key parameters in the parametric equations are unknown - or at least not known precisely). In each case we may separate all these unknown variables into three groups:

- Those which are independent from the players and simply reflect some uncertainty about the underlying economic model (for instance, the parameters for the S-curve function).
- Those which represent the strategy of the players, that is, what needs to be optimized.
- Last, the variables that are associated to each player, which set the behavior of the payers (such as the three "tactical" levers that we mentioned for the CGS model).

3 GTES: Game Theoretical and Evolutionary Simulation

3.1 Principles

GTES is a framework designed to study a model through simulation, in order to extract a few properties from this model (learning through examples), either explicitly or implicitly. The input material is a set of parametric equations, with a number of unknown parameters, which represent the behavior of a set of interacting actors/players. The result is a set of computational experiments, from which some information may be extracted.

GTES is based upon the combination of three techniques:

- Sampling: since the value of the parameters that occur in the economic equations is unknown, we draw them randomly from a confidence interval, using a Monte-Carlo approach. This will generate a large number of simulation runs, from which we look to extract common characteristics.
- Search for Nash Equilibrium in a repeated game [5]: We set the parameters that define the player's objective functions and look for an equilibrium using an iterative fixed-point approach (in the tradition of the Cournot Adjustment [6]).
- Local Search as a machine learning technique to solve the sub-problem of tactical optimization. Once the parameters that define the objective function are set (what we call a "strategy" from a business perspective), the other parameters that define the behavior of each player may be computed to find each player's "best response" to the current situation. This search is performed using classical local optimization techniques [7].

This approach sits at the intersection of « classical » lines of work such as economic modeling, game theory and evolutionary game theory [8] [9] [10] [11], and local search (genetic algorithms, simulated annealing, stochastic and Tabu search [12] [13] [14], etc.). We depart from the classical approach of evolutionary game theory since only one agent is introduced for each player, and genetic algorithms is only one of many optimization techniques that we use to compute each player's response. On the other hand, GTES is clearly an offspring of evolutionary game theory [2], since we look for equilibriums that are the result of evolutionary processes. We care about the path to get to the equilibrium as much as the equilibrium itself.

This paper has been strongly influenced by works from R. Axelrod. In his book [15], R. Axelrod uses genetic algorithms to find the best strategy to play a repeated version of the "prisoner dilemma". Very interestingly, he compares an experimental protocol and computer simulations (where the combination between many initial strategies is optimized with genetic algorithms). Both approaches reach a similar conclusion, that TIT-for-TAT is the most robust strategy for this game. More generally, this work also belongs to the field of economic modeling and simulation with multi-agent systems, using evolutionary algorithms [16] [17]. For instance, chapter 7 of R. Nelson and S. Winter's book show an example of the kind of model that we try to study here, which the authors propose to simulate with Markov Chains. Using machine learning is a common thread in Evolutionary Games [6] [18] [19].

The different application examples that were developed using GTES are quite similar to simulations that have been made in the past following the "System Dynamics" framework [20] [21]. System Dynamics has a rich history throughout many decades, and has been applied successfully to corporate strategy and market analysis. The heart of this approach is to describe a model as a network of relationships between a few key variables from a system. The main feature is the "polarity" associated to each link: positive means that an increase of the originating parameter causes an increase in the destination parameter; negative means that an increase of the first causes a decrease of the second. Our experience is that this network is relatively easy to identify, whereas the actual equations that express the relationships between variables require more time to fine-tune. Because GTES uses local search and randomization, the underlying mathematical model is "shaken" during the simulation (cf. Section 5), and model tuning is much more demanding than it would be for producing a simpler simulation.

3.2 GTES Model

We shall now describe GTES with some details in the remainder of this section (a more formal description may be found in [1]). From now on, we will consider that the input model may be seen as a parametric function f_p which computes the satisfaction of each actor. We suppose that the computation of f_p entails the computation of all the local variables that describe each actor's state. In the remainder of the paper f_p^i represents the i-th component of the result tuple, that is, the satisfaction of the i-th actor.

The parameter p represents the "strategy" of the model (not in the sense of game theory), that is how we evaluate the satisfaction of each player. Thus we will be able to compare the properties of the two parametric games f_p and $f_{p'}$, that is, with two different ways to evaluate success for each player.

The arguments of f_p are the model's parameters, which we represent by two variables x and e, according to the distinction made previously: x is a vector of numbers which represents the tactics of each player and e is a vector of unknown econometric parameters. If the tactic of an actor (what we called a "tactical play" in Section 2) may represented with m numbers and if there are n actors, x is a vector of size n x m, and x_i will represent the tactic of the i-th player. We call X the set of all possible tactic vectors, and X_i the set of all possible tactics for the i-th player ($X = X_1 \times .. \times X_m$). To facilitate reading, we shall use the letter x to represent vectors of (n) tactics and the letter t (or x_i) to represent one tactic – which is also a vector – for instance we may write $x = (t, t', \ldots) = (x_i, x_{-i})$. We borrow the convenient notation x_{-i} from game theory [6].

Given the values of x and e, $f_p(x,e)$ represents the computation of the satisfaction of all players. This says that once a value is given for each unknown parameter, the set of equations that make the model may be resolved to find each actor's behavior and, therefore, satisfaction. We call E the Cartesian product of each possible range for the parameters that are associated with e. Hence we can state the problem that we want to resolve as:

$$\max_{x \in X} f_p(x,e), \qquad e \in E \tag{1}$$

Since f_p is a tuple-valued function, its maximization is a "multi-objective" maximization problem which fits precisely the framework of game theory.

3.3 Equilibriums and Neighborhoods

If we assume that all parameters are known but for x_i, this turns into a classical optimization problem (single objective function f_p^i) and we may define the "best response" (BR in the tradition of game theory [6]) as a tactic (not necessarily unique) that maximizes the satisfaction of the i-th player:

$$\max_{x_i \in X_i} f_p{}^i(x_i, x_{-i}, e), \qquad e \in E$$

BR(*i,x*) is a solution to this maximization problem, which may be solved using traditional techniques from operations research (depending on the complexity of the underlying model f_p). In this section, *e* is assumed to be chosen and constant, so we will drop it from the formulas.

The most natural approach with our problem (1) is to search for is a Nash equilibrium [3], that is a tactic vector *x* such that:

$$\forall i, \forall t \in X_i, f^i(t, x_{-i}) \leq f^i(x_i, x_{-i})$$

There does not always exist a Nash equilibrium in a game with deterministic tactics. In a reciprocate way, there may exist more than one equilibrium. We will look for an equilibrium that is characterized as a fixed-point of a sequence of "best response" moves (where each player successively replaces its tactic with BR(i,t), what is called a *Cournot Adjustment*).

In the tradition of evolutionary game theory [2], we look not only for plausible equilibriums but also for plausible trajectories. If the set of tactics that represent the equilibrium requires a very complex mathematical computation to find it, one may wonder if it will be reached by real players. Hence we introduce a neighborhood structure on the set of tactics, similar to the neighborhood structure that is used with local optimization algorithms [7]. A neighborhood *V* associates to each tactic vector *x* a set of "near" tactics *V*(*x*). For instance, a k-neighborhood associates to *x* all the tactics that may be obtained by changing *k* parameters only.

We can restrict the concept of "best response" to the search of tactics from the existing neighborhood:

$$BR_V(i, x) := t \text{ } such \text{ } that \text{ } f^i(t, x_{-i}) = \max_{y \in V(x_i)} f^i(y, x_{-i})$$

Local optimization is defined as the exploration of the transitive closure of the neighborhood structure: V*(*x*) is the set tactics *t* that be found as the end of a path ($x, x_1, \ldots, t$) where each tactic of the path belong to the neighborhood of its predecessor, according to a given local search heuristic. For instance, the simplest search heuristic is Hill-Climbing, where we only look for increasing chains. We extend the previous definition to BR_{V*}(i,x) which is the best response in the extended neighborhood V*(x_i). A local optimization method is complete when BR_{V*} = BR.

This structure defines the concept of a pseudo-Nash equilibrium, when each player's tactic is optimal modulo the set of alternate tactic from that neighborhood:

$$\forall i, \forall t \in V^*(X_i), f^i(t, x_{-i}) \leq f^i(x_i, x_{-i})$$

One may see this as the application of the "bounded rationality" concept: from a given position, each actor is not able to contemplate all possible tactics, but only a smaller subset which may be characterized through an operational process. Obviously, the choice of the neighborhood structure becomes a key feature of the actor model. The larger the neighborhoods are, the "smarter" the actors are supposed to be. The search for a pseudo-Nash equilibrium is an iterative process that looks for a fixed-point. We apply BR_{V*} to each actor successively.

More precisely, we may define the iterative process as follows:

- Select each player in turn with their current tactic t = x_i,
- Apply a local search algorithm to compute t' = $BR_{V*}(i,x)$
- Replace t by t' in the tactic vector x,
- Measure the convergence (see later),
- Exit after a given time-out (or a given number of iteration *M*), or repeat.

Iterative search does not necessarily converge. When it does, we obtain a pseudo-Nash equilibrium by construction, and we qualify the GTES simulation as stable. If it does not, we distinguish between two cases: either there is a divergence pattern, where each actor become less satisfied after each iteration, which we qualify as a "war", otherwise we cannot say anything and we quality the simulation as "chaos".

The following figure shows two examples of GTES simulations, which we call trajectories. The first one exhibits a "chaos" behavior, while the second one shows a "stable trajectory" with the convergence towards a pseudo-Nash equilibrium.

We use three families of metrics to evaluate convergence:

- We perform a linear regression on the total satisfaction of all actors.
- We define a Euclidian distance and its associate norm on tactic vectors, to measure the distance between two tactics during the adjustment.
- We also define a Euclidian distance over a vector of internal state variables, to ensure that the states of actors converge when the tactics do.

We cut the trajectory into two equal pieces and evaluate its final status as follows:

- If the deviation of the linear regression is less than 5% and if the resulting slope is less than 1%, the trajectory is called "stable" [6].
- If the deviation is less than 10% and if the slope is less than -2%, the trajectory's status is called "war"
- Otherwise, we set the status to "chaos".

In the example from Figure 3, the metric that is used is the sum of the economic results (EBITDA) for all players. Hence the result of one iterative search is a triplet: *status, typical values, convergence measure.* The "typical values" come from a few selected state variables associated to the actors. In the CGS model, we use the EBIDTA, the market share, the total number of customers, the ARPU (average revenue per user). Obviously the most important of these values is the actors's "satisfaction" function $f_p^i(x^*)$. These typical values give a sense of what "the situation is like" when the equilibrium is reached. The convergence measures are drawn from the three previously mentioned metrics to qualify the degree of convergence.

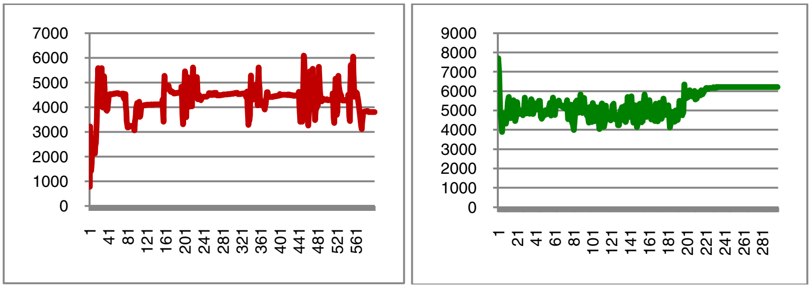


Fig. 3 Two trajectories for the search of an equilibrium (total satisfaction showed)

3.4 Monte-Carlo Sampling

We have described how a GTES trajectory is computed for a given parameter vector *e* (by convention we use Greek letters to designate such parameters). Providing *e* is performed through sampling [22]; that is, randomly picking a value for each "economic parameter" from a confidence interval. There are typically between 5 to 10 such parameters in most models we have applied GTES to. Since nothing is known about a probability distribution, each random picking follows a uniform distribution.

Here are some examples from the CGS model (Section 2.3):

- α is the price-to-volume sensitivity (the slope of an S-curve, cf. 4th step).
- β is slope of the S-curve that is used to derive renewal figures from price variations (slope of the *f* function = derivative at the 0 value).
- γ is the slope of the S-curve that is used to derive sales figures from price variations.
- δ is the number of month that the average customer uses when evaluating the TCO (*total cost of ownership*) of a phone package when comparing two operators (how to balance between the handset price and the monthly fee).

A complete GTES simulation (for a given value of the strategy parameter *p* which "defines the game") is defined as the repetition of *N* identical steps: sampling to build the *e* vector, followed by the iterative search of an equilibrium as explained in the previous section. Each trajectory's result is aggregated into an "overall" result triple:

- Status distribution: % of trajectories that are respectively stable, wars and chaos.
- Average/deviation for "typical values" (satisfaction, EBITDA, …)
- Average/deviation for the convergence metrics.

A key issue with Monte-Carlo sampling is the number (*N*) of drawings necessary to get statistically significant results. We measure the standard deviation of all the result components. It is easy to observe the increase stability of results and their interpretation when N grows. For most problems, we get interesting results with only a few hundred sampling trajectories, although a few thousand seems the ideal trade-off. Going to higher value of N only yields very small improvements, which are not relevant for such approximate models, nor for the way GTES is used (serious gaming).

3.5 Search for Equilibriums

We shall conclude this section with different techniques that may be applied to improve the search for equilibrium, namely: **interweaving** (of the two optimization loops presented in Section 3.3), **parallel** application of BR moves and "*minmax*" evaluation, yielding the concept of a "**Forward Nash Equilibrium**".

The idea behind **interweaving** comes from the realization that GTES uses two embedded loops that search for a fixed point: the search for a local optimum for the tactic of a given player (computing BR_{V*}) and the search for a pseudo-Nash equilibrium, with at most *M* iterations. Hence we may decide to approximate the search for the optimal tactic with BR_{V_q} which is the best tactic found in the V^q neighborhood, which is the set of tactics that may be found with at most q moves from V. By construction $V^* = \lim_{n\to\infty} V^n$ (the limit exists because V^n is an increasing sequence).

Interweaving means computing t' = BR_{V_q}(i,x) in the sequence described in Section 3.3, and expecting the outside loop to progressively approximate best responses from V*. There is a trade-off: if we select a small value of q, it will take many outside loops to reconstruct a longer chain and there is no guarantee that the interplay between the actors will not prevent from finding these longer optimization chains. However, the inner loop (exploring V^q for each actor) will run faster so we can run more occurrences of the outside loop (moving from one actor to another). Computational experiments could not be reproduced here for lack of space, but they show that a "moderate" form (q ranging from 5 to 10) of interweaving produces an improvement (faster convergence for the same run-time).

The choice of searching the best response of each actor in turn, sequentially, may seem surprising (this is the "alternate-move" versus "simultaneous-move" approach [6]). A more natural approach would be to apply the search of the best response in **parallel**: exploring BR_{V_q}(i,x) for all i and from the same x simultaneously. When then apply the transformation simultaneously to each actor.

Hence one step of the algorithm may be described as changing the vector of tactics x as follows:

$$x \rightarrow (BR_{Vq}(1,x), BR_{Vq}(2,x), \ldots, BR_{Vq}(n,x))$$

instead of the sequential transformation:

$$x \rightarrow (BR_{Vq}(1,x), x_{-1}), x \rightarrow (BR_{Vq}(2,x), x_{-2}), \ldots, x \rightarrow (BR_{Vq}(n,x), x_{-n})$$

For lack of space we do not include the result of comparing the two approaches. The short summary is that they are very similar. A more detailed analysis shows that the parallel approach converges faster in the early cycles but slower in the last runs. For the practical application to the CGS problem, we found that there was no sufficient difference between the two options.

Many of the situations without Nash equilibriums are resolved in the real world precisely though a *maxmin* evaluation, that is, when each player considers the possible reactions of other players to her/his moves [23]. This consideration may be the fruit of experience in a repeated game or the result of a thought experiment (e.g. chess players). We can introduce this “*maxmin*” evaluation in our GTES model as follows. First we define a new valuation function for tactics:

$$f_i^*(t_i, t_{-i}) = \min_{j \neq i}(f_i(t_i, BR_V(j), t_{-i-j})$$

This means that the evaluation of a given tactic t_i for an actor i is obtained by considering sequentially all possible “best responses” from other actors, when “best response” means to explore the neighborhood of t_j according to V . With this new valuation, we can extend the concept of a Nash equilibrium to what we will call a *Forward Nash Equilibrium (FNE)*:

$$\forall i, \forall x \in T_i, f_i^*(x,t) \leq f_i^*(t_i, t_{-i})$$

To extend GTES to search for FNE, we need to extend the search of a local improvement move through the neighborhood structure V. This requires to replace the inner local optimization loop that computes $BR_V(i,x)$ by a double loop:

- The first loop explores all possible moves from *V(t_i)* – as previously (Section 3.3) – that is, consider all tactics *x* that belong to *V(t_i).*
- For each possible new tactic from this neighborhood, we look for the best response for all other players. This means that we recursively enumerate, for all *j* different from *i*, the tactics that are reachable in *V(t_j).* We look for the tactic that maximizes the satisfaction of the actor *j* and record the satisfaction of the first actor (*i*).

The evaluation of a tactic vector *x* is the *min* of all recorded value (the worst possible response from one of the other player). The goal of the first loop is to maximize the minimum satisfaction (hence the “*maxmin*”).

We have applied the search for FNE in the CGS example (cf. Section 4.2) because the competitive nature of the business problem yields many “non-stable” situations, with no Nash equilibriums. We applied this FNE approach with only 50 iterations. When we ran the FNE algorithms over a large number of experiments, we obtained a clear improvement in stability (i.e., percentage of “stable” trajectories as defined in Section 3.3), and the behavior of the players became “less aggressive”, resulting in an overall higher satisfaction. There is, however, a significant problem with this approach, namely the computational time. It takes two hours on a fast PC to run the 50 iterations of the FN search loop, as opposed to a few minutes required for the 400 iterations of the BR loop. We will return to the performance issue in Section 5.

4 Applications

4.1 Computational Experiments with Smart Grids

GTES is a framework, which has been implemented as a library. The part that simulates each situation is specific to each problem, but the control algorithms that implement learning (local optimization) and randomization (Monte-Carlo) are generic. The S3G model is implemented with three files: a data model which describes the structure from Figure 1, a simulation file that implements the behavior of the different players as summarized in section 2.2, and a “control” file that contains the generic GTES methods. The following figure shows, on the right part, a rough summary of the simulation loop that is run for each time period (3 hours, hence 8 times per day). On the left part, it shows the three main GTES generic procedures that were introduced in Section 3 [1].

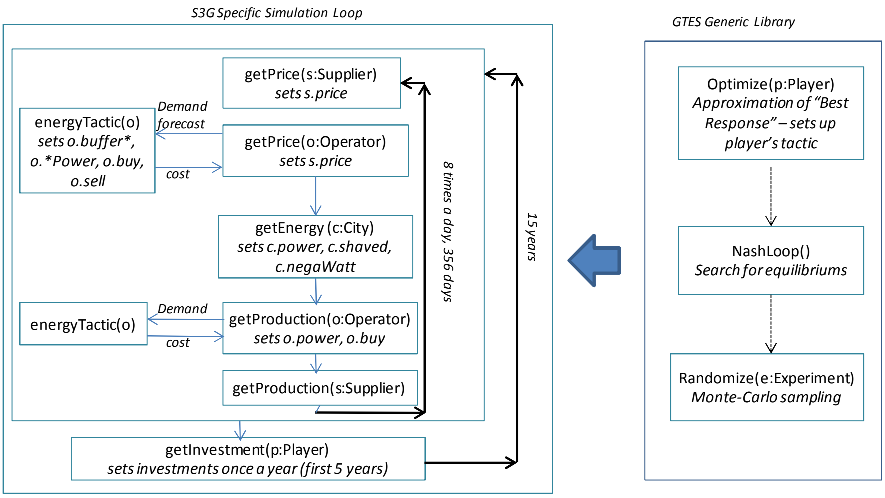


Fig. 4 S3G GTES Architecture

A “serious game” session is made of interactive runs of “experiments”, which are GTES computational executions. More precisely, an experiment is defined through two things:

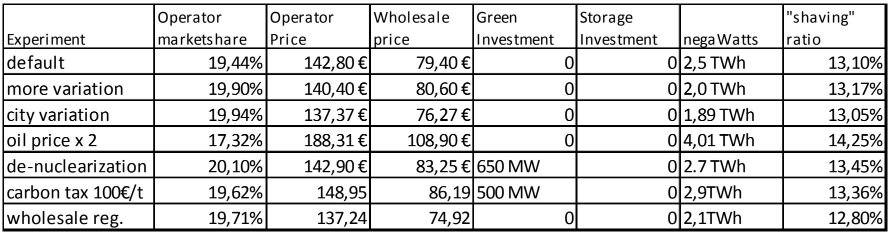
- The randomization boundaries, for those parameters that will be sampled using Monte-Carlo technique (Section 3.4).
- Some specific values for some parameters, since the goal of a "serious game" is to play "what-if scenarios", by explicitly changing these parameters. For instance, we may play with the investment cost of storage, to see if storage is or will be critical to smart grids

A GTES simulation run returns the average and standard deviation of a few key business parameters, as well as some indication of the Nash convergence. In the S3G instance, finding a pseudo-Nash equilibrium is simple and all trajectories are stable. Giving averages and a few deviations is a poor restitution of the rich data gathered during the computational experiment, but the goal is simply to "get a feeling for what is happening", as opposed to producing a forecast. The following table shows some results obtained with a list of experiments designed to understand the main issues that were exposed in Section 2.2. This is a simple experiment, with a fictional country somehow similar to France decomposed into 10 regions/cities. We only picked 7 resulting parameters, whereas a typical output is between 20 and 50 pairs (average/variation). This table shows 8 experiments that may be defined as follows:

- The "default" is a reference point, from which "what-if" sensitivity analysis is made. The economic parameters are set in such a way that alternate operators start with a 20% market-share and should be able to increase it if they demonstrate a better management of variability.
- The second experience raises the variability of energy consumption (globally), while the third experience raises the local variability (each city is more different from each other)
- The fourth experiment doubles the fossil energy price (gas and coal). In the default scenario, it is randomly drawn between 20€ and 40€/MWh.
- The fifth experiment imposes a 5% reduction of the nuclear assets for the supplier during the first 5 years.
- The sixth experiment sets a carbon tax at 100€/t, the proceeds of which is used by the "regulator" to subsidize green energy investment.
- The last experiment is a small variation of the first one, where wholesale prices are more rigidly constrained.

GTES is a "serious gaming" framework, whose value is *implicit learning* while running multiple experiments, and playing "what-if" scenario. Hence it would be illogical to see the previous table as a computational results from which conclusions may be drawn. Furthermore, this table is not really meaningful without the full list of hypothesis that is part of the randomization/systemic parameter settings. The expected use of such results is to show them to a domain expert who will instantly criticize some of the figures (there is not enough "negaWatt", your fossil price is too low, etc.), propose an alternate value … and the "serious game" begins ! Most of the time, the benefit of a "serious game" session is to help oneself understand a few things that are "obvious" in hindsight, but not so much when you start.

Table 1 A few experiments with Systemic Simulation of Smart Grids



Experiment	Operator marketshare	Operator Price	Wholesale price	Green Investment	Storage Investment	negaWatts	"shaving" ratio
default	19,44%	142,80 €	79,40 €	0	0	2,5 TWh	13,10%
more variation	19,90%	140,40 €	80,60 €	0	0	2,0 TWh	13,17%
city variation	19,94%	137,37 €	76,27 €	0	0	1,89 TWh	13,05%
oil price x 2	17,32%	188,31 €	108,90 €	0	0	4,01 TWh	14,25%
de-nuclearization	20,10%	142,90 €	83,25 €	650 MW	0	2.7 TWh	13,45%
carbon tax 100€/t	19,62%	148,95	86,19	500 MW	0	2,9TWh	13,36%
wholesale reg.	19,71%	137,24	74,92	0	0	2,1TWh	12,80%

This being said, here are a few findings that may be drawn from the hundreds of runs made with the S3G model, and that are worth sharing:

- There is a systemic benefit of distribution and autonomy to cope with variation. This is shown in our second and third line. It is a “subtle” variation (small effects), which means that the economy of the local operator is dominated by its capacity to operate at much lower customer management costs than the supplier.
- CO2 tax increases play a very small role, and one that is difficult to anticipate since it both favors the local operator (support green subsidies) and the supplier (raises the difference between fossil and nuclear).
- “De-nuclearization” is a favorable scenario for smart grid operators, as are most regulations that are adverse to the supplier. The obvious limitation is the resulting price increase that reduces the total economy output (and the country’s competitiveness).
- The “community advantage” (that is, the ability for a local operator to better manage the demand-response loop because it is “closer” to its end customer) is marginal, and it is quite unclear if the payback from demand-response management is enough to sustain the operator’s business model.
- Investing in local storage is never an interesting option (at current prices). We needed to slash the price by over an order of magnitude to see a viable payback in less than 10 years.
- There is a clear competition between local operators and suppliers. The learning component of GTES makes for “agile” players who react closely to each other signals. The pricing structure plays an important role (we have only explored a simple variable pricing scheme). A logical consequence is the importance of regulation.
- The results are sensitive to the strategies of the player. A next step for S3G is to build a “strategy matrix” similar to the one shown in the following section. A strategy matrix is a tabular “what-if” sensitivity analysis where we see what happens if the goals of the players (cf. Section 2.2) are changed.

4.2 Various CGS Games

The following figure is drawn from a previous article where GTES was applied to CGS with three players [1]. The *x* axis shows different experiments (E1 to E6), where the parameters define the objectives for these players. Note that the concept of an "experiment" is related to the set of parameter *p* from the formal model in Section 3.2. E1, E3 and E5 represent similar situations but the goals are more and more aggressive. This means that the target figures for EBITDA gets higher from E3 to E1 and E5 to E3. As a matter of fact, for CGS we state the objective as the combination of three goals which are all expressed as a yearly growth rate: EBITDA, market share and sales (turnover). Each even strategy (E2, E4, E6) is a variation of the previous one with a focus on market share and sales (as opposed to E1-E5 which are more "financial" strategies). Figure 5 shows three types of outputs from the GTES simulation. First, we indicate the status distribution, that is, the percentage of trajectories that respectively yielded a stable, war and chaos status, as explained in Section 3.3. Second, we indicate the standard deviation of the overall satisfaction of all players. Last, we print the average satisfaction of each player.

These types of results tell a story about the different competitive situations. Obviously the E6 context is more difficult for all players than the initial situation E1. The percentage of chaotic trajectories yields a kind of "confidence index". Success for a GTES simulation translates into a large majority of stable trajectories. When this is not the case, we usually analyze the trajectories and, most often, we either find a weakness in the model or a business contradiction in the way the objectives are stated. We also look for a relative ranking of satisfactions, more than the absolute values, when we compare one "situation" against another.

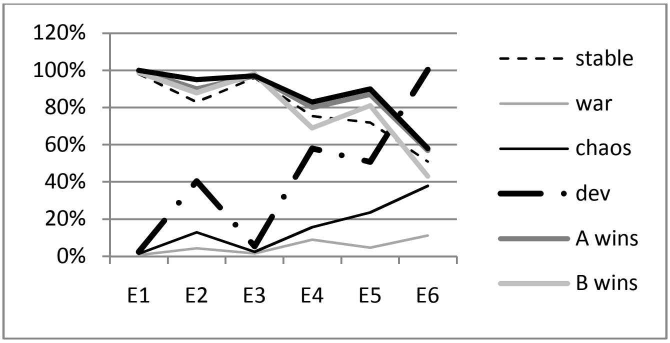


Fig. 5 Sample of GTES results on CGS problem

In this example, the "strategies" are similar for all players. Obviously we can compare the influence of strategies actor against actor, in a game theory fashion (somehow, we apply game theory twice: once internally in GTES and once when we perform manually a parametric analysis). We illustrate this with another scenario with "four players", where a new operator is introduced (with no customer base but an aggressive cost structure and a low price). The interest of the CGS model (few macro variables) is that making such an experiment is easy. The following table shows the matrix-analysis of three strategies for the three "original" players:

- S1 is a "conservative strategy" where the goal of the operator is to *maintain* its market share, its turnover and its EBITDA (actually, quite an aggressive goal).
- S2 is a quieter strategy, where the operator expects a small setback when the fourth player is introduced.
- S3 is a "financial strategy" that focuses on maintaining the EBITDA, at the possible expense of market share or turnover.

The matrix compares the strategy picked by the first player (lines) and the strategies chosen by the other two players (columns). We assume that the fourth player (the small one that get introduced) does not vary (mostly, to try to break even as soon as possible). The table shows the satisfaction of the first player against the satisfaction of the two others.

Table 2 A strategy matrix for CGS games

Player 1:	Players 2 & 3: S1	Players 2 & 3: S2	Players 2 & 3: S3
Strategy S1	Sat : 51% vs 5/74%	98% vs 5%/61%	6% vs 5%/76%
Strategy S2	23% vs 47/97%	80% vs 57%/98%	0% vs 87%/96%
Strategy S3	45% vs 9%/61%	97% vs 5%/52%	5% vs 14%/52%

This type of matrix tells that a more aggressive choice is a better strategy for Player 1, but it is also full of interesting insights. For instance, first column – second row shows that the satisfaction of the first player is worse although its goals are easier to reach. Even when all three players have a moderate strategy (second column, second row) the EBITDA is poor (remember that a fourth player is introduced). On the contrary, although the satisfaction of the first player in the first column-third row is only 45%, this translates into a better EBITDA than the "default" strategy. The real benefit from GTES is not what is obtained from such a result matrix but rather what is learned through a number of experiments, or through a "serious game" scenario. This is precisely the same argument made for the S3G (previous) example. Here is a list of more global insights that were derived by running many experiments for a couple of months:

- The best strategy for a small player is to be slightly more aggressive than the bigger ones, but not too much.
- Defining mostly financial goals for the players yield a stable game (i.e., most trajectories are characterized as stable since a pseudo-equilibrium is found). The search for market share growth is quite a different story that yield either chaotic or war trajectories (which ends with the failure of one of the players).
- If the strategies coincide towards the search for a global increase of profit, acquisition costs are lowered down while loyalty expenses (for renewals) are raised up.

- When competition increases (when all players pick more aggressive strategies), a price pressure occurs, which is then stabilized with respect to the EBITDA goal.
- This price war is favorable to large players, or more precisely, players with the best fixed/variable cost structure (who tend to be the large players, but not always).
- Competitions that generate stable trajectories display a form of "mimetic" behavior between players: price evolutions follow similar patterns. The optimal tactic found by machine learning is both prudent (price slashing occurs through small decrements) and coupled to each other's behavior.

Many of these insights either relate to common sense (something that most managers would say they knew already) or to previous results of economic studies of competition in an open market. However, each insight is still valuable because it is illustrated with "trajectories" that are closer to the business situation than most analytical models.

4.3 Two Other Applications from the Telecom Industry

We have used GTES over a number of different real-life problems during the past 10 years. Here we briefly mention two problems for which GTES has shown to be useful. The first example models the two first years of a customer lifecycle. A customer enters a shop from one of the six possible distribution channels (such as an online shop). The customer picks a package (phone and service plan) and uses the product accordingly for a year. Then, depending on the current price levels and the evolution of mobile phone technology, she or he may decide that time has come to replace the phone. A double choice is made: either to simply buy a new one or to "renew" the phone – while keeping the service plan – by taking advantage of the operator's renewal proposition. A second year of phone use is then simulated so that the customer "total two year value" may be shared between the actors.

This simple model is interesting because it describes the coupling between the phone operators through distribution channels. By optimizing their own revenue, distribution channels have a clear impact on the competition between operators. A key insight of one executive at Bouygues Telecom was to envision that "channels could think in term of full lifecycle too" and favor those operators who pay more "over a complete cycle of sales & renewal". To experiment with this model, two ingredients were needed that were absent from the spreadsheet model: to factor in the price-based competition and to optimize the behavior of the distribution channels automatically as a response to the operator's strategy. This required building a simple model on how customers would react, during each phase of the game, to price changes made by the channels. The main result achieved with GTES simulation was to confirm that it makes a lot of sense for the operator to put himself in the shoes of the distribution channels and think in terms of lifecycle. As such, it would not be worth a computational experiment, but the GTES serious game is able to translate this intuition into real numbers, where one can grasp the sensitivity of the equilibrium building (how fast do other actors react to one's own moves).

The second example deals with the allocation of resources to distribution channels, including those controlled by the operators (branded shops and web sites) for which long-term investments can be made. A critical issue is the level of internal competition between the channels for one given operator. This is hard to measure, hard to evaluate and there is no consensus among the operational experts of the subject matter. This second model was built as a tool to illustrate various scenarios and make those experts react to these scenarios.

The model is also quite simple, but is built on top of two qualitative equations whose parameters are unknown. The first one describes how customers react to price changes from the channels. The second qualitative model is a simpler channel competition matrix, which represent how likely one would switch from one channel to another. This matrix focuses the wide spectrum of opinions: for some, there was no issue, the customer would go to where the price was lower; for others, people would shop where it was convenient and where they were used to go, especially for specific demographics. Hence we develop and used GTES, to capture and play with this uncertainty and wide range of conflicting opinions. The second example – optimizing distribution costs over different channels – showed the value of practical demonstration over theory. GTES illustrates the concept of "elasticity" and how different channels can cannibalize each other. The value of the experiment does not come from the elasticity parameters (which are a "wild guess" at first) but from the numerous interaction with the channel managers who react to the GTES outcome and build, through successive iteration, a commonly accepted picture of this customer elasticity to price variation.

5 Discussion and Future Directions

Most models are quite simple from a computational point of view – tens to hundreds of parameters and tens of equations - but are already more complex than typical spreadsheet applications, because of loops and temporal series. With the smart grids example, one computation (i.e., computing $f_p(x,e)$) takes up to one second (on a fast personal computer). In the three telecom examples, computing $f_p(x,e)$ takes between 10 to 100 ms. However, the multiplying factor to run GTES is between 10^6 and 10^9. Indeed, we need to multiply three factors (three nested loops). Finding the best response tactic ($BR_V(i,t)$) typically requires between 100 and 1000 optimization cycles. Searching for pseudo-Nash equilibriums usually requires a few hundred iterations (cf. Section 3). Interweaving yields a gain of a factor of 10. Last, Monte-Carlo sampling requires computing between a few hundred up to a few thousand trajectories. All this translates into total run time ranging from one day to a year (with a complex model or if Forward Nash is introduced). Since this is obviously not practical, we compromise for less precise/stable experiments: we both reduce to sampling size and reduce the time allocated to Nash convergence. We still use long-running sessions to perform the analysis and the fine-tuning (from an hour to a day of computing), and run simpler/faster experiment when playing the serious game (a few minutes). However, it is clear that GTES is ideally suited for parallel computation, since the search for the best response in V may be parallelized (using classical local search techniques), the search for the pseudo-Nash equilibrium can process the moves in parallel and Monte-Carlo sampling is obviously a candidate for parallel evaluation.

One major benefit of the GTES approach is that it is a true "torture test" for models. The combination of random sampling and machine learning (which will find and exploit all breeches in the model if there is a way to increase satisfaction through a poorly calibrated equation) "shake" the input model and quickly point out faults. A very similar observation was made ten years ago when we worked on automatic generation of optimization algorithm for vehicle routing [24]. Using machine learning coupled with random exploration technique over parameterized algorithms is an effective way to discover bugs and limitations. In particular, GTES requires a stable behavior of the model "at its boundaries". Very often, the simpler equations of a "naïve model" are very rough when the values get close to their limits of validity. One of the benefit of S-curves (over linear equations) is precisely to combine the differential analysis of linear/derivative approaches with an overall "global system" approach (defining overall boundaries). Another complexity comes from the multi-criteria nature of "player's satisfaction". It usually takes a while to tune the formula that defines satisfaction (a weighted combination of terms that tell how far the player is from his strategic goals – cf. Section 2.2 for the S3G example). For GTES to deliver interesting "serious gaming", the satisfaction objective function needs to be tuned with domain experts.

In addition to model definition, neighborhood structure and exploration strategies are key choices as far as performance and quality of the results are concerned. The performance part is rather obvious and was mentioned earlier: larger neighborhood and more sophisticated local search strategies increase the computing time. The second point is that if the local search strategy is too simple, experience shows that we get fewer stable strategies. There is a compromise to be found: in many cases we had to increase the neighborhood structure to obtain a satisfactory game, that is, to ensure that all "logical answers" (or defined as such by domain practitioners) were actually found by the local search phase of GTES. The criterion that we use is to make sure that local search always gets the same performance level irrespectively of the initial (randomly chosen) tactic. We have experimented with genetic algorithms in the past [1], and we plan to experiment with more complex neighborhood structures and Tabu search to improve the speed and accuracy of learning, which translates into better performance and stability for GTES.

6 Conclusions

Complex problems in an enterprise require *learning* more often than they require *solving*, because of the uncertainty, the feedback loops with the environment and the rapidly changing nature of business. The GTES approach is a practical toolbox to play with under-specified models and transform them into serious games. Through the combination of techniques from evolutionary game theory, operations research and stochastic simulation, GTES is a workbench where complex models may be simulated and interacted with. GTES is a generic approach, which could be used for a variety of problems. We have shown here a few applications, ranging from complex and speculative models, such as “Systemic Simulation of Smart Grids” to the very practical business applications of Section 4.3. GTES has been used, to give another example, to evaluate the best strategies for the 2011 800 MHz LTE bidding in France.

The value of the GTES approach is demonstrated through *experiments*, when the strategies of one or a few players are changed and we may observe how the other actors would react. This is most of all a learning experiment, which is why the model needs to be simple (i.e. each business variable needs to make “business sense” for the human players). When the outcome seems surprising or counter-intuitive, it is often necessary to look at a few trajectories, which is why “white box” approaches are preferred. If the goal is to run simulation as a forecasting tool, a “black box” may work, once the technique has acquired some credibility, but if the goal is to learn from practical experiments, it is crucial that the participants understand how the model works.

References

1. Caseau, Y.: GTES: une méthode de simulation par jeux et apprentissage pour l’analyse des systèmes d’acteurs. RAIRO Operations Research 43(4) (2009)
2. Weibull, J.: Evolutionary Game Theory. The MIT Press (1995)
3. Gibbons, R.: Game Theory for Applied Economists. Princeton University Press (1992)
4. Korn, G.A.: Advanced Dynamic-system Simulation: Model-replication Techniques and Monte Carlo Simulation. Wiley Interscience (2007)
5. Slantchev, B.: Game Theory: Repeated Games. University of California – San Diego (2004), `http://polisci.ucsd.edu/~bslantch/courses/gt/07-repeated-games.pdf`
6. Fudenberg, D., Levine, D.: The Theory of Learning in Games. The MIT Press (1998)
7. Aarts, E., Lenstra, J.K.: Local Search in Combinatorial Optimisation. Wiley (1993)
8. Jørgensen, S., Quincampoix, M., Vincent, T. (eds.): Advances in Dynamic Game Theory: Numerical Methods, Algorithms, and Applications to Ecology and Economics. Annals of the International Society of Dynamic Games. Birkhauser, Boston (2007)
9. Nissan, N., Roughgarden, T., et al.: Algorithmic Game Theory. Cambridge University Press (2007)
10. Duncan Luce, R., Raiffa, H.: Games and Decisions – Introduction and Critical Survey. Dover Publications, New York (1957)
11. Alkemade, F., La Poutré, H., Amman, H.: Robust Evolutionary Algorithm Design for Socio-economic Simulation. Computational Economics (28), 355–380 (2006)
12. Siarry, P., Dréo, J., et al.: Métaheuristiques pour l’optimisation difficile. Eyrolles, Paris (2003)

13. Horst, R., Tuy, H.: Global Optimization: Deterministic Approaches. Springer, Berlin (2010)
14. Milano, M.: Constraint and Integer Programming. Kluwer Academic Publishers (2004)
15. Axelrod, R.: The Complexity of Cooperation- Agent-Based Models of Competitions and Cooperation. Princeton University Press (1997)
16. Nelson, R., Winter, S.: An Evolutionary Theory of Economic Change. Belknap Harvard (1982)
17. Ferber, J.: Multi-agent systems: An introduction do distributed artificial intelligence. Addison-Wesley (1999)
18. Kandori, M., Mailath, G., Rob, R.: Learning, Mutation and Long Run Equilibria in Games. Econometrica 61(1), 29–56 (1993)
19. Blum, A., Blum, M., Kearns, M., Sandholm, T., Hajiaghayi, M.T.: Machine Learning, Game Theory and Mechanism Design for a Networked World. NSF proposal (2006), `http://www.cs.cmu.edu/~mblum/search/AGTML35.pdf`
20. Forrester, J.: Principles of Systems. System Dynamics Series. Pegasus Communications, Waltham (1971)
21. Sterman, J.: Business Dynamics – System Thinking and Modeling for a Complex World. McGraw Hill (2001)
22. Lucas, S.M., Kendall, G.: Evolutionary Computation and Games. IEEE Computational Intelligence Magazine (February 2006)
23. Bowling, M., Veloso, M.: An Analysis of Stochastic Game Theory for Multiagent Reinforcement Learning. Carnegie Mellon University, CMU-CS-00-165 (2000)
24. Caseau, Y., Silverstein, G., Laburthe, F.: Learning Hybrid Algorithms for Vehicle Routing Problems. TPLP 1(6), 779–806 (2001)

Chapter 3
Architecting Complex Systems in New Domains and Problems: Making Sense of Complexity and Managing Its Unintended Consequences

Patrick Godfrey

Abstract. Complex problems usually span many technical domains. Formalised methods have been developed and reported to address defence, aerospace and ICT issues. Architecting approaches for other domains and problems such as for infrastructure do not yet exist. Design methods need to be developed to address these complex problems. This paper reports on an ongoing programme of teaching and *Learning Together* that is developing an approach for the creation of systems architectures. The paper reflects on the work of some 300 Research Engineers and students who have been engaged in designing complex sustainable systems. It characterises formative principles for architecting frameworks and indicates ways in which they can be used to deliver emergent properties and manage unintended consequences.

1 Purpose

Most of current Systems Architecting guidance focuses on architecture frameworks developed for specific defence and information system purposes (e.g. NAF, DODAF, MODAF, Zachman). Architecture frameworks are a means of dealing with system complexity. **While the principles behind these frameworks, and the domain-independent skills of the System Architect (ref e.g Rechtin, "Systems Architecting"), are widely applicable, Architecture Frameworks developed for specific applications areas are not.** However the need for complex systems design methodology is not restricted to these domains but challenges most industries. Some Factors driving this include:

Patrick Godfrey
University of Bristol, Office 0.37b
Queen's Building, University Walk, Clifton BS8 1TR,
Great Britain
e-mail: patrick.godfrey@bristol.ac.uk

- The need for sustainability in the face of climate change and resource availability constraints
- Trend towards globalization of industrial markets, ownership and manufacture,
- Creation by the internet of constantly evolving global knowledge systems
- The interdependence of nations in the face of unintended consequences of human activity eg acid rain, nuclear disasters, ozone layer depletion.

The key issue is that all of these topics cut across traditional engineering and other disciplines, so no individual discipline's set of mental models and language is sufficient to manage the whole-system issues. The need to address this topic is becoming particularly urgent in the infrastructure sector in view of the economic importance attached to interdependency [1].

We will in this paper use the need to architect (design) sustainable complex systems as a means of developing a generic approach to designing or problem resolution in complex systems, for three related reasons:

1. We believe that what we learn from this approach can be abstracted for general applicability
2. Sponsored by the Royal Academy of Engineering UK, we have had a 5 year collaborative programme, with industry of teaching and learning, with this as its focus.
3. In addition about 25 % of the Research Engineers in the Systems Centre are undertaking collaborative research with industry concerning the sustainable development of infrastructure.

This paper abstracts some generic learning from the 300 Masters level and Doctorate level assignments and theses. Table 1 lists assignment topics from the Masters level unit. We are seeking to discover how students and researchers who have been introduced to the principles for architecting systems referred to above, learn to apply them to such a diverse range of systems problematiques.

Table 1 Examples of topics addressed in the Sustainable Systems Programme

Countries	Haiti, Afghanistan
Mega projects	3 Gorges dam, Crossrail, London, Olympics, Aircraft carrier, Airbus A380.
Managing resources	Polar Mineral Extraction, Rainforest, Carbon capture, coal fired Power stations, Hydrogen Infrastructure, Eating Meat, Euro currency
Institutions / companies	NHS, Supermarket Chain, BP, University
Leisure	F1 Motor sport, Rugby World Cup, Eden Project, Ski resort in Dubai, Rare earth metals
Infrastructure	Nuclear Power, Air Transport, Sustainable Tourism, An eco-district, Internet infrastructure

1.1 Stakeholder Alignment

To an ecologist a tree is a complex interaction of a wide diversity of organisms. To the structural engineer it may be much less complex because the calculation of loading from any branch is deterministic and calculable from the mass distribution within the tree. The complexity of a system can therefore be related to the viewpoint of the stakeholder and the purpose of the system. Sillitto [2] calls this 'subjective' complexity. Physical systems have no purpose in themselves, this is an attribute that is conferred by people. For example consider, a kitchen knife which could be part of a system to prepare vegetables but that same knife might be a murder weapon. The knife has not changed. It is the change in its purpose that changes the outcome.

Since a lot of the complexity in most systems arises from the intentionality of the people involved, most of whom may be influenced from within the system but cannot be controlled, it follows that an analysis of the different stakeholder viewpoints, to establish their interests and issues with the purpose of the system, provides an excellent starting point. It is often the lack of attention to this throughout the life of the project that is the source of unintended consequences.

In common with Zachman [3], we have found it useful to adopt a simple common language to describe both hard (physical) and soft (people) systems based upon Kipling's 6 natural language questions to define any system (or process): Why? How? What? Who? When? Where? [4] . The purpose (Why) is delivered by the means (How) operating on the other 4 attributes (What, Where, When and Who)[5]. This idea was important to the development of Terminal 5 at Heathrow. The 'Why' team, which represented the client's interest worked together with the 'How' team to resolve what would be done, by whom when and where. A simplified version of the process is shown in Figure 1 [4].

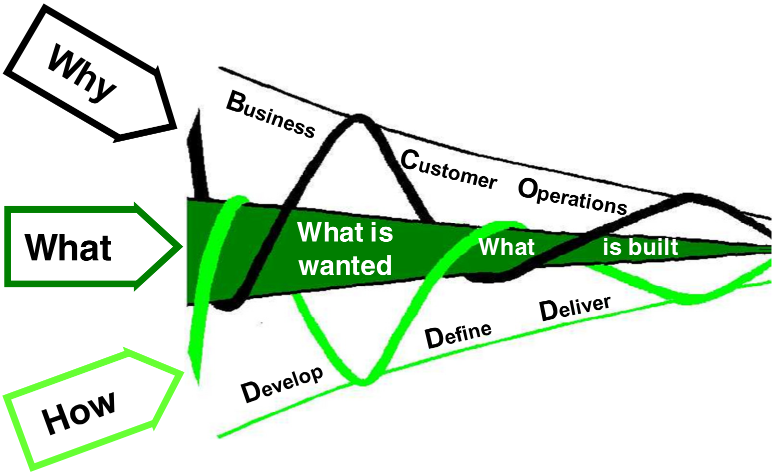


Fig. 1 Design spirals for construction (after T5 Handbook)

2 Dealing with Complex Problem Situations

Kurtz and Snowden [6] Figure 2 provide a knowledge orientated view of complexity which has been found to be particularly helpful because it helps to diagnose the type of process needed to deal with it. It postulates that in a complex state, "cause and effect are only coherent in retrospect and do not repeat", we require pattern management and perspective filters to make sense of the system. The system is managed by a process of 'probe-sense-respond' which implies that the need is to learn our way to understanding the system for its specified purpose rather than just knowing it well enough to be able to make dependable predictions. Each stakeholder has a perspective filter or point of view that will be different from that of other stakeholders.

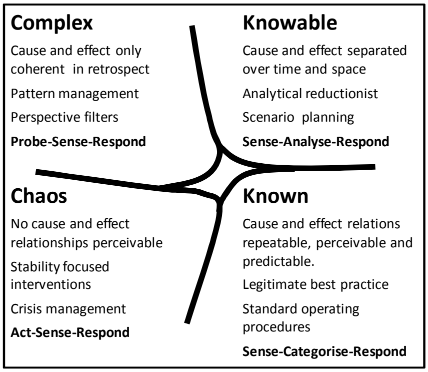

Complex
Cause and effect only coherent in retrospect
Pattern management
Perspective filters
Probe-Sense-Respond
Knowable
Cause and effect separated over time and space
Analytical reductionist
Scenario planning
Sense-Analyse-Respond
Chaos
No cause and effect relationships perceivable
Stability focused interventions
Crisis management
Act-Sense-Respond
Known
Cause and effect relations repeatable, perceivable and predictable.
Legitimate best practice
Standard operating procedures
Sense-Categorise-Respond

Fig. 2 Sense making in a complex world after [6]

2.1 Emergent Properties

Outcomes from complex systems can be seen as 'emergent properties' of the system. These properties derive from the relationship between the components not just the properties of the components themselves and these relationships are often many to many within the system. Just as the positive aim is to deliver intended outcomes so it will be necessary to manage the unintended emergent behaviours throughout the life of the system. This is particularly important in the commissioning phase of new complex systems. Learning processes need to be included. Unfortunately in many large scale systems there can be no opportunity to build a prototype. It follows that the system has to be designed to have sufficient learning processes within it, combined with resilience and adaptability so that unintended consequences are managed to a successful outcome.

2.2 Core Process for Architecting Complex Systems

By reflecting the work of 300 students and Research Engineers, common themes have been extracted from the more successful and reflected on to industrial experience with the help of Industrial Partners at the Systems Centre, as identified in acknowledgements below. It is apparent that there is a fundamental organising principle underpinning success. At its top level this is shown in Figure 3.

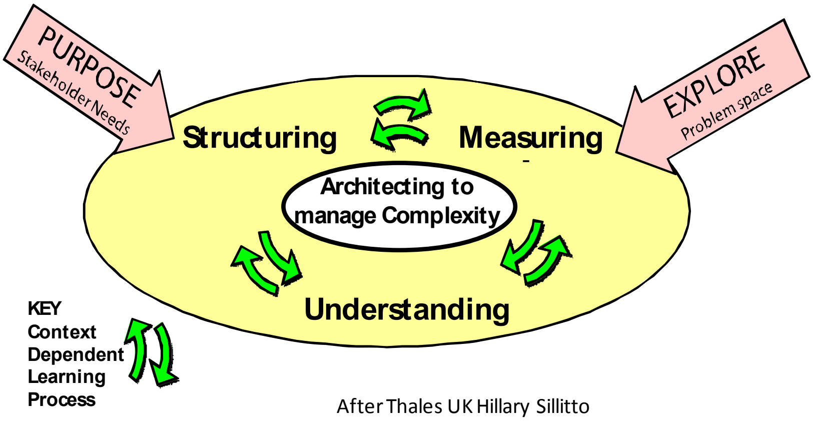

PURPOSE
Stakeholder Needs
EXPLORE
Problem space
Structuring
Measuring
Architecting to manage Complexity
Understanding
KEY
Context Dependent Learning Process
After Thales UK Hillary Sillitto

Fig. 3 Architecting process for the design of a complex system

2.3 Frameworks for Structuring and Measuring

The concept of a framework for systems architecture owes its origins to information systems and particularly Zachmann [2] and is seen as *a set of tools which can be used for developing a broad range of different* architectures [7]. As indicated earlier, it is necessary to go back to first principles, when addressing new domains and problems, because if a framework is not suitable to a particular domain it tends to obfuscate rather than clarify. Zachmann's specialisation of the general structure to the information system domain does not seem to be transferable to other domains. Frameworks are used to reduce perceived complexity by separation of concerns hence generating understanding in the face of complexity. Inappropriate frameworks increase the perceived complexity. The goal of the context dependent learning process shown as double arrows in the diagram, is clarity of understanding. This provides a criteria for success for the framework generation process. The double arrows indicate repeated iterations until sufficient clarity is generated. However this requires experience and an ethical judgement that avoids confusing clarity with simplicity.

3 Attributes of a Good Framework

The problem is usually too complex to be dealt with in a single diagram instead as with Zackman a layered approach is recommended. By exposing these relationships, changes can be proposed and evaluated. Frameworks generally need to be multidimensional, representing various important viewpoints on the problem situation. The attributes of a good framework include that the dimensions in any one view should be "MECE" – mutually exclusive, and collectively exhaustive [8]. The dimensional subdivisions should also be of a similar level of importance.

3.1 Stakeholder Needs Defining Purpose

It is clearly necessary to explore the problem space in order to appreciate the context. However a key process is also to identify purpose through an analysis of stakeholder views of the system. This links the framework to purpose ie Why, through the stakeholders who are generally a significant source of complexity. The processes tend to identify particular issues and relationships. There are a wide range of tools for this including: UML Use cases, analysis of roles and responsibilities, importance-influence analysis[3] also grounded theory interviews[9] and story telling [9].

4 Sustainability

Sustainability is a desired attribute of many systems. It is complex and multidimensional. There are many examples of narrow definitions of sustainability leading to decisions being made, that have been at best partially effective and at worst counterproductive when considered holistically. These unintended consequences, resulting from well intentioned decisions have been a motivating force for the sustainable systems research described earlier. For example:

- the decision to insist on the use of bio-fuels in transport, lead to the emergence of food riots caused by change in agricultural and market driven practice [10]
- The un-sustainability of the green energy subsidies [11]
- Forest fire suppression causes greater tree density and fuel accumulation, leading to larger, hotter, and more dangerous fires, often consuming trees that previously survived smaller fires unharmed. [12]
- the emergence of instabilities in the stock market from automatic trading [13]

Sustainability itself has different meanings for people with different points of view. To eliminate this potential for confusion, the sustainable systems team at Bristol shared their understandings and through a process of group model building developed a simple layered structure for Sustainable Systems as shown in Figure 4: Blue and green sustainability.

Green sustainability applies to the Sustainable Development of infrastructure. The Bruntiland Report [14] defines it as "development that meets the needs of the present without compromising the ability of future generations to meet their own needs". For example wind farms, reed beds and wind up radios may all contribute to “green sustainability”. Blue sustainability refers to systems that are capable of operating long term in conditions that are very challenging because they are isolated, safety critical or extreme. Examples include: space, nuclear reactors and deep sea. Both are seen to be within a meta view that requires the outcome to be efficient, effective, resilient, robust and affordable.

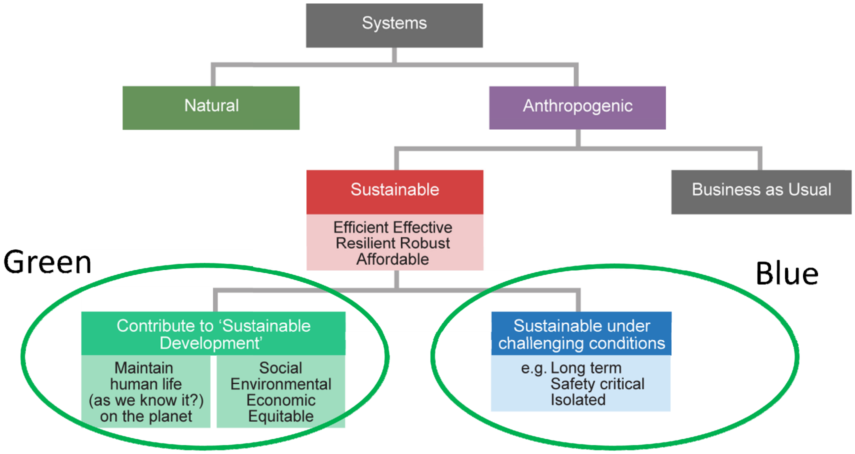


Fig. 4 Blue-green sustainability

4.1 Holistic Dimensions for Sustainability

For business there are a range of holistic scales that are used to establish a meta framework for business strategy analysis. PESTLE: Political, Economic, Social, Technical, Legal, Environmental is a commonly used example. This is sometimes extended to STEEPLED to identify additional sub-division of the whole for Ethics and Demographics. One of these can be used for sustainable systems design. The 5 capitals scale [15] Figure 5, being expressed as capitals, recognises that resource is finite and at least conceptually quantifiable. It too can have additional subdivisions. For example, some would distinguish between 'Social' and 'Institutions' because in some contexts institutions are as important as social and human. So, it becomes 6 capitals. For example the Regulator(s) of a system can have a profound and not necessarily beneficial influence on the sustainability of a system.

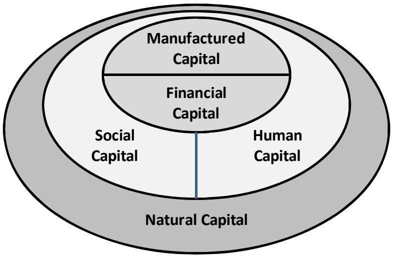


Fig. 5 Five capitals

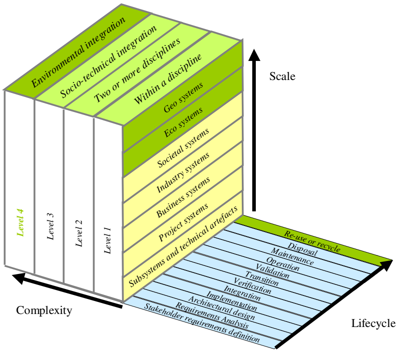


Fig. 6 Extension of standard scales for sustainability

Alternatively standard scales can be extended as is shown in Figure 6 [16] to include for environmental measures. In this case 'lifecycle' is an extension of the scale provided in ISO 15288, 'scale' is an extension of Hitchin's 5 level scale[17].

Complexity is an extension of Supples scale published by Royal Academy of Engineering [18].

5 Using the Framework

Conceptually the layered framework is a problem structuring method that sits between policy or need and a real world context Fig 7. Ideally the framework should be used to test policy before it is enacted as is evident from the examples identified under the sustainability heading.

Because performance depends upon the interaction of the components it is usually necessary to use interpretive models to understand what is going on and predictive models to understand the implication of the interdependencies which can be multidiscipline. Systems Dynamics[19] and concept mapping are useful tools to understand causal loops. Systems Dynamics is extensible to simulation with the inclusion of stock-flows. Agent based modelling is also useful and can be used to feed relationships into the simulation. Sometimes it is useful to use shared model building to engage people who are involved in the process [20].

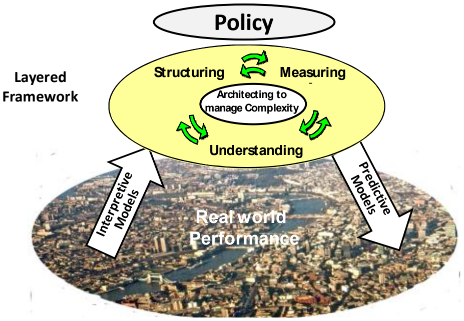


Fig. 7 Developing the Framework to inform policy

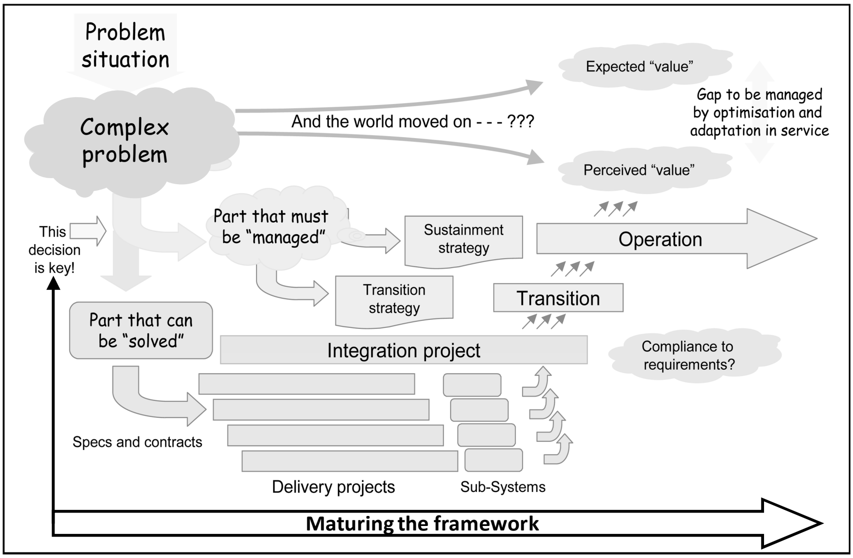


Fig. 8 Complex Systems Engineering after Sillitto [2010]

Figure 8 superimposes the framework development process on Sillitto's model for complex system engineering of ultra large scale systems[21]. The framework needs to be in place to inform the initial key decision as to what part of the problem is engineered and what part is to be managed on an ongoing basis. It is very risky to establish a project before it is in a sufficiently knowable state. The logistic disruption that occurs when an unintended consequence emerges is generally very costly and time consuming. There are traditional processes for this transfer. For example the process of obtaining planning permission resolves a range of social objections to the project which could otherwise delay and disrupt the work. These processes are necessary but not sufficient. For example, the start of the Channel Tunnel Rail Link was contemporary with the Newbery by-pass protests but through effective stakeholder management the vulnerability to protests was avoided.

6 Conclusions

This work has shown the importance of establishing a meta framework and language to deal with the emergent behaviours and unintended consequences that characterise complex systems.

Architectures are intended to generate understanding in the face of complexity that will move the problem into a knowable state at least sufficiently to enable design that will fulfil the systems purpose. Stakeholder needs define purpose but also reveal conflicts that will need to be managed. Once the important relationships have been identified then causal loops can be abstracted and if necessary extended through stock flows to provide simulations of performance. At the top level these can be used to test the economics or policy. At the systems engineering level, it can also provide a base line to monitor emerging knowledge from complexity which will inevitably be the source of uncertainty. It is also used to design and manage a project portfolio approach to project organisation as was the case for the London 2012 Olympics. The learning process is particularly important at the commissioning stage when full integration in service can and normally will reveals unintended consequences. In order to manage the uncertainty it is necessary to have a design that is both resilient and adaptable.

Acknowledgments. I wish to pay tribute to the enthusiasm and creativity of the Research Engineers and Students that have contributed to the basis for this paper. I also wish to express my appreciation of the other professionals and academics who have participated in this sustainable systems programme. This includes the unstinting contributions by: Peter Head, Gordon Hutchinson, David Oxenham, Ges Rosenberg, Chris Broadbent, Neil Allan, Mike Yearworth, Linda Farrow, John Davis, Edd Godard, Paul Best, Ruth Deakin-Crick, David Blockley. I particularly wish to pay tribute to Hillary Sillitto of Thales UK, who has been my mentor and the source of critically important experience in the Sustainable Systems programme and this paper. None of this would have been achievable without the support and funding from the EPSRC, Royal Academy of Engineering and the University of Bristol.

References

1. Frontier, Systemic Risks and Opportunities in UK infrastructure, HM Treasury & Infrastructure UK (2012), `http://www.frontier-economcs.com/_library/publications/Frontier%20Report%20-%20Systemic%20Risk%20and%20Opportunities%20in%20UK%20infrastructure.pdf`
2. Sillitto, H.: On Systems Architects and Systems Architecting: some thoughts on explaining and improving the art and science of systems architecting. In: Proceedings of 19th Annual International Symposium of INCOSE, Singapore (2009)
3. Zachmann, J.: A framework for information systems architecture. IBM Systems Journal 26(3) (1987)
4. Blockley, D., Godfrey, P.: Doing it differently, Systems for rethinking construction, p. 45, p. 280. Thomas Telford, London (2000) ISBN 0-7277-2748-6
5. Blockley, D.: The Importance of Being Process. Journal of Civil Engineering and Environmental Systems 27(3) (2010)
6. Kurtz, C., Snowden, D.: The New Dynamics of Strategy: sense making in a complex and complicated world. IBM Systems Journal 42(3), 462–483 (2003)
7. The Open Group Architecture Framework, `http://en.wikipedia.org/wiki/The_Open_Group_Architecture_Framework`
8. Rasiel, E.: The McKinsey Way, 1st edn. McGraw-Hill (1999) ISBN978-0-07-053448-3
9. Strauss, A., Corbin, J.: Basics of qualitative research: Grounded theory procedures and techniques. Sage Publications (1990)
10. Braun, J.: Biofuels, International Food Prices, and the Poor. Testimony to the United States Senate Committee on Energy and Natural Resources (2008)
11. Economist, `http://www.economist.com/node/21532285` (accessed October 20, 2011)
12. US Forest Service. Influence of forest structure on wild fire behaviour and the severity of its effects (2003), `http://www.fs.fed.us/projects/hfi/science.shtml` (accessed March 29, 2012)
13. Buchanan, M.: Dangerous Liaisons, pp. 21–24. Physics World (March 2011)
14. Brundtland Commission, Our Common Future. Oxford University Press (1987)
15. Forum for the Future. Five Capitals (2011), `http://www.forumforthefuture.org/project/five-capitals/overview` (opened August 17, 2012)
16. Sillitto, H.: Unravelling Systems Engineers from Systems Engineering: frameworks for describing the extent variety and ambiguity of system Engineering and Systems Engineers. In: Proceedings of 21st Annual International Symposium of INCOSE. Denver (2011)
17. Hitchins, D.: World class systems engineering in the UK. In: Euroforum Conference Managing Systems Engineering in UK for Competitive Advantage, Kensington Close Hotel, London, June 28-29 (1995)
18. Elliot, C., Deasley, P.: Creating Systems that Work. Royal Academy of Engineering (2007), `https://www.raeng.org.uk/education/vps/pdf/RAE_Systems_Report.pdf` (opened August 17, 2012)

19. Sterman, J.: Business Dynamics: Systems Thinking and Modelling for a Complex World. Irwin/McGraw-Hill, Boston (2000)
20. Dunford, C., Yearworth, M., Godfrey, P., York, D.: Using Systems Practice to Enable Quality in Design. In: IEEE International Systems Conference (SysCon), Vancouver, BC (2012), doi:10.1109/SysCon.2012.6189497
21. Sillitto, H.: Design principles for Ultra-Large-Scale (ULS) Systems. In: Proceedings of 20th Annual International Symposium of INCOSE Chicago (2010)

Chapter 4
Agility Problems in Traditional Systems Engineering – A Case Study

Emrah Asan and Semih Bilgen

Abstract. Well-established systems engineering approaches are becoming more inadequate as today's systems are becoming more complex, more global, more COTS/re-use based and more evolving. Increased level of outsourcing, significant amount of subcontractors, more integration than development, reduced project cycles, ecosystem like collaborative developments, software product lines and global development are some of the changes in the project life cycle approaches. In this paper, we present the results of an exploratory case study which tries to identify the agility problems in large scale software intensive defense projects. This is the first step of our research in which the overall objective is to improve the agility attributes of the traditional systems engineering approach.

1 Introduction

A lot of things have changed since F.P. Brooks made the famous statement that there was no silver bullet for the software problem back in 1987 [1]. Changeability and complexity, essential properties of software, are still there and no silver bullets have been produced to deal with them. The only thing that is changing about this fact is the increasing size and complexity of the software. Today we are talking about large scale, complex software intensive systems of systems (SISOS) in almost all sectors.

Size and complexity of projects/systems, increased use of COTS and subcontractors, global team structure, issues of interoperability with legacy systems,

Emrah Asan
EADS Deutschland GmbH.
Landshuter Str. 26
D-85716 Unterschleissheim, Germany
e-mail: emrah.asan@gmail.com

Semih Bilgen
Department of Electrical & Electronics Engineering,
Middle East Technical University
06800 Ankara, Turkey
e-mail: bilgen@metu.edu.tr

re-use considerations, large number of stakeholders with conflicting interests constitute only a subset of the problems that result from the changes in the systems engineering (SE) environment [2–7].

SISOS development can be viewed from three aspects: business, system and software [6]. Business layer has already defined the need for agility in the development and acquisition of software intensive systems. In the section 804 of “National Defense Authorization Act for Fiscal Year 2010, US congress has directed the US Secretary of Defense to “develop and implement a new acquisition process for IT systems”, especially for the rapid fielding of urgent operational capabilities. National Defense Authorization Act also states that the new process should be based on the March 2009 report of the Defense Science Board Task Force on Department of Defense Policies and Procedures for the Acquisition of Information Technology, which states “The conventional DOD acquisition process is too long and too cumbersome to fit the needs of the many IT systems that require continuous changes and upgrades” [6] [8].

Project management communities have already started to question the traditional project management approaches [9]. It is good news that PMI declared a new certification program in agile project management [10]. "Agile Process" has already found itself a place in the "Chaos Success Factors" list in the Standish group's chaos report in 2009 [11].

The pitfalls of the traditional system development approaches are also addressed in the software layer and several agile development methodologies are developed. Since the declaration of the agile manifesto (maybe earlier than that), agile methods have been criticized such that they were not proper for the large scale, complex system projects. However, while approaching agile methods as software engineering methodologies and comparing them with the plan driven methods, we have spent relatively less effort in trying to identify and solve the actual problem which was the need for agility in the large scale, software intensive system projects or to put it in another way, the “need for agile SE”.

In this paper, based on the motivation detailed in our previous work [12], we present the results of an exploratory case study which tries to identify the agility problems in large scale software intensive defense projects. This is the first step of our research in which the overall objective is to improve the agility attributes of the traditional SE approach.

2 Research Method

The overall objective of this research is to first uncover the agility related problems of the traditional SE approach in the large scale, software intensive, socio-technical military system projects and then to address some of them in order to improve the agility of the traditional SE approach. In this initial study we want to make an empirical start in exploring/validating the problems with the traditional SE approach. Our intention is not to study all kinds of problems but the ones related to the agility attributes as described below. Although we call it agile SE, what we aim at is a SE approach with the necessary improvements to deal with the trends and challenges of today's SISOS projects that we summarized in [12].

The perception of agile has almost always been limited to "fast", and "document-free" [13]. In this study, the discussions on agile SE are based on the agility attributes defined in [5] for the SE processes: Flexibility, learning attitude, focus on customer value, short iterations delivering value, continuous integration & test driven development, lean attitude and team ownership.

Since a key characteristic of any agile approach is its ability to deal with changes (i.e. flexibility), we shall first analyze the changes in large scale software intensive system projects. The following research questions are identified as an entry point to the subject:

RQ1: What are the main reasons for changes in the large scale, software intensive, socio-technical military system projects?

RQ2: In which lifecycle phase do most of the requirements changes occur?

RQ3: Which of these changes could be categorized as expected changes?

The research was designed as an iterative exploratory case study which allowed us to come up with new research questions during the iterations between data collection and data analysis processes. In such an approach, researchers develop categories and meaning from data through an iterative process. That understanding is then tested and modified through cycles of additional data collection and analysis [14].

The main data collection method of the study was semi-structured open-ended interviews. Interviews were initiated on a template of pre-defined questions (questionnaire) but new questions are discussed according to the answers obtained. Most of the initial interviews were face to face or telephone interviews but due to the iterative nature of the case study, re-discussion of the answers, summary of the interview results and discussions of the new issues were conducted via telephone interviews and emails.

We have conducted 59 interviews. Interviewees were selected from 10 different defense companies and from 18 different projects. Selected companies are mid-to-large scale companies from around the world (4 from Turkey, 3 from Europe, 2 from USA and 1 from Australia). Our target population is systems engineers with more than 5 years experience of which at least 3 years is gained in a large scale software intensive project. At least 3 systems engineers and at most 4 systems engineers are selected from each project. Projects are selected with team sizes larger than 30 engineers, duration more than 2 years and budget more than 10 million US dollars. Each project is a new system development project.

Although there are slight differences in the adapted traditional waterfall model between the projects, all systems engineers agree on the linear lifecycle model as: requirements, design, development, verification and maintenance.

3 Case Study Findings

3.1 Reasons for Change

Agility attribute "flexibility" is about dealing with changes. In today's SE environment, changes occur in the concept, scope, requirements, design and even in the already implemented part of the system during the system lifecycle. However, in the scope of this study, we limited our focus to the changes in the requirements (especially system requirements) with the assumption that all the other major changes should be reflected by the changes in the requirements.

Each interviewee was asked to identify 3 major reasons for the requirements change. In addition to this, in order to keep the set more understandable and workable, similar reasons are grouped under more general definitions. For example, different answers like:

- Late understanding of the operational processes;
- Late identification of the complete set of users, their responsibilities and interactions;
- Increased understanding of the customer's decision making mechanism;
- Late discovery of the existing system's performance limitations on the operations

are all grouped under "increased understanding of the operational processes". Table 1 summarizes the major reasons for requirements change.

Table 1 Reasons for Requirements Changes

ID	Reasons for Change	Total	%
R1	Increased understanding of the initially defined operational problem	45	76
R2	Late clarification/definition of the non-functional requirements	37	63
R3	Late understanding of the capabilities/constraints of the COTS/outsourced products	36	61
R4	Changes in the technology	23	39
R5	Initial problem has changed as a consequence of other changes	22	37
R6	Late understanding of the legacy/external system constraints.	14	24

3.2 Operational Domain Knowledge

The primary reason for requirement changes is the increased understanding of the initially defined operational problem. R1, R5 and R6 are considered as closely related to the operational domain knowledge. Many different answers are collected addressing this problem from different perspectives but all were related to the customer's operating environment and the missing knowledge regarding this environment such as customer's business processes, organizational structure, decision hierarchy, user types and their interaction, etc. During the interviews, all systems engineers stated that their main problem was to understand the customer's operating environment and their problems in this environment. For example, only one systems engineer stated that they had real difficulty in developing the technical solution after the customer's operational problem was clearly understood.

On the other hand, the case study results showed us that although the systems engineers are having problems due to the lack of operational knowledge, most of them do not make use of the relevant SE processes and methods that are already defined in the traditional SE literature to address this knowledge gap. Traditional SE literature proposes processes and methods for better understanding of the customer's operational needs and the problem domain such as concept of operations (ConOps) development, scenario development, prototype development, stakeholder analysis, etc. However, systems engineers do not consult them. Following are some of the findings from the case study:

- Only %24 of the systems engineers stated that they have a defined ConOps development (or a similar one) process in the project/organization. However, the resources and time allocated to this process is not enough to conduct this process properly.
- None of the projects/organizations has a defined process (or a guideline) for
  - Operational scenario development,
  - Prototype development,
  - Stakeholder analysis.
- None of the organizations has a business analyst/business process engineer (or a similar) role in the team. In any of the projects, there isn't a task defined in the WBS to capture and document the existing business/operational processes of the customer.
- None of the systems engineers establish traceability from the requirements to the source stakeholder and his/her responsibilities in the organization.
- No task or document exists for the complete definition of the stakeholder and user types.
- None of the systems engineers has ever defined effectiveness measures for the system.

Such findings were surprising as the traditional SE approach receives most critiques on the heavy processes and resulting documentation. However, our observations indicate that documents are produced but the process steps (e.g. analysis) to produce such documents are not properly conducted. As a result, SE artifacts were produced without the valuable information in them.

Highlight1: Operational domain knowledge is defined as the most valuable knowledge by the systems engineers. This knowledge is indicated as a key to understand the real problem to be solved.

Highlight2: Systems engineers do not perform the requirements/knowledge elicitation and analysis processes that are proposed by the traditional SE literature to address the operational domain knowledge.

3.3 Customer Interaction

As the interviewees highlighted the need for the operational domain knowledge, we also discussed the interaction of the systems engineers with customers. The case study findings are summarized in Table 2, Table 3 and Table 4.

The results are mostly surprising. It was expected that most of the systems engineers would have difficulty in accessing the customer, especially since these are all military projects. However, the results showed that more than half of the systems engineers (%53) were able to communicate with the customer whenever they needed customer support. Similarly, a significant portion of the interviewees (%39) stated that they could reach the customers when a serious problem occurs. In total, %92 of them did not have real difficulty in reaching the customer when they needed.

On the other hand, we observed that the real problem about customer collaboration is not the difficulty in accessing the customer but the difficulty in communicating with the correct customer representative with the required knowledge, experience and decision authority. %75 of the systems engineers stated that they did not have a chance to work with the specific customer representative but with the one that is provided by the customer. Such kind of situations did not help the systems engineers to resolve their problems.

Table 2 Ability to Reach Customer

I can communicate/collaborate with the customer	Total	%
Whenever I want	31	53
Whenever a significant problem occurs	23	39
When the customer wants	5	8
Not at all	0	0

Table 3 Ability to Select Customer Representative

I can chose the customer representative according to my needs	Total	%
Yes	15	25
No	44	75

Table 4 Need for the Customer Collaboration vs. Project Phase

I need the customer collaboration mostly during the:	Total	%
Requirements Phase	42	71
Design Phase	13	22
Development Phase	4	7

The case study also revealed that there is no specific effort to increase the level and effectiveness of communication and collaboration with the customer representatives. Apart from the planned review meetings with the customer (such as requirements review, preliminary design review, critical design review, etc), all the communication and collaboration activities are ad-hoc upon the request of the SE team. For example, in all projects there were detailed training requirements in the contract. However, in any of the projects trainings were not planned and detailed until the end of the development phase and were not performed after the whole system is tested.

Another observation indicates that systems engineers need customer collaboration mostly during the requirements phase. This finding is in parallel with the observation which indicated the operational domain knowledge as the main reason for the requirements changes. We observed that the contract documents do not provide sufficient information about the operational problem and the organizational processes of the customers. As the requirements definition phase is when systems engineers first meet the customer and try to understand the problem as defined in the customer requirements, they require significant customer support in this phase.

Case study results showed us that requirements phase provides an environment with dense customer interaction. It was expected that the systems engineers learn a lot from the customers about the operational problem such as business processes, interacting legacy systems, types of users etc. during this phase. However, it is found that the systems engineers mostly interact with the customers to clarify the ambiguities in the customer defined requirements.

Highlight3: The real problem in collaborating with the customer is not to reach the customer but to find and reach the customer with the required skills, experience and decision authority.

Highlight4: Systems engineers do not proactively plan and manage their interaction with the customer to increase the effectiveness of the collaboration.

Highlight5: Customer requirements are considered as the scope of the project and no extra effort is spent to understand the operational objectives and current limitations of the customer in the mission environment to reach those objectives. It is assumed that the customer requirements define the problem completely.

3.4 Non-functional Requirements

Late clarification/definition of the non-functional requirements is the other main reason for requirements changes. Non-functional requirements are known as constraints or quality requirements. They can be further classified according to whether they are performance requirements, maintainability requirements, safety requirements, reliability requirements, or one of many other types of system requirements [15].

Non-functional requirements are sometimes addressed under the specialty engineering title in the SE literature. INCOSE handbook defines 12 areas (Table 5) in the specialty engineering section. Our discussions regarding the problems with the non-functional requirements are based on these areas [16].

The case study results revealed that specialty engineering tasks are almost always performed by the SE group. However, the SE groups do not have a fixed, well-defined position for a specialty engineer. We have observed only 4 titles defined for specialty engineering within the 18 projects: 2 logistics analyst, 1 human engineer and 1 specialty engineer. All these positions are allocated to a single project (no matrix organization for specialty engineers). In the case study companies there is no separate group for specialty engineering. They are working within the SE group.

In the rest of the projects, the tasks required for specialty engineering is performed by the systems engineers. However, these systems engineers do not have the required training and experience to perform such tasks (summarized in Table 5). For example, among the 59 interviewees, only 2 of them had training on electromagnetic compatibility analysis and only 3 of them have past experience on this area. Similarly, only 1 systems engineer have previously performed training needs analysis. However, this systems engineer did not have any education/training for this task.

Table 5 Specialty Engineering Competencies

Specialty Engineering Analysis Area	Training	Experience	Specialty Engineering Analysis Area	Training	Experience
Cost effectiveness	-	-	Mass properties engineering	-	-
Electromagnetic compatibility	2	3	Safety & health hazard	1	3
Environmental impact	-	-	Sustainment engineering	-	-
Interoperability	-	-	Training needs	-	1
Lifecycle cost	-	-	Usability/human systems integration	-	-
Manufacturing & producibility	-	-	Value engineering	-	-

Most of the systems engineers find it relatively difficult to define non-functional requirements compared to defining functional requirements. This is the major reason for addressing the non-functional requirements very late during the project lifecycle. We observed that systems engineers do not like dealing with the non-functional aspects of the requirements. In all of the case study projects, the document templates that are used for the requirements and design artifacts have dedicated sections that are addressing the non-functional requirements. Our findings indicate that unless a non-functional requirement is defined directly by the customer in the stakeholder requirements document, systems engineers leave those document sections as TBD (to be determined) as long as possible.

Highlight6: Although most of the systems engineers do not have the required training and experience, non-functional requirements are considered in the responsibility of the SE teams.

Highlight7: As the systems engineers do not feel comfortable working on the non-functional requirements, definition/clarification of the requirements related to specialty engineering is delayed late in the project lifecycle.

3.5 Time of Requirements Change

Discussions on the time of the requirements changes provided us with interesting findings. The results of the case study interviews are summarized in Table 6. There are two things to note about the findings. Firstly, since the changes during the requirements phase is expected and mostly welcome without any side effects, the changes during the requirements phase is not discussed during the interviews. Secondly, in most of the projects the final customer acceptance is obtained before the maintenance phase starts. Therefore, the systems engineers we interviewed are not knowledgeable about the maintenance period as they have already started to work for other projects.

Table 6 Time of Requirements Change

#	Design	Development	Verification
Total	9	34	16
%	15	58	27

According to the results in Table 6, majority of the requirements changes occur after the design phase. Therefore, it is possible to interpret that most of the requirements changes have significant impacts as they occur late during the project. They result in rework both in the design and in the developed product.

A significant number of changes were expected in the verification phase. Since the final system is tested against the system requirements together with the end users, it is the first time that the users interact with the overall system. It is quite normal that the users discover what they really needed and what they did not like as a result of this interaction. On the other hand, large amount of changes in the development phase was an unexpected result for us. During this phase, system requirements and system design is already baselined and the system is realized by developing and integrating the subsystems. This is the phase that software, hardware and integration engineers are mostly on the scene.

After re-discussing the findings we found that keeping the software and hardware engineers (system developers) out of the decision making process during the requirements and design phases is the main reason. During the requirements phase where most of the fruitful customer interaction occurs and systems engineers try to understand the problem, software and hardware engineers are not consulted. As a result, software and hardware engineers try to understand the problem to be solved from the artifacts of the requirements and design phases.

As the system developers try to implement the abstract design and requirements into a physical and functioning system, they discover that significant amount of details for this transition is missing in the SE documents. Questions raised by the system developers against the systems engineers to understand the problem domain result in the discovery of new knowledge which in turn initiates changes to the requirements. Similarly, as the functioning system gets more visible, the systems engineers understand that this cannot be the feature/behavior that customer was asking for and asks for changes.

Although most of the changes occur during the development phase (Table 6), the results presented in Table 4 show that only %7 of the participants highlighted the need for customer collaboration during the development phase. During the discussions on this finding we have observed that when the systems engineers discovers a need for customer collaboration to bridge some knowledge gap, their approach towards the resolution of this problem changes from phase to phase.

If they discover that they need customer support during the requirements definition phase, they prefer to contact the customer without any hesitation. On the other hand during the design or development phases, their behavior is somewhat different. Case study results indicate that as the time passes and as decision point gets far and far from the requirements definition phase, systems engineers prefer the decisions that result in less rework and less changes in the previous artifacts. Especially, if the decision point is in the development phase they prefer not to consult the customer.

Most of the systems engineers state that they discover new knowledge related to the problem domain as a result of the interaction between the development teams and the SE team during the development phase. Most of the time, such new knowledge make them think that the requirements and design should change in a way to provide better solution with more customer satisfaction. However, they are reluctant to discuss such major changes with the customer since such changes are considered to have significant cost and schedule impact.

Table 7 summarizes how the change decisions are supported during different phases. As the decision point gets away from the requirements phase, systems engineers prefer making assumptions towards the option which requires less changes instead of asking for customers support and opinion. On the other hand, participants note that most of these decisions are rejected by the customer in later stages and had more serious cost consequences.

Table 7 Change Decisions vs. Phases

Method	Requirements	Design	Development
Make assumptions	5	23	30
Consult the customer	53	20	9
Execute decision support processes supported with the required analysis work	1	16	20

Although significant amount of the participants stated that they would execute decision support processes with the necessary analysis during the design and development phases, in practice only 4 of the 59 systems engineers have previously performed a formal analysis study with properly planned evaluation criteria and with properly documented analysis results.

Highlight8: Since the majority of the changes occur after the design phase, it can be interpreted that the costs of the changes are high.

Highlight9: The interactions between the systems engineers and the system developers during the development phase create a great environment for improving the understanding of the customer's problem and identifying the knowledge gaps in the requirements and design artifacts.

Highlight10: Nature of the traditional SE approach discourages the collaboration between the systems engineers and the customer/system developers.

3.6 Validation Process and Customer Focus

All the participants were theoretically aware of the difference between the verification and validation. They all referred to the widely known definition of verification and validation as: "*…While verification proves whether 'the system was done right'; validation proves whether 'the right system was done"* [17]. Validation is about user's operational needs and expectations [18]. Therefore, if there is a problem in understanding the true operational needs of the customer, then it is not possible to properly validate the system. Following are the highlights from the case study:

- %54 of the interviewees were performing requirements analysis activities in defining the system requirements from the stakeholder requirements. %46 were defining the system requirements from the stakeholder requirements based on their experience without performing any requirements analysis activities.
- Only %20 were performing requirements validation activities such as reviews, analysis, prototyping, etc.
- Only %10 were utilizing a requirements prioritization scheme and any of them did not use "value for customer" or stakeholder expectations as criteria. All criteria were chosen based on the developer's technical, budget and cost considerations.
- Verification & validation plans were not prepared in any one of the cases before the implementation started.
- %100 assume that the operational needs of the customer are documented in the stakeholder requirements (customer technical requirements provided with the contract) and a final system passing the tests that are defined against the stakeholder requirements in the operational environment results in a validated system.
- All agree that the technical review meetings (apart from the requirements review meeting) do not effectively provide customer feedback to systems engineers. Most of the time, customer's do not understand much from these technical review meetings.

Validated set of acquirer requirements are defined as a requirement/prerequisite for the end system validation in the ANSI/EIA 632 [19]. Therefore, in order to perform a proper validation process, one needs to perform requirements validation activities. Case study results indicate that requirements validation is not a well known concept and systems engineers do not perform this process properly.

Highlight11: Validation activities are poorly understood and weakly performed in practice.

Highlight12: Current SE practice is not customer and value focused. No special attention is given to understanding the specific needs of the stakeholders and their relative importance.

3.7 Project Documentation

Traditional SE approaches are also known as "document-driven" or "document-heavy" processes. Systems engineers defined different purposes for the usage of documentation:

- We capture and communicate the information via documentation.
- We are paid for the documentation.
- Documentation is required during the operation and maintenance of the system after delivery.
- Documentation is necessary if some of the engineers leave the project or the company.
- Documentation is used for future projects to help estimating and planning.
- Documentation helps us to prevent/control changes.
- Progress is monitored by the completion of the documents.

Documentation we refer in this section is the technical documentation such as requirements, design and interface documents. The user documentation like the manuals is not discussed during the case study.

We observed that producing a lot of documentation is not the main agility problem of the traditional SE processes. Every piece of information in the documents somehow reflects a decision. Systems engineers need to deliver the documents with the required decisions within a pre-determined timeframe ended at a project milestone. In this approach they are expected to make the decisions when the necessary information is missing. The rest of the design and development activities (and the associated documentation) are based on such ill-made decisions. Systems engineers state that most of their time is spent on changing the decisions when the required information is available and re-working on the documents in the later phases of the projects.

Findings also indicate that the existing processes are weak in measuring the quality and completeness of the documents. Systems engineers use most of their time to write further requirements on the problem domain that they are familiar with. In such an approach while the details of the well-known areas increase with a lot of redundant requirements, the unknown space remains to be unknown. Time is used to increase the thickness of the documents. However, knowledge creation and hence the value provided to the systems engineers is not directly proportional with the thickness of the documents.

Highlight13: Current document-oriented SE practice forces systems engineers to make decisions when the necessary information is not mature or available. This causes serious rework and lack of team's ownership.

Highlight14: Current SE practice focuses on document generation and is weak in creating value for the systems engineers. In such an approach it is not easy to control whether the processes are redundantly capturing the already known information or new and valuable knowledge is being created. Such an approach makes it difficult to monitor the progress in the project.

4 Discussion

4.1 Knowledge-Based Model for Change Initiation

Our discussions with the systems engineers to explore the reasons and nature of changes in the system projects resulted in a simple model (Fig. 1) which relates changes with the discovery of new knowledge. Our model is based mainly on the categorization of the knowledge as described in Donald Rumsfeld's famous statement: *"…as we know, there are known knowns; there are things we know we know. We also know there are known unknowns; that is to say we know there are some things we do not know. But there are also unknown unknowns – the ones we don't know we don't know."* Although they are different concepts, in this paper knowledge and information is used interchangeably.

As Donald Rumsfeld has defined, there are three categories of knowledge: Known knowns (KKs), known unknowns (KUs) and unknown unknowns (UUs). KKs represent our knowledge in relation to the project. Every piece of information that we have and we are planning to use (i.e. valuable information) during the system development is in this category. Our technical experience, our operational domain knowledge, information captured in the initial project documentation, etc.

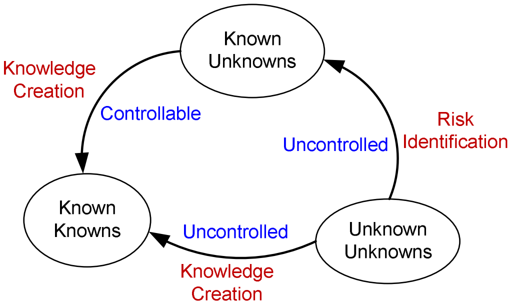


Fig. 1 Knowledge-based change initiation model

KUs are the missing information that we need. This is defined according to our understanding of the initial problem. For example, we know that we need to know all the user types and their operational responsibilities in order to solve their operational problem. However, initial project documents do not include such information. Therefore, this missing information is identified (i.e. known) as necessary but we don't have it yet (i.e. unknown at the moment).

UUs are the missing information that is required for the solution development but we are not aware of the fact that this piece of information exists and is necessary. This type of knowledge remains completely in the unknown space.

During the case study it's agreed with the participants that every change request is a result of new knowledge (i.e. new KKs). Of course not necessarily that every new information triggers changes. According to our model there are two ways to create new knowledge. The first is a result of the transition of some KUs into KKs. We acquire new information which was already identified as valuable but missing. The second is a result of the transition of some UUs into KKs. Some new information is discovered and at the same time we discover that this new information affects the project. In both transitions, new valuable information is available and it is expected that this new information will trigger changes on the previously made decisions.

There is one other transition in the model which is from the UUs to the KUs. This occurs when we discover that we need some information which will affect our decisions. We discover the need for that information but we don't have it yet. This is the transition that systems engineers are supposed to define risks.

4.2 Flexibility with Focus on Learning

The key characteristic of agile approaches is flexibility. This requires dealing with expected and unexpected changes. In today's system projects, trying to eliminate changes is not realistic and no more a valid option, then what we need is to reduce the cost of responding to changes [2].

Case study results showed us that most of the change reasons can be associated to missing knowledge (unknowns) such as operational domain knowledge, knowledge on the legacy systems and knowledge on the COTS/outsourced products, etc. In addition to this, participants categorize this missing information as KUs. For example, all the systems engineers know that they will need information on all types of users and the business processes they are involved (HL1). They all know that their system design will be affected by the legacy systems that are in use. They all know that they need to know about the security, interoperability, safety, etc. needs of the customer (HL6). Traditional ConOps templates and requirements templates all address such information. However, current linear, document oriented practice does not focus on getting or creating the required knowledge (HL2, HL4). Most of the new knowledge comes uncontrolled and hence comes late in the project. Therefore, the changes initiated by such late and uncontrolled knowledge have costly consequences (HL8).

According to our model, all the transitions from the UUs are uncontrollable. However, transitions from the KUs to KKs can be controlled. By identifying the valuable but missing knowledge (i.e. KUs) early in the project and by defining/tailoring the SE processes to focus on getting/creating the required knowledge can shift these transitions to early phases and thus reduce the costs of changes. Such an approach will trigger learning and knowledge focused SE and will also be flexible from the agility point of view.

4.3 Focus on Customer Value and Lean Behavior

The agility attributes "focus on customer value" and "lean attitude" are closely related to each other. Lean attitude is about removing non-value added activities and delaying decisions as much as possible [5]. Lean thinking defines three conditions for value added activities [20] [21]:

- The external customer is willing to pay for it explicitly or implicitly,
- Transforms information or material, or reduce uncertainty,
- Provides specified performance right the first time.

In order to develop something valuable for the customer we need to understand the customer's real problem. Systems engineers state that they need problem domain information to understand the real operational problem but they are not consulting the necessary processes for this need (HL1, HL12). Difficulty in understanding the real problem and real value for the customer makes it also difficult for the systems engineers to properly define and perform the validation activities (HL11). Although the systems engineers agree and are very well aware of the fact that the customers do not know what they need at the beginning of the project and the system requirements derived from these customer requirements do not reflect the true and complete operational need of the customer, they still try to develop a system based on the ill-defined customer/system requirements (HL5).

In real life, operational problems are mostly non-functional oriented rather than functionally oriented. Whether a software system will satisfy a customer will mostly be determined by its quality attributes but not its functions. Every competing system will more or less provide similar functionality [22]. The key characteristics of the successful systems differentiating them from the others are their non-functional requirements. A system's utility and effectiveness will be determined by its non-functional requirements.

Main distinguishing characteristics of SE is its interdisciplinary nature. Systems engineers are not expected to be experts in all of the areas related to whole system lifecycle. However, it is the main contribution of the systems engineers to coordinate the activities of different disciplines for the success of the overall system. Coherent integration of the specialty engineering into the project/program at the right time is one of the responsibilities of the systems engineers [23]. Developing specialty engineering competencies in the SE teams and changing the current processes in a way to integrate the specialty engineering efforts early in the lifecycle is necessary for developing valuable systems (HL6, HL7, HL12).

The value for the systems engineers is the timely and accurate knowledge. However, with the current document-oriented SE practices they cannot focus on the activities that create/transform knowledge to reduce uncertainty. In contrast, current SE practices encourage ignoring the valuable knowledge in decision making since this knowledge comes uncontrolled and late (HL8). In addition to this, linear and document-driven approach forces systems engineers to make decisions early in the project lifecycle when the necessary information is missing. This is in contrast to the lean behavior which supports delaying decisions as much as possible (HL13).

Instead of focusing on the documentation and document-based processes, systems engineers should focus on their own needs. In such an approach, the side effects of the document heavy approach will be reduced and team ownership will be increased. Systems engineers will find the opportunity to plan and perform tasks to reach what they really need instead of trying to fill out the document templates with poorly made decisions (HL14).

4.4 Collaborative Systems Engineering and Trainings

In this case study we have learnt that traditional, document-oriented, linear SE approach does not emphasize and support the collaboration and interaction both internally and externally. Therefore, agile SE approach should be designed to support collaboration of the systems engineers with the customer, with the internal system developers and with the external parties (COTS companies and subcontractors). HL1, HL2, HL3, HL4, HL5, HL9 and HL10 can all be associated with the problem of timely and effective collaboration.

Increased understanding of the initial operational problem by both the customer's team and the supplier's team is found to be the primary reason for changes (HL1). We propose that trainings can help systems engineers on this problem. As the users get more familiar with the concept and the technical capabilities they can address their problems better. In the current practice, trainings are used to teach the users and maintainers how to operate and sustain the delivered system. That is why the trainings are not planned until almost all the features of the system are designed and implemented. This is always considered as something valuable to the customer but a non-value added activity from the supplier's point of view during the development.

Considering the knowledge-based change model and the other findings of this case study, we believe that trainings can be used to achieve several agility attributes by starting the training sessions very early in the project and conducting in a periodic manner. The objectives of the trainings will be defined in a large spectrum such as SE and system lifecycle training, concept of operations training, user interface training, COTS capabilities trainings, etc. In addition, the trainings should not be limited to the customer but also many other stakeholders. We expect several benefits from such a training dense SE approach:

- Effective knowledge sharing/creating environment and reducing the paper based engineering efforts,
- It is possible to increase the concept and technology awareness of the customer so that the customer can discover his operational needs early in the project. Consequently, change requests will occur early in the project with relatively reduced costs.
- It will be possible to interact with different types of customer representatives with the required skill set and experience. Customers will not see this collaboration

activity as a burden because this is a training session and they will benefit from this.

- It will be possible to give the context and system view to the subcontractors and obtain early feedback about their limitations.
- Better understanding of customer value.
- Early feedback from the systems developers regarding the feasible technical options and innovative improvement suggestions.

5 Threats to Validity

There are four widely accepted categories of validity related to qualitative research: construct validity, internal validity, external validity and reliability [24]. Internal validity is relevant for the explanatory case studies [25]. Construct validity reflects how much the findings represent what the researcher is really investigating. Researcher's subjectivity is always a threat. The results may be misleading if the interview questions are misunderstood or misinterpreted by the participants. We used both written questionnaires and interviews for data collection. By selecting the systems engineers with more than 5 years SE experience we tried to minimize the possibility of the misunderstanding in the concepts used in the interviews. In addition to this, we explained every key concept using the definitions and examples from the well known IEEE and INCOSE SE literature. As we discovered some ambiguity we iteratively re-discussed the findings with the participants. As suggested by [26], all the findings and our interpretations were reviewed by the participants and corrected if necessary according to their feedback.

External validity determines whether the findings can be generalized beyond the setting of the group studied [24]. In order to increase the external validity of our study we carefully planned and selected multiple case interview set. We have conducted 59 interviews. Interviewees were selected from 10 different companies and from 18 different projects. Selected companies are mid-to-large scale companies from around the world (4 from Turkey, 3 from Europe, 2 from USA and 1 from Australia). At least 3 systems engineers and at most 4 systems engineers are selected from each project. All projects are large scale software intensive defense projects. It is argued in the case study literature that cross-case analysis involving 4 to 10 case studies may provide a good basis for analytical generalization [27]. Yin also states that these case studies can be from the same organization [26]. Since we have worked on 18 different cases from 10 different organizations, the findings are expected to be generalizable to other cases.

We tried to increase the reliability of findings by both using an interview protocol based on a template questionnaire and open interviews as data sources. On the other hand, due to the nature of the military projects we unfortunately did not have a chance to review the documents of the projects or make observations in the projects. For example, the results would be more realistic and reliable if we could have a chance to review the change request forms of the projects.

6 Conclusions

Identifying the need for agile SE is not new. However, studies in this area are strongly affected by the agile software methods and mostly present the effort to adopt agile software engineering methodologies in the agile SE area. It can no longer be an excuse for systems engineers to just say that agile is not for large scale complex system projects, but it is for small scale -purely- software projects. The real problem is not the adoption of the current agile methods in large scale system projects, or it is not the problem of tailoring the light weighted software engineering methods to the SE. The real problem is that we need agility in SE, we need agile SE processes and methodologies to be applied in complex, large scale, software intensive systems of systems projects.

Based on this motivation we started a research to develop a novel SE approach with improved agility attributes. In this initial study, we performed an exploratory case study to explore the practical problems associated with the agility attributes of the traditional SE approach in defense projects. In an attempt to understand the nature and causes of changes in the projects, we proposed a knowledge-based change model. The results of the case study showed us that traditional SE in practice is mostly process and document focused and ignoring the real needs of both the customer and the systems engineers themselves. The findings will be very valuable in the next step while we are trying to address the agility problems in the traditional SE approach.

References

[1] Brooks, F.P.: No Silver Bullet. IEEE Computer 20(4), 10–19 (1987)

[2] Highsmith, J., Cockburn, A.: Agile software development: the business of innovation, vol. 34(9), pp. 120–127. IEEE (2001)

[3] Gilb, T.: Competitive Engineering. In: Competitive Engineering. Elsevier (2005)

[4] Boehm, B.: Some future trends and implications for systems and software engineering processes. Systems Engineering 9(1), 1–19 (2006)

[5] Turner, R.: Toward Agile Systems Engineering Processes. CrossTalk the Journal of Defense Software Engineering, 11–15 (April 2007)

[6] Kennedy, M.R., Umphress, D.A., Ph, D.: An Agile Systems Engineering Process the Missing Link? CrossTalk the Journal of Defense Software Engineering 4(3), 16–20 (2011)

[7] Bosch, J., Bosch-Sijtsema, P.: From integration to composition: On the impact of software product lines, global development and ecosystems. Journal of Systems and Software 83(1), 67–76 (2010)

[8] One Hundred Eleventh Congress of the United States of America (2010)

[9] Koskela, L., Howell, G.: The underlying theory of project management is obsolete. IEEE Engineering Management Review 36(2), 22–34 (2002)

[10] PMI Agile Certified Practitioner (PMI-ACP), http://www.pmi.org/en/Certification/New-PMI-Agile-Certification.aspx (accessed November 7, 2011)

[11] The Standish Group International, CHAOS Summary 2009, The Standish Group (2009)

[12] Asan, E., Bilgen, S.: Agile collaborative systems engineering- motivation for a novel approach to systems engineering. In: INCOSE 2012 Symposium (2012)

[13] Qumer, A., Henderson-Sellers, B.: Crystallization of agility back to basics. In: ICSOFT, pp. 121–126 (2006)

[14] Kaplan, B., Duchon, D.: Combining Qualitative and Quantitative Methods in Information Systems Research: A Case Study. MIS Quarterly 12(4), 571–586 (1988)

[15] Abran, A., Moore, J.W., Bourque, P., Dupuis, R., Tripp, L.L.: Guide to the Software Engineering Body of Knowledge, vol. 19759, p. 204. IEEE (2004)

[16] Haskins, C., Forsberg, K., Krueger, M., Walden, D., Hamelin, R.D.: Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities, 3.2 edn., vol. 3.2. INCOSE (2010)

[17] NASA, NASA Systems Engineering Handbook. In: Systems Engineering, vol. 6105, p. 360 (June 2007)

[18] Wasson, C.S.: System Analysis, Design, and Development: Concepts, Principles, and Practices, vol. 22, p. 832. Wiley-Interscience (2005)

[19] ANSI/EIA-632 Standard, Processes for Engineering a System (1998)

[20] Womack, J.P., Jones, D.T.: Lean thinking: banish waste and create wealth in your corporation, vol. 4(1), p. 384. Free Press (2003)

[21] Oppenheim, B.W., Murman, E.M., Secor, D.A.: Lean for systems engineering with lean enablers for systems engineering. Wley Series in Systems Engineering and Management (September 2011)

[22] Chung, L., do Prado Leite, J.C.S.: On Non-Functional Requirements in Software Engineering. In: Borgida, A.T., Chaudhri, V.K., Giorgini, P., Yu, E.S. (eds.) Conceptual Modeling: Foundations and Applications. LNCS, vol. 5600, pp. 363–379. Springer, Heidelberg (2009)

[23] Systems Engineering Competencies Framework (2010)

[24] Runeson, P., Höst, M.: Guidelines for conducting and reporting case study research in software engineering. Empirical Software Engineering 14(2), 131–164 (2008)

[25] Tellis, W.: Introduction to Case Study. The Qualitative Report 3(2), 1–11 (1997)

[26] Yin, R.K.: Case study research - design and methods, vol. (5), p. 200. SAGE Publications (2003)

[27] Eisenhardt, K.M.: Building Theories from Case Study Research. Academy of Management Review 14(4), 532–550 (1989)

Chapter 5
Modeling Transportation Systems: A Case Study with the Open Method Praxeme

Dominique Vauquier

Abstract. This paper is based on the outcome of a research project developed at the RATP Group (the world's fifth largest public transport company). This ongoing research project aims to provide RATP with a proper methodology that will help it to cope with complexity and to address the challenges of the future market (competition, large programs...). The company has chosen the open method Praxeme and was willing to check whether it could apply to the realm of transportation. Praxeme proposes an efficient architecture framework that links the aspects of the Enterprise System together. This framework lays the groundwork for a comprehensive approach to socio-technical systems; it strongly ties the various models together moving progressively from clarified business knowledge to concrete solutions. A very specific case study was used to demonstrate the value of a modeling approach in the eyes of the decision-makers. The approach proved able to oversee issues in the running of the system.

Keywords: transportation, modeling, UML, methodology, Praxeme, system.

Introduction

The RATP Group is the world's fifth largest public transport company operating all modes of collective mobility – bus, metro, trains and trams. In Île-de-France it runs, maintains, and develops one of the world's densest multimodal networks. Every day it transports over 10 million people. In addition to its traditional role in and around Paris, RATP and its subsidiaries export this expertise across all continents. The offer relies on transport systems that combine numerous elements developed by many industrial providers. Any failed element may cause the failure of the entire system and damage the service, with an impact on the comfort and satisfaction of tens of thousands of clients.

Dominique Vauquier
PRAXEME Institute (A not-for-profit association whose purpose is to develop and promote the open method Praxeme)
21, chemin des Sapins
NOISY LE GRAND 93100
France

In the face of aging lines and upcoming requests for proposals, the "*Maîtrise d'ouvrage des transports*" department, in charge of running the lines, launched a research project to investigate the field of methodology, in quest of a method that would allow it to improve its activity and its relationships with its providers. The challenge is to establish new practices in order to tame the complexity of transportation systems.

The first step was to explain the principles and to adapt the Praxeme methodology and terminology to the context. Praxeme is an open method that results from an initiative supported by the French administration and many public and private organizations (the French army, SAGEM, CNAF, AXA Group…). It offers a comprehensive approach that covers every aspect of the enterprise. In fact, changes to the method have been minor as the people involved have been willing to stick to the open method as much as possible. We only had to change one dependency in the architecture framework, also referred to as the Enterprise System Topology[1]. Nevertheless, this was a significant change, which marks the adjustment of the methodology to physical systems, in addition to its applications to information systems and human organizations.

Then, a proof of concept was used to confirm the validity of the approach: this will be the case study detailed hereafter. The challenge consisted in demonstrating that, had the method been used, some issues encountered in the current systems could have been anticipated and avoided. At stake are:

- the quality of service,
- cost-cutting thanks to a more rigorous approach to specifications,
- renewing the relationships between the system owners, the engineering department and the industrial providers.

After a brief review of the difficulties encountered, we will set forth the general answers provided by the Praxeme method. We will then look firstly at the findings from a system level and secondly from the detailed case-study level. The case study was chosen by the project sponsors in order to assess the validity and applicability of the method.

1 Difficulties Encountered When Addressing Transport Systems

1.1 Complexity or Complication

Transport systems are said to be complex systems. This is not something we will dispute. That's right, blame complexity! However, a great deal of the difficulties do not stem from the system itself but from the human beings involved and their lack of rigor. Vocabulary – a loose, ambiguous and unstable terminology – arises

[1] For a presentation of the Enterprise System Topology, see "Enterprise Methodology: an Approach to Multisystems", in Complex Systems Design & Management, 2010 or the white paper and the general guide available on the Praxeme Institute's website.

as the first obstacle on our path to understanding and designing the system. We were able to collect several glossaries but we found that defining terms was not an easy task.

Moreover, it proved very difficult to build a picture of the systems that were abstract enough from the very details of the current implementations. A lot of functions are named after the solutions provided by the industry. As solutions vary from one line to another and even more from one means of transport to another (metro, bus, tram), building a common and generic representation seemed out of reach, and some participants simply dismissed this target. Even the boundaries of the subsystems did not resist to thorough examination, despite the creed that this exercise is easier when it comes to physical systems as opposed to organizations. Habits and preconceived ideas hinder the reflection and prevent us from seeing the fundamental simplicity behind the accumulated complications.

1.2 The Curse of Requirements

As in most places within the industry sector, the description of the systems is made up of an almost overwhelming number of requirements. This description suffers from the usual flaws:

- The requirements are badly formulated, using an ambiguous vocabulary and relying on assumptions that are not always clarified.
- They are poorly managed, as they exist in the form of documents. A given requirement may appear in several places within the entire documentation and may also refer to several other document elements.
- Taken at the overall level of the transport network, the documentation shows a huge redundancy rate, due to the absence of an overall endeavor to arrange it in a proper structure that would embrace all lines and transport types. Redundancy raises the risk of discrepancy, which, in turn, translates into additional costs and operational risks.

To take an example, ergonomics entails hundreds of requirement elements. Not only are these duplicated but the prescriptions vary from one line to another, including items such as the color code or button shapes. This leads to increased training efforts and costs when a driver moves from one line to another; it also multiplies the investment and maintenance of software interfaces.

In theory, the system owner is the sole authority on the functional specifications which are supposed to perfectly reflect the reality of the system, at least from a functional perspective. In practice, things go slightly differently. Firstly, the functional nature of the specifications is hardly preserved at all, since the physical systems tend to be perceived through their current solutions and terminology rather than in terms of functions. Writing pure functional specifications demands an effort of abstraction and a specific sense of concept over percept. Secondly, even though the system owner develops a first version of the functional specifications and engages his or her own responsibility, the industry provider responds with another document. This document, called a technical specification, partly rephrases the requirements in a more “concrete” way, and partly complements them with additional and technology-oriented requirements. Afterward, the technical specifications will serve as sole reference, replacing the initial functional specifications. The latter do not evolve anyway and are rarely updated because they are seen as less detailed and less accurate than the former. The end result is a functional repository that lacks consistency, accuracy and relevance. Instead, the only valid description of the system – at least in its current state – is the technical one, in the hands of the industrial provider. This says a lot, as far as autonomy and control are concerned.

After a while, the only document which all players, including the system owner, refer to is the technical document. This tends to bias the understanding and definition of the system. Obviously, it hinders new design and slows innovation down, as will be demonstrated in the last part of this paper.

To a certain extent, quantity harms quality, especially when the structure of the corpus has nothing to do with the inner structure of the reality observed. This happens with the traditional approach based on documents or spreadsheets, the very structure of which is detrimental to the natural organization of any system. The mismatch of the description feeds the complication of the design, which augments the complexity of the system.

2 Initial Answers by the Praxeme Method

Praxeme presents itself as an enterprise methodology, which means that it aims to cover every aspect of the enterprise, from strategy to deployment. Indeed, Praxeme insists on the need for linking together all specialties and disciplines that contribute to think, design and transform the systems. By "enterprise", we mean any kind of organized and willful entity or action. By "Enterprise System", we assert the rational approach to the enterprise and the use of the intellectual tools we find in the system theory[2].

Praxeme provides the practitioners (strategists, organization designers, architects, modelers…) with a set of techniques, split over its framework. There are numerous elements to be considered and decisions to be made about the Enterprise System. The framework – the Enterprise System Topology – distributes these elements and decisions throughout well-connected aspects. Here, the notion of aspect is key. It must not be confused with either view or layer[3]. The aspects are supposed to be embedded in the very nature of any system. This tenet – quite metaphysical – sets the stage for the process of analysis and design, at every level and in every dimension, maintaining the natural articulations between the artifacts produced. Thanks to this foundation, Praxeme can easily link the disciplines together into an interdisciplinary approach to complex systems.

[2] The Enterprise Transformation Manifesto simply defines the "Enterprise System" as "the enterprise that perceives itself as a system" (see on `www.enterprisetransformationmanifesto.org`).

[3] Both notions are part of the method, too, but play a secondary role. As opposed to a view (see IEEE Std 1471-2000), an aspect is not a representation but a certain portion of reality to be considered.

Going back to our research project, this foundation was adopted from the very start. It showed itself in the deliverables that were produced, which are presented hereafter under the headings named after some of the aspects that were identified. As usual, the company does not own a proper representation that captures its core knowledge. This leads to confusion and a waste of time and energy, whenever people have to gather and address issues. The effect is worsened when representatives from several organization units, carrying various cognitive universes are called upon. To tackle this situation, Praxeme proposes two kinds of techniques: terminology and semantic modeling.

2.1 The Intentional Aspect

The intentional aspect of the Enterprise System is made up of objectives (from strategic objectives to operational objectives), terms, requirements, rules, performance indicators. In short, it collects all expressions that cannot be referred to as models, due to the simplicity of their underlying approach. However, these expressions bring value insofar as they convey the fundamental will of the system builders and users. Among these expressions can be found the ultimate goal of the system itself and its ensuing objectives and feature requirements.

One of the first tasks of any project is to clarify the terminology used. To this end, the method recommends collecting available glossaries and developing a thesaurus. The terminological work shows two facets: on the one hand, it passively collects the terms used and analyzes their usages; on the other hand, it elaborates a canonical vocabulary, cleaned from ambiguity and polysemy as much as possible[4]. In so doing, the terminologist paves the way for the modeler. Both disciplines are connected and can help each other when it comes to formulating good definitions. The meaning of a specific term results from its place in the network of terms. Precisely, the best network of terms is the semantic model. Some vague concepts can only receive a proper definition when deduced from the model.

Once the canonical dictionary is available – and ideally confirmed through the semantic model – we can turn to the requirements and review their formulation. Ideally, every term of a requirement expression should have a corresponding item in the canonical dictionary. This is not enough: the sentences should obey syntactical rules that have been thought through to avoid confusion and to detect all possible options.

These tasks can only be conducted with appropriate tooling. There are many tools available on

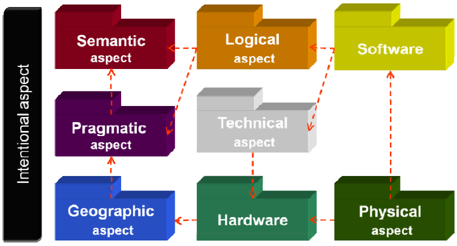


Fig. 1 The Enterprise System Topology (in the form of a UML diagram)

[4] Incidentally, terminology is a scientific discipline, a branch of linguistics, and it can benefit from a couple of ISO standards.

the market. It is important to consider one major requirement when selecting such tools: the need for dictionary and requirement management tools to smoothly interface with modeling tools. Indeed, we will have to link requirements and terms to modeling elements. The resulting traceability chains provide us with the mechanism to justify the modeling decisions and to check the requirement coverage by the model.

2.2 The Semantic Aspect

Semantics isolate the core business knowledge, that is: knowledge regardless of the way players operate and equipment supports the operations. So, a semantic model is the most abstract of all models, focusing on real objects and concepts and expelling any element that comes from organizational choices or technical solutions. We could think of several means to formally express this knowledge. In our case, we used the semantic modeling technique proposed by Praxeme, which harnesses the object-oriented approach. Alternative techniques include ontologies and formal notations.

We can find some models available on the market, in the field of transportation,[5] as well as in other sectors[6]. Unfortunately none of these models can be deemed a fully-fledged semantic model. They are essentially data models, sometimes conceptual, most of the time logical, and always far from assuming all the dimensions and reach of business knowledge. Consequently, they can help as input but we still need to undertake the modeling endeavor that captures and clarifies the business knowledge. If we fail to do so, we will not reap the benefits of thorough examination and any investment will not yield its full potential.

3 System, Where Are You?

3.1 Everything Is Not a System

Professionals in the field of transportation commonly use the term "system", as does everyone else. There is even a norm[7] that lists the sixteen systems composing a transport system. In these usages, the term "system" does not convey the meaning it has inside engineering and science. It is worth spending some time on this issue because it fuels misunderstanding and makes it more difficult to apply a system approach to the domain of study.

Let us make the assumption that "system" – as our core intellectual tool – is a term and notion that has to be taken care of. A common and widespread usage of the term tends to weaken the notion and to obliterate its real and operative meaning. When the norm speaks of sixteen so-called systems and people get

[5] See Transmodel on `www.transmodel.org`

[6] An example is given by the ACORD Framework in the insurance sector. See `www.acorg.org`

[7] Norms issued by the SRTMG (*Service Technique des Remontées Mécaniques et des Transports Guidés*).

accustomed to seeing these systems in their reality (or, better said, to see reality through these systems), it becomes improbable to exert a systemic approach unless we start raising awareness. At some point, the normative list was coined a "taxonomy" to say what it really was: a list of things we have to deal with. This decision freed the imagination: we were then able to think the system anew, in accordance with the true definition of system. Figure 2 shows how the systems have been put together to form consistent blocks, according to the norm (S1, S2, etc.), in preparation for identifying systems with specified interfaces.

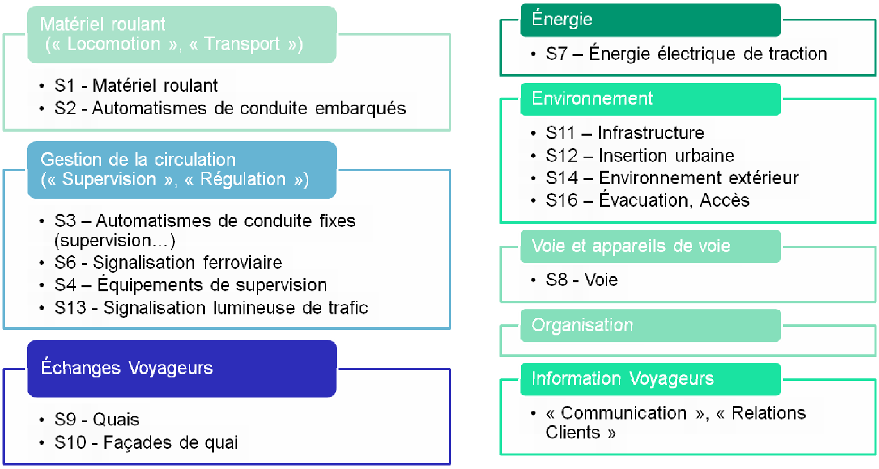

Matériel roulant
(« Locomotion », « Transport »)
• S1 - Matériel roulant
• S2 - Automatismes de conduite embarqués
Gestion de la circulation
(« Supervision », « Régulation »)
• S3 – Automatismes de conduite fixes (supervision…)
• S6 - Signalisation ferroviaire
• S4 – Équipements de supervision
• S13 - Signalisation lumineuse de trafic
Échanges Voyageurs
• S9 - Quais
• S10 - Façades de quai
Énergie
• S7 – Énergie électrique de traction
Environnement
• S11 – Infrastructure
• S12 – Insertion urbaine
• S14 – Environnement extérieur
• S16 – Évacuation, Accès
Voie et appareils de voie
• S8 - Voie
Organisation
Information Voyageurs
• « Communication », « Relations Clients »

Fig. 2. The systems according to the norm, rearranged as blocks

The system "S5" covers all communication means. It has been removed from this figure, since the list represents a step toward the logical architecture. Precisely, a logical architecture aims to reveal the interactions among systems and subsystems. Had we introduced the communication system, every system would have been connected to it and this would have masked the dialogues among the systems. This is why the Praxeme method suggests expelling the "middleware" from the logical architecture. Such an architecture endeavor assumes that there will be a solution to connect one point to another at the physical level. The appropriate solution has to be detailed and specified through other kinds of architectures (technical architecture).

3.2 Think the System Anew

Let us take a transport system (in our project, it was a given underground line). Our task is to build a representation that is both accurate and efficient: to be a good-enough model, it has to express a maximum number of things with a minimum of terms. Prior to any effort, we have to set aside – or even better discard totally – the usual representations we may have (like the norm discussed above).

The best way is to adopt a naïve approach, so as to get rid of preconceived ideas. We identify the mission of the system and its main functions.

The choice of terms is of paramount importance and adopting a renewed vocabulary can be a prerequisite to creativity. For example, if we start saying "drive", we imply that there must be a driver, thus restricting our design to the manual mode. This is why we favor the more neutral and more open term "transport", which is closer to the core mission of the system and leaves the options for design open.

Of course, we can always challenge the answer. Why do we have transport? There is always an opportunity for thinking out of the box. Indeed, why do so many people have to commute on a daily basis, generating pollution, wasting time and energy and transforming cities into hell holes? This is a relevant question, but out of the scope of our project and out of the remit of the department involved as well. This is a question for political decision-makers, if they are able to broaden the topic of transportation to consider the much larger system: society itself, with all its dimensions (working hours, organization modes, remote work, culture, etc.).

Words play a huge role here in the following design process and the methodology recommends changing the wording in order to free our imagination. Most of the time, our understanding remains stuck to the current state of things. Using the usual, traditional, technical vocabulary reinforces this alienation. This is why we proposed that the RATP adopts new terms, evoking basic functions, for example: "locomotion system" instead of "rolling material"; "regulator" instead of "command-control station".

3.3 The Logical Aspect: Representation of Systems

In the previous chapter, we introduced the intentional and the semantic aspects, respectively the realm of will and of knowledge. The notion of system as a modeling tool emerges from the logical aspect. This aspect plays a very specific role in the method. It is an intermediary aspect between the business reality (business knowledge, organization, business activity) and the artificial solutions we may build (technology, logistics…). As an intermediary aspect, it leaves us free to choose the mindset we apply to it, starting with an appropriate metaphor. The use of metaphor to address this aspect is not meaningless: urbanization for the information system, service-oriented architecture for the IT system, functional design, agents, events… We choose our vocabulary because we stand in a situation where we assume the role of designer and creator. Therefore, it is quite natural to resort to the system notion and theory: it is our conscious choice, exactly as a worker selects his/her tool from a toolbox[8].

The whole system theory is not sufficient to act effectively: we also need to equip ourselves with a notation that will enforce the operative notions. Here, Praxeme proposes a pragmatic option, providing that its main criterion is the ability to

[8] This is not to say that system as a notion may apply only to logical aspect. We can even imagine an approach to an organization that analyzes the organizational unit in terms of systems, meaning real systems in the sense of our definition above.

connect the models of every aspect according to the Enterprise System Topology schema. This basic solution is UML[9], with the advantages of an international standard and a broad offer for tooling. UML has no notion of system but it is easy to extend it so that we use it to represent systems in a rigorous manner[10]. When we want to enact a systemic approach via this tooling, we need some key notions that include: visibility, interface, interaction, flow. We also need to establish a clear relation between:

- the systems and subsystem of the logical aspect,
- the classes of the semantic aspect and the roles and use cases of the pragmatic aspect[11],
- the logical constituents and the physical solutions.

The notation bears stringent rules. The main one: a system can interact with another system only via a specified interface. The dialogues can only convey things that are specified in the interfaces (information, objects, services, signals…).

Figure 3 epitomizes the use of the notation. It shows four systems with some of their interfaces (provided or required). Two systems may be connected if, and only if, they expose the same interface (or compatible interfaces) required for one of them (in this example “Pilotage” for Slq_Regulator) and provided for by the other (Sql_Locomotor). The figure also displays the representation of an interface (in the top left corner) with some of its properties. These properties will be the only elements to be used in the interactions between the systems. This calls for the modeler to explicitly describe every possible interaction and locate them on the model.

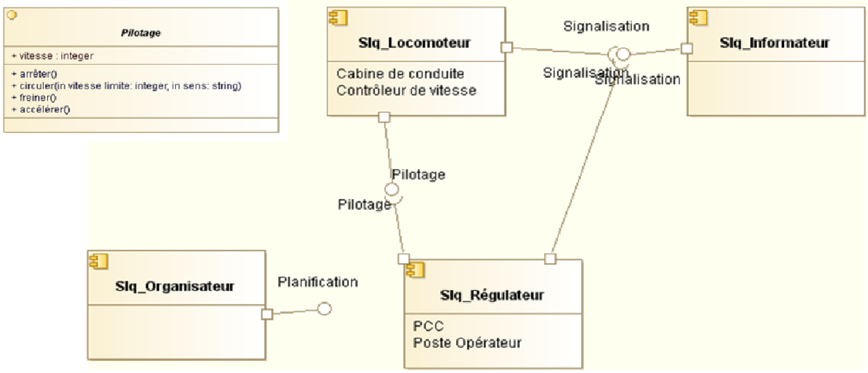


Fig. 3 Representation of systems and their connections (example)

[9] Unified Modeling Language, a standard issued by the Object Management Group.

[10] There is another option: SysML, which is built upon UML and is dedicated to system modeling.

[11] While the semantic aspect deals with business and real objects, the pragmatic aspect deals with activity (from processes to use cases) and actors (roles, organization…).

3.4 The Human Being and the System Thing

One of the findings when designing the transport system from a logical architecture perspective has been that, sometimes, we ought to put the players inside the system, even when it is an artificial system. Doing otherwise would impair our ability to design systems in accordance with the encapsulation principle. This is particularly obvious in the case of a vehicle. The company handles two transportation modes: traditional, with a driver onboard and automatic (e.g., line 14 of the Paris metro). In some cases, both modes are mixed (e.g., line 1). From the traffic regulator's perspective, the mobile system should be described the same way, whichever the mode. When we are not able to do so, we increase the complexity of the representation and we fail in our bid to construct a generic and convergent architecture. Therefore, the mobile system exposes an interface with all required services to steer the vehicle. These services are to be interpreted from inside the system, depending on the solution implemented (manual or automated). In so doing, we preserve the genericity of the architecture; the regulator system can be designed in such a way that it can cover every case. Had the command-control station been designed following this rule, the transition of the underground service from manual to automated mode would have been facilitated.

Some people may feel uncomfortable with this idea that the human actor is "lowered" and reduced to the level of a component.

Here, we are dealing with the logical aspect, and we need to put all kinds of resources together to make the system run. If we study an artificial system as being separate from the human system, we miss the true meaning and complexity of the system. This is why the method recommends modeling human beings among the material artifacts, thereby anticipating that the former could revert to the latter from time to time. It is the only way to comply with the encapsulation principle and, thus, to simplify the design.

3.5 The System Boundaries: Between Arbitrary and Arbitration

A critical question when designing systems is one about the breakdown structure. A spontaneous and unchallenged representation is not always the best one. Let us take the example of the track. The semantic model attaches many properties to the Track class, notably its states: occupied, free… These features certainly belong to the semantics of a track. It is a no brainer, as far as semantic modeling is concerned. But when it comes to logical modeling and to delineating the subsystems, how should we proceed? On the one hand, the tracks are part of the infrastructure, which is a separate subsystem (with a notion of dedicated accountability). On the other hand, the state of a track is checked and displayed thanks to signals, which pertain to the means of signaling. There is a great temptation to create a "signaling" subsystem, inasmuch as it reflects the human organization (there is a department devoted to this topic). Choosing this spontaneous option leads to breaking the semantics of a track, scattering it among at least two subsystems. A better option would be to adopt the view that signals intervene only as means to express the inner states of objects and, as such, they should be modeled close to these objects.

Unfortunately, this purist idea goes against the usual way of perceiving things and organizing people; it hits an immutable factor that habits have engendered. It is not to say that the culture of the company is impervious to change, but we need time for this kind of cultural shift to happen. Modeling attempts always unravel such knots. If not, this is a sign that modeling has not dug deeply enough. To conclude with this example, identifying systems and establishing their boundaries are acts subjected to uncertainty and a topic for debate. There may be a best solution but it will not necessarily be accepted or, to put it in other terms, one may have the feeling that there still remains a sort of arbitrary approach when delineating the systems.

A second example will show that the affair can be even trickier. Let us examine the classical dichotomy Product *versus* Production (or system-to-be-designed *versus* the socio-technical system, i.e., the project that produces the product). Where should maintenance stand? It would make sense to put it on either side:

- Maintenance contributes to transforming the product or to creating a new version of the product, or a new product. As such, we can justifiably decide that maintenance pertains to production. By the way, the total cost of ownership includes the maintenance costs, and the technical debt should be estimated once the project starts.
- However, there is another way to see things; this will appear if we put ourselves in the customer's shoes. From the client's perspective, the product system encompasses every element, every service and interaction that makes the product work. And the product works provided that the fees are paid, that the call-center answers the request, that the after-sales service repairs the failed parts, and so on and so forth. Following this analysis, maintenance is part of the product.

How can we decide between these two options? They are equally logical and, even though we fully develop both architecture scenarios, it is not sure that we will be able to differentiate between them with quantitative metrics. Nevertheless, their implications differ dramatically. I bet that the second option would drive a significant improvement in terms of quality of service and corporate image. But we face what we could coin arbitrariness or, at least, liberty: it's up to us, as architects and designers, to choose. When confronted with several equivalent options, the decision falls to us. With liberty comes duty.

Both examples (the signaling system and the maintenance case) share the sense of arbitrariness when drawing the boundaries of the systems. The second example is more about the organization and the discussion will be positioned in the pragmatic aspect, in Praxeme terms. On the contrary, the first example – taken from our case study – epitomizes the decisions architects have to make through the logical architecture. At this level, it is worth noticing an amazing phenomenon that tempers the debate: the subsystem boundaries are not so important! At least, they are less important than the interfaces that we bring out. There may be several ways to break down the system into pieces while supporting the same set of interfaces. At the end of the day, what really matters are not so much the boundaries of the subsystems, than the set of interfaces that compose the system and that ensure the interactions and operations. This is the outcome we expect from the logical design.

We can vary the way we decompose the system, but we will always need the same set of interfaces. We can dispatch the interfaces through several scenarios of architecture. A given interface may simply appear at various levels depending on the architecture scenario. This phenomenon holds true within certain limits, of course. It may be more easily understood when we examine the impact of the logical model. First and foremost, the logical model can be seen as the catalog of contracts, expressed through the interfaces (in the sense given by UML). Specific solutions, as they are delivered by the industry, refer to these contracts. In UML terms, we would say that the solution components implement the interfaces. This task occurs in the hardware aspect of the Enterprise System (tantamount to logistics). At this level, the material structure is not necessarily isomorphic to the logical structure. What is mandatory is that the material or physical solution concretely translates every required interface.

4 How Can Models Anticipate Issues in the Real World?

The case study was proposed by the steering committee. In the past, the signaling system was based on incandescent bulbs. As this technology was known to be subject to failure, the company developed a control device that informed the traffic regulator should an incident arise. As a result, the regulator in the central command post always had the same perception as the driver on the ground. Then came LED technology. In a general excess of optimism, the industry suppliers and the company decided that, due to the resilience of this technology, the control device was not necessary anymore. The migration started. Unfortunately the damaged filament was not the only cause of failure. For a signal – even an LED display – to work, it needs electricity! And sometimes there are problems with the electricity network. Because the new configuration gets rid of the feedback loop, it results in a situation where the traffic regulator has a different view of the reality perceived by the drivers. When drivers find themselves in front of a switched-off signal, they have to stop, even though they may know this is only a minor issue and the track is free. The regulation imposes such mandatory behavior. At the same time, the traffic regulator still thinks that the situation is fine and that the train is able to move on. This results in a delay before the situation is handled correctly. The full picture is a little more complicated, since it also contains a device that automatically stops the train in case the driver does not pay attention to the signal (see figure 4).

The question posed was: would a proper modeling approach have emphasized this issue?

4.1 The Problem Solving Approach and the Techniques Used

The first task is to represent the components involved in the case. To this end, we used the UML component symbol. The semantics of component in UML are restricted to software components, but they have been extended to cover all kinds of things (in SysML for example).

Figure 4 fits the elements of the problem together. It also displays the interfaces between these elements. The picture is certainly a simple one, but it took a while to stabilize it and it required us to uncover all implicit knowledge. For instance, we discovered that the lights were plugged into the signaling post and not directly into the electricity network. We had to make the interactions clear too.

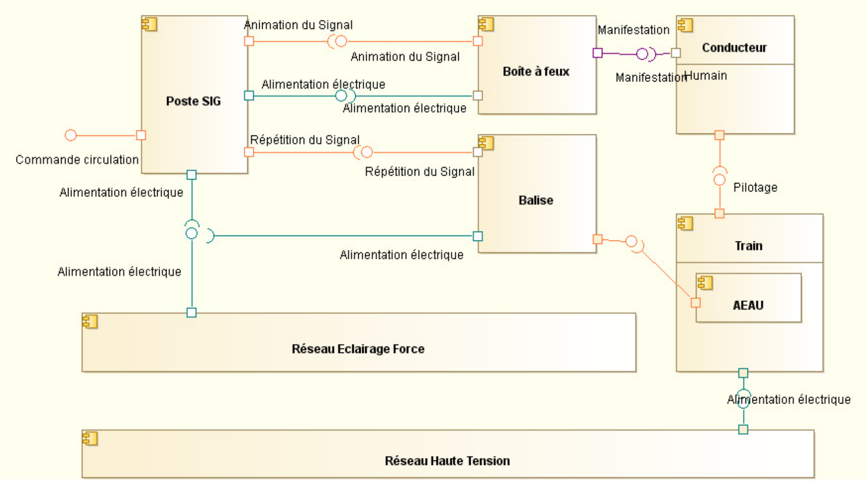


Fig. 4 Representation of the components that come into play in the case study (UML)

The interfaces are typified, and the customized notation adopts the color code of the Enterprise System Topology: every aspect is given a special color. Many interfaces come from the logical aspect and, thus, express command-control interactions. Their symbols are colored in orange, which is the color of the logical aspect. The physical interfaces – here with the electricity network – are blue. Last of all, we have to take into account interactions between the system and the human actors. To understand the overall behavior of the transport system, we have to introduce the behavior of the driver in front of the signal. Human perception and the attached states need to be included in the model. The human-machine interface is represented with the color of the pragmatic aspect, the aspect of the Enterprise System dedicated to human beings and to organization.

That way, the model accounts for all the dimensions of the problem. These dimensions are weaved together at the core of the model, still being disentangled according to the “separation of concerns” principle.

Knowing the elements, we have to examine their behavior. The state diagram below (figure 5) illustrates this modeling technique, which equips us with a powerful tool to analyze complex systems and their dynamics. This technique reveals itself to be not too difficult in practice because it applies to objects, at the scale of a single object. It consists in capturing the adjectives and phrases used in connection to the object name, then in linking them in a logical manner to reflect the behavior of the object.

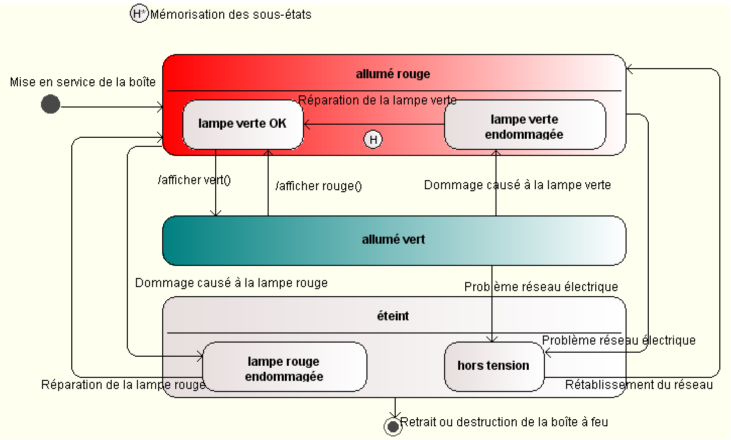

H* Mémorisation des sous-états
Mise en service de la boîte
allumé rouge
Réparation de la lampe verte
lampe verte OK
lampe verte endommagée
H
/afficher vert()
/afficher rouge()
Dommage causé à la lampe verte
allumé vert
Dommage causé à la lampe rouge
Problème réseau électrique
éteint
Problème réseau électrique
lampe rouge endommagée
hors tension
Réparation de la lampe rouge
Rétablissement du réseau
Retrait ou destruction de la boîte à feu

Fig. 5 The state diagram of a signal box

Now we are ready for the final act: with sequence diagrams, we document several scenarios that involve the component. The example below details the case of the LED device without feedback, when the regulator sends the authorization to move ahead but the signal box is out of order. In this configuration, the failure provokes two discrepancies, leading to a traffic incident.

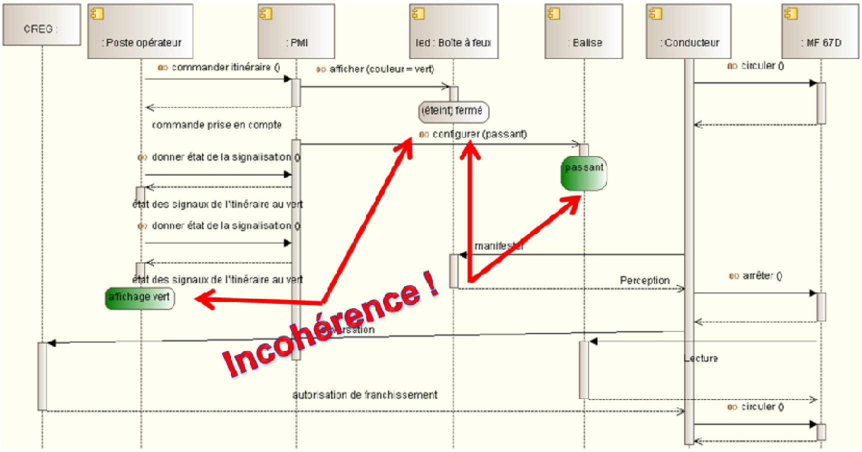


Fig. 6 A sequence diagram that helps to identify the issues in a special script

4.2 The Outcome

We used several sequence diagrams to analyze execution scenarios in various configurations: the previous solution with incandescent bulbs and feedback device; the new solution with LED; the LED solution with a new feedback device to be deployed… Not only did this proof of concept confirm the value of modeling in these matters, but it also revealed that the current solution being deployed does not work perfectly. Reading the corresponding sequence diagram, it became obvious that this was a partial solution – despite its cost – and that it will not prevent some incidents from happening.

This experience shed light on the current status of the documentation and on the potential of a more rigorous approach based on modeling techniques. The participants in the project were familiar neither with engineering nor with modeling, but were aware of the weaknesses and risks related to the status quo. At the end, they were convinced Praxeme offers an appropriate approach to physical systems, able to cast new light on design, to stimulate innovation and to tackle specific issues, filling the spectrum from overall architecture to detailed design.

From this specific example we drew a more general conclusion. The following principles were adopted that condense the essence of requirements, from the system owner's perspective:

- Whatever the system, subsystem or part at any level, the solution should guarantee constant feedback throughout the system. This high-level requirement is represented by a "synchronous" message in UML, as shown on figure 7 (with adapted semantics, not to be confused with its meaning in IT).
- Responsibilities are stated in terms of aspects and of systemic levels.

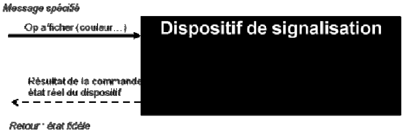


Fig. 7 A way to represent the generic feedback requirement and to assess compliance

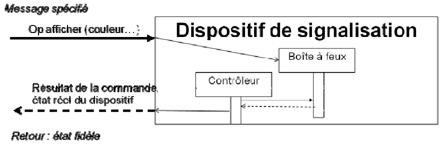


Fig. 8 White box design with arrangement of connection types to meet requirements (on a simplified sequence diagram)

4.3 *In Terms of Responsibilities*

A major concern in the company is with regard to the clarification of responsibilities between the system owners, the other departments and the component providers. With this aim in mind, the method frame – the Enterprise System Topology – can be used as a grid to revisit the process and to specify the deliverables with quality criteria and clear responsibilities.

Using the architecture framework (the Enterprise System Topology) helps to assign responsibilities in terms of models, rather than assigning them all along a process. The easy way consists in fixing responsibilities per aspect. Obviously, the models of intentional, semantic, pragmatic and geographic aspects are under the system owner's remit. As regards the logical aspect, things get more subtle. We could say that the system owner is the only person responsible for the logical model, while the industry providers take care of the material aspect. This could work in some contexts but not here. The logical model is far too detailed and the owner refuses to dive into excruciating details. As a result, we can specify responsibilities by fixing a level of detail for the modeling. The system owner's responsibilities cover the big picture, the overall design, and it goes down to the level where industrial partners are identified. At this level, components are black boxes in the eyes of the system owner. These boxes are perfectly specified to such a point that test cases are attached to them. Then the industrial provider has to satisfy the specification and to comply with the modeling approach. To the provider, the component is a white box, designed so that it fits the upper-level design, as shown in figure 8.

5 Conclusion

Praxeme can apply to physical systems[12], provided that some adjustments are made on its framework. We are carrying out an ongoing reflection and collective discussion inside the Praxeme Institute in order to review the Enterprise System Topology. Our goal is to generalize the architecture frame, so as to apply it to any kind of system without further adjustment. We also seek to build it on a solid theoretical basis, and we call for the support of the scientific community to this end.

Modeling is a powerful tool that helps us to anticipate and avoid issues in the real world. This is common wisdom, but only inside the community of modelers, architects and system practitioners. Nevertheless, it has to be constantly reasserted against the common mood that affects the management sphere and ruins any attempt to take the necessary time for thorough reflection.

This demonstration has been done in the context of the department in charge of the transport systems at RATP. The question now is whether this organization is willing to follow through with the demonstration and to cope with the growing complexity of its systems. Will it shy away in the face of technical difficulties, organizational deadlock and cultural resistance? Or will it be able to build on these conclusions and adopt a new approach, in spite of its fear that it lacks the competencies and resources to undergo this cultural shift? Ten million commuters are expecting them to!

References

Aiguier, M., Le Gall, P., Mabrouki, M.: A formal denotation of complex systems: how to use algebraic refinement to deal with complexity of systems. Technical Reports, http://www.epigenomique.genopole.fr

Aiguier, M., Le Gall, P., Mabrouki, M.: Complex software systems: Formalization and Applications. International Journal on Advances in Software 2(1), 47–62 (2009)

[12] In 2003, the first application of Praxeme was to unmanned aircraft vehicle systems, and SAGEM was the first of the contributors to join the initiative for an open method. There have been experiences in the field of weapon systems too (Thales, DGA, Dassault).

Léo, A.: Syntaxe, Sémantique et Pragmatique. In: Logique et Connaissance Scientifique dir. Jean Piaget, coll. La Pléiade, Gallimard (1967)

Bonnet, P., Detavernier, J.-M., Vauquier, D.: Sustainable IT Architecture. Wiley (2009)

Caseau, Y., Krob, D., Peyronnet, S.: Complexité des systèmes d'information: une famille de mesures de la complexité scalaire d'un schéma d'architecture. Génie Logiciel (2007)

Chingareva-Slavine, E.: Sémiotique, linguistique et modélisation, Lavoisier (2003)

Depecker, L.: Entre signe et concept, Eléments de terminologie générale. Presses Sorbonne Nouvelle (2003)

Enterprise Transformation Manifesto (In the face of complexity, this manifesto articulates core principles and offers an escape from confusion, gloom and doom. It aims to reinforce our ability to act.), http://www.enterprisetransformationmanifesto.org

Garajedaghi, J.: Systems Thinking. Butterworth-Heinemann (2006)

IEEE Recommended Practice for Architectural Description of Software-Intensive Systems, IEEE Std 1471-2000

ISO: Norme internationale ISO 704, Travail terminologique-Principes et méthodes (français et anglais), Afnor, 3e édition, 65 p. (1ère édition, 2e édition 2000) (décembre 2009)

Le Moigne, J.L.: La Théorie du système general. PUF (1984)

Meinadier, J.P.: Ingénierie et intégration des systèmes. Hermès (1998)

Object Management Group, OMG Systems Modeling Language (SysML) OMG (2010)

Object Management Group, OMG Unified Modeling Language (UML), OMG (2010)

Poincaré, H.: Science et method. Flammarion (éd.) (1908)

Vauquier, D.: Praxeme methodological guides, http://www.praxeme.org

Vauquier, D.: Développement Orienté Objet. Eyrolles (1993)

Walliser, B.: Systèmes et modèles. Editions du Seuil (1977)

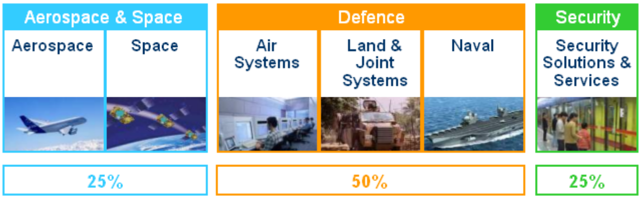
Chapter 6
PBS: A Major Enabler for Systems Engineering
A Thematic Thinking by Thales Systèmes Aéroportés

Edmond Tonnellier and Olivier Terrien

Abstract. In 2012, Thales Systèmes Aéroportés initiated a study on the PBS (Product Breakdown Structure) as the central enabler for Systems Engineering throughout the lifecycle of a Solution. The obvious similarities between tree representations suggest in a simplistic and reductive way that breakdown structures are only an unnecessary redundancy due to complementary businesses designating the same reality in different ways. However, a more careful examination of international standards and stepped uses of these tree frames shows how different their purposes are and how they support different features of the system being designed. Nevertheless a confusion exists between these tools. Our paper asks a question of principle: "does this confusion come from imprecise and unclear standards on this subject, from incorrect interpretations of these by system architects or from vague approximations in the use of technical tools that structure and size R&D projects?".

1 Context

Thales Systèmes Aéroportés is a company within the Thales group.



Edmond Tonnellier · Olivier Terrien
Thales Systèmes Aéroportés
2 Av Gay Lussac
Elancourt, 78851
France
e-mail: {olivier.terrien,edmond.tonnellier}@fr.thalesgroup.com

Thales Systèmes Aéroportés is currently the European leader and the third player worldwide on the market of airborne and naval defense equipment and systems. The company designs, develops and produces solutions at the cutting edge of technology for platforms as varied as fighter aircrafts, drones, surveillance aircrafts as well as ships and submarines.

About 20 % of the turnover is dedicated to R&D activities with a large proportion for systems engineering. With facilities in many European countries, the company employs numerous highly-qualified experts to design solutions matching increasingly complex customer requirements as closely as possible. Benefiting from traditional positions as leader in the electronic intelligence and surveillance systems, recognized know-how in radar for fighter aircraft and electronic warfare, Thales Systèmes Aéroportés involves its customers from the solution definition phase up to the operational validation of the products. This requires high reactivity to changes in specifications and an ongoing effort to reduce development cycles. In addition, international competition and the effect of the euro/dollar conversion results in major budget constraints (in recurring and non-recurring costs).

The complexity of the systems developed within Thales is due to the number of components involved, the numerous technologies used and the volume of the embedded software. Due to the systematic combination of technical and non-technical constraints in new contracts, the technical teams must synchronize more and more precisely, skills from complementary disciplines whose contributors are often based in several countries. Lastly, the accelerated pace of developments requires detailed definitions as well as optimum reactivity to the events or defects encountered, inevitable consequences of faster changes in requirements and markets.

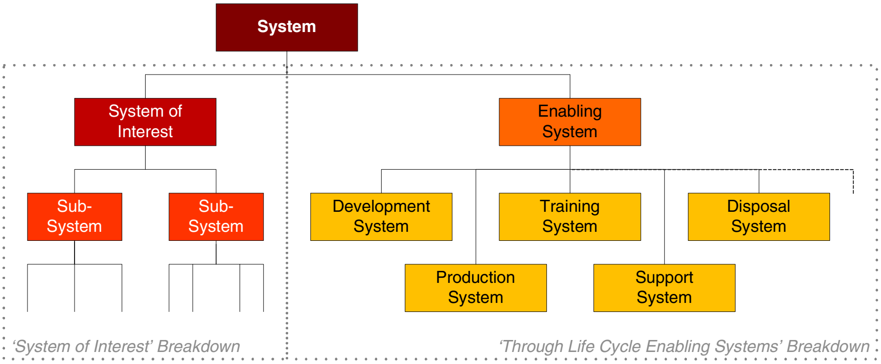
In 2012 Thales Systèmes Aéroportés initiated a study on the PBS (Product Breakdown Structure) as the central enabler for Systems Engineering throughout the lifecycle of a Solution.

2 Trouble with Breakdown Structures

Fundamental for Configuration Management, a technical tree is a structured representation of the various components of a system ('a tree frame'). Actually, there are a multitude of breakdown structures mentioned in technical processes required to design and develop a new solution. The obvious similarities between these various representations suggest in a simplistic and reductive way that these are only an unnecessary redundancy due to complementary businesses designating the same reality in different ways. However, a more careful examination of international standards and stepped uses of these breakdown structures shows how different their purposes are and how they support different features of the system to be designed.

2.1 Question of Principle

At the end of a workshop to adjust development processes of a new solution, a question of principle raised by a participant triggered this more in-depth thinking



Typical 'technical tree' breakdown of a system / solution

on our understanding and control of the PBS (Product Breakdown Structure) during R&D projects. This question dealt with the possible confusion between technical trees or breakdown structures.

"Does this confusion come from imprecise and unclear standards on this subject, incorrect interpretations of these breakdowns by system architects or by vague approximations in the use of tools which structure and size R&D projects?"

Our thinking is organized around these three themes:

- The incomplete and/or inaccurate definition of technical breakdown structures within design and development processes;
- The appropriation and real use of them in each phase of a product/solution development cycle;
- The behavior of designers located between "too much" and "too little" when developing complex systems and the associated level of visibility.

2.2 Surveys and Benchmarks

To go beyond theories found in texts and to monitor how breakdown structures are applied in the field, we conducted a number of interviews with various contributors to product/solution development. Several benchmarks from companies that implement long and risky design processes confirmed the potential risk of incomplete, incorrect or inaccurate use of technical trees spontaneously mentioned during our interviews.

The immediate conclusion from these feedback comes from the inevitable uncertainty about the future that causes two diametrically opposed behaviors: the desire to anticipate every possible situation by describing in detail all the requirements of a solution, and the choice of dealing only with problems when they occur without making a lot of extra effort to eliminate blocking situations that are not necessarily going to appear. In the first case, describing in detail may lead to a humanly unmanageable, and therefore unacceptable, degree of complexity. In the second case, reacting only to dangerous situations when the defect appears cause problems to pile up that are impossible to overcome and that may stop the project.

This first observation shows how the choice made by Systems Engineering is at the heart of the issue of breakdown structures: *“what level of visibility and what depth of analysis are required to control the complexity of a new development?”*

2.3 Diagnosis

Rather than thinking in absolute terms and about intricacies of technical tools, let us try to take a somewhat stereotyped look at the different representations.

‘What’, ‘Who’ and ‘How’

Imagine a technical team not supplying physical equipment but mainly focusing on its workload. Why use the PBS (Product Breakdown Structure), when the WBS (Work Breakdown Structure) directly represents the activities to be carried out? At this point, a confusion exists between the 'what to do' which is usually closed to similar products and the 'how to do it' which is based on business standards. Building a WBS results from iterative choices and interactions with the PBS. This confusion starts when people do not know which one starts the other. Especially when an OBS (Organizational Breakdown Structure) is superimposed on the activities to be assigned.

‘The Development of a Solution’ and ‘The Solution Developed’

Take a team made up only of designers and architects preparing a new development. Why use the PBS (Product Breakdown Structure), when the development breakdown directly structures the functional characteristics of the product to be designed? A confusion arises from the goal to be reached and the path taken to reach it. The final product may require preliminary models or specific means of validation (simulators, test benches, etc.) but also all enabling systems such as operations, training, maintenance, production, logistics, etc. Building a development breakdown supports all the specifications of the new solution. This confusion starts when people do not discriminate the product which has to be accepted once by the customer and the copies which has to be accepted at each delivery.

NRCs & RCs

Consider a team designing a new system that will be replicated to the extent of several dozen copies. Why use the PBS (Product Breakdown Structure), when the design breakdown identifies all physical components to be made, acquired and assembled? A confusion arises from this difference between costs not yet spent (like the recurring costs of productions, RC) but already committed by the development decisions (due to architecture, choice of components, etc through the non recurring costs of development, NRC). This first stage leads to the validation of a system but only represents a small proportion of the total expenditures, over the complete life cycle. Estimating a design breakdown only provides an order of magnitude for the recurring portion of costs. This confusion starts when people misunderstand the one that pays an activity and the one who decides of it: the functional acceptance of a system and the recurrent acceptance of all the constituents of a copy.

Re-use, Product Lines and Interchangeability

Imagine a sales team drawing up the Bid for a new solution. Why use the PBS (Product Breakdown Structure), when the concatenation of sub-sections of design breakdown trees from previous projects responds roughly to the request? A confusion arises from the conditions of use, the (un)known limits of use and the assumptions about reuse. The combination of single risks may generate an overall risk much greater than the simple summation of the single problems under consideration. Estimating a solution for a Bid is the sum of elementary costs models but also the combination of associated risks and possible technical impossibilities. This confusion starts when people only focused on visible elements of a proposal and not awared of the invisible and negative impacts of the choices made.

2.4 *Structure of This Document*

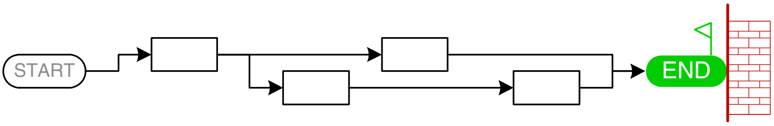
For long life cycle solutions with high levels of risk, each development phase can identify its own costs and risks. However, each phase cannot quickly estimate overcosts and troubles that it might generate for the rest of the life cycle of the solution being developed. Yet this is a key enabler for controlling the overall cost of a project. A local optimum may be dangerous for the whole project. While the customer of a new system quickly grasps the procurement and development costs, he must also consider the overall costs of his system including probable overcosts and possible risks for each of these aspects through a complete life cycle: operation, reparation, training... right up to decommission.

3 PBS and Bids

During this preliminary step in designing a new solution, the way this solution is structured is of paramount importance because the exploration of the initial concepts define the first levels of technical and functional trees, the activities and therefore their estimates, the partnership strategies, manufacturing and logistics constraints, etc. This is why the PBS is the technical tree at the heart of this first subsection. A reminder of the specific features of the Bid process will be followed by a definition and an illustration of a PBS.

3.1 *Bid Process*

In the Aeronautics & Defense fields, the Bid process has various characteristics of which the main one is to stand as an immutable T_{END} process. If the response is a day late, all the work done is worthless because the proposal submitted will not be analyzed by the potential customer. This constraint leads to two notions: restricted visibility on certain sub-sections of the solution and de-risking actions on the chosen solution.



In constrained time, it is impossible to obtain a comprehensive, consolidated and 100% sure assessment (apart from responding outside the time frame). The challenge for Systems Engineering is to limit the risks taken so that they do not become out of all proportion during the project, i.e. when the company is fully engaged in the contract. From this constrained time, two imperatives emerge:

1) formulate a solution that covers 100% of the customer's requests (one or more responses for each question identified) and that deals with 100 % of the life cycle of the solution (from development to decommissioning). Cover everything in constrained time but not every thing is covered in detail, hence the notion of restricted visibility.

2) minimize the risks taken by detailing / examining areas that are vague, ambiguous or obscure. Refine the most critical parts in order to limit the risks taken to an acceptable level and to make sure that the chosen solution is secure, hence the notion of "de-risking" the inevitable uncertainty reigning during design and development processes.

3.2 Definition of the PBS

The definition of a PBS by ANSI/EIA-632 Standards shows how relevant this technical breakdown structure is during this exploration of initial concepts in Bid phase: *"PBS is a hierarchical structure of the complete set of physical systems and subsystems including operational system, training system, development support, production support, etc which identifies the configuration items"*.

The purpose of a PBS is therefore to structure the solution, taking into account its entire lifecycle (design, development, operation, training, repair, etc.) but also determining the types of its components (e.g. prototypes, testing facilities, supplied constituents, etc.). From this tree representation, it is possible to estimate the overall cost of this solution with a satisfactory visibility (for example, depending on the criticality of particular constituents). The PBS also makes it possible to define strategies for the "system of interest" (SOI) but also for the enabling systems ("to make" and "to maintain").

The main constraint of the PBS lies in the time available to create it. The PBS, initialized in the Bid phase, usually a very tense one, must stop at an acceptable level of detail that secures the solution. The acceptable is then the criterion to be specified. The balance between "too much" and "too little" and the associated level of visibility becomes the real challenge for Systems Engineering during this preliminary phase of a project.

3.3 Levels of Visibility and Stopping Criteria

Lightening dark areas, illuminating shadowy areas and confirming bright areas are three activities to be conducted in the Bid process. However, within the limited time, they must only immobilize available resources to the absolute minimum necessary and then stop at the satisfactory level to reallocate resources onto higher risk areas. What criteria should be provided to Systems Engineering to stop an analysis in time and focus efforts on a more relevant subject?

The feedback obtained through our interviews shows that an acceptable level for a sub-section of the PBS is one that allows the Systems Engineering to make an estimate based on its usual cost models and to stabilize the associated assumptions. From the choices made concerning re-use, compliance with existing product lines, partnerships and subcontracting, cost models estimate the cost of a constituent, value the potential risks (technical, financial, scheduling, organizational, contractual, etc.) and compare the solution to previous similar designs (main problems experienced, observed instability of customer requests, etc.).

So the PBS structures the solution into large sub-sections but also into activities with an index expressing confidence in its being achieved.

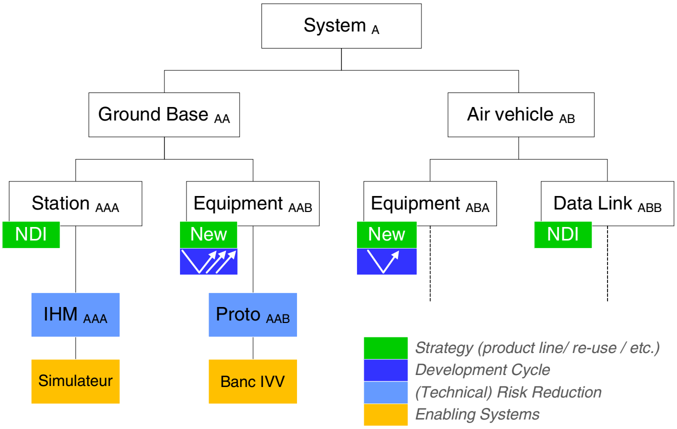
3.4 Illustration of a PBS in Bid Phase

The breakdown of a system is progressive and iterative until one reaches the stopping criteria mentioned above.

Break-Down and Types to Identify Constituents
A first pass is used to break the system down into subsets by identifying strategies of re-use/ acquisition/ development and Technical Readiness Level (TRL). This first representation quickly gives direction to the project strategy.

Compliance and Risk Estimation
A second pass verifies compliance with the needs expressed by the customer and provides an estimated risk taken for each component (including any de-risking action). This second representation is more complete and shows how the different elements of a system are structured and how the system can be recomposed (architecture & schedule).

Interdependence and Qualifications
A third pass characterizes development cycles of the sub-sections to be designed, the “Team / Make / Buy” strategies, the development and IVV strategies (Integration, Verification, Validation with the associated enabling systems) and all products in order to make, to produce and to support. This third representation identifies the interactions and dependencies between constituents of the complete system.



PBS of an Unmanned Aerial System during Bid phase

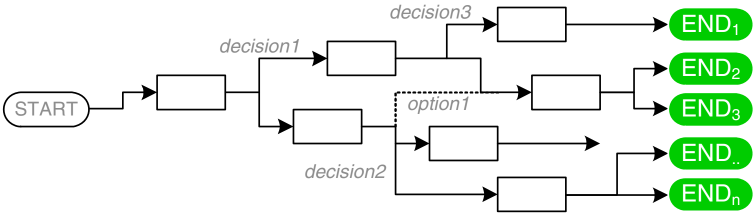
During Bid phases, the PBS appears as a major act in Systems Engineering. It is an important enabler for structuring the solution (architecture & development cycle) through the life cycle and sizing the project (activities & strategies).

4 PBS and Projects

The contract has been signed and the complete solution has yet to be designed. Initialized as of the Bid phase, the PBS directs development. The PBS includes Non Development Items (NDI, to be directed towards the purchase teams), hardware and software items (HWCI and SWCI to be designed, to be assigned to teams of Hardware/Software designers), together with information on manufacturing, logistics, etc.

4.1 Development Process

In the Aeronautics & Defense fields, development processes have certain characteristics of which the main one is to transform information by linking decision-making actions. They are therefore a divergent process because all decisions provide some added value for the solution but also new risks to be taken.



The sum of all decisions does not guarantee the success of a development as the transformations of information are most often dependent on each other. A well-known example shows that fixing twenty or so screws is not the sum of tightening twenty screws. Tightening one screw after the other often means that it will be impossible to fix the final one (thanks to Murphy's Law). Fastening all the screws is the result of twenty tightening operations but also the way in which tightening operations are ordered. Similarly, a development process is not the sum of all the activities to be carried out but an accumulation of work with a common vision and a common purpose. This vision is provided by the PBS. This technical tree makes it possible to stay focused on purpose when choices must be made, and to avoid differences that each individual designer may introduce into the complete system to achieve a local optimum. This notion is the major lever of the PBS which ensures convergence of forces towards acceptance of the complete system. All strategies being planned accordingly.

4.2 Development Breakdown

On the basis of the PBS, the development breakdown is a structured representation of the system divided into subsystems and configuration items (CI). It takes into account the functional characteristics of the system by organizing all the configuration items and linking between levels all the traceability links of the specified requirements (requirements of level N are directly derived from level N-1). The purpose of this tree is to support the "functional" and "'specified" statuses of the solution during design and development activities.

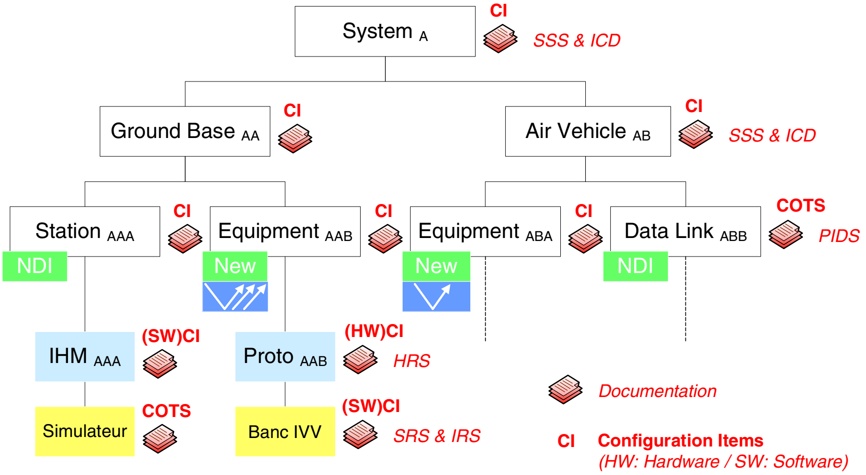
4.3 Configuration Item and Selection Criteria

By means of a generally top-down, progressive and iterative approach, the development breakdown divides the system into its main constituents. Once again, the question arises about the level of detail and the complexity of the Configuration Management. This is why this representation involves identifying the main components and using the functional architecture, the physical architecture of the system and also the interactions between all the components of the solution. Systems Engineering allocates characteristics (functions, requirements, constraints) to each of these elements. So a configuration item is a particular article since it has to be specified, identified, assembled, tested and maintained.

Increasing the number of configuration items (CI) provides a detailed overview of the system but requires complex and rigorous management. Multiplying CIs leads to a proliferation of documents, reviews, etc. On the other hand, decreasing the number of CIs makes Configuration Management easy but complicates Change Management. Reducing CIs extends their size to an extent which encourages the propagation of changes between items.

4.4 Illustration of a Development Breakdown

The construction of a development breakdown starts out from the PBS and applies criteria for selecting configuration items such as Technical Readiness Levels (TRL), common components between several products, co- and sub-contracting, planned releases, etc.



Adjustment of a PBS to prepare a development breakdown

4.5 IVVQ Strategies

Often called the “Vee up” of the V cycle, the ‘Integration, Verification, Validation, Qualification’ sequence is part of the development process. These activities prove whether the system functions as designed: demonstrating that the product is right and that it is the right product. To do this requires passing milestones, and meeting the criteria for completion of the development. The specific nature of this sub-process lies in its predefined criteria as of the start of development. The IVVQ strategy therefore depends on the PBS which structures the solution and the development tree that specifies the break-down of the system to be approved. Proper control of the development is judged by the end of the project. The experience of previous V cycles is therefore essential to build these technical trees. As the IVVQ scenario involves a number of items and a number of increments, the PBS structures project planning by introducing incremental life cycles and/or changing components (the V cycle is only theoretical and cannot withstand the reality of the versions of a system).

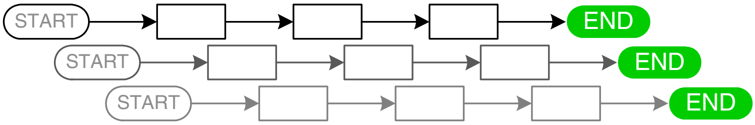
At project start-up, the PBS is seen as a crucial adjustment enabler for Systems Engineering. It is the common vision to unite efforts of all involved disciplines and to guide all decisions towards a common goal: a successful developed solution.

5 PBS and Productions

Multiple stakeholders may be involved in the building of the PBS. One of the main users of development processes may introduce an additional tree.

5.1 Production Processes

The production process has the special feature of reproducing an approved configuration and assembling the components of a system because their interactions have been validated previously. So this is a recurring process that leads to repetition.



An initial state of the system, meaning that it was originally approved, may change by modifications: a defined state is the initial approved state supplemented by all approved changes. This definition and this approbation become two major issues of this production process. The notion of interchangeability is therefore essential beyond the mere correct operation of each component. The PBS can stand as a common vision to insure coherence through the complete life cycle of a solution: interchangeability within the system of interest but also in enabling systems.

5.2 Design Breakdown

Deducted from the development breakdown without calling it into question, the design breakdown is the structured representation of all items of a system. It reflects the physical characteristics of the system by structuring the items but also the definition documents that can reproduce multiple identical copies. The links between the levels of this representation reflect the assembly links (the level N article enters into the breakdown of level N-1).

These two technical trees - development and design breakdowns - are two representations of the same system. They do not have the same purpose but are not independent from each other because one is derived from the other. Activities and decisions during the building of the first one have a strong impact on the other.

5.3 Logistic Breakdown / Exchange Criteria

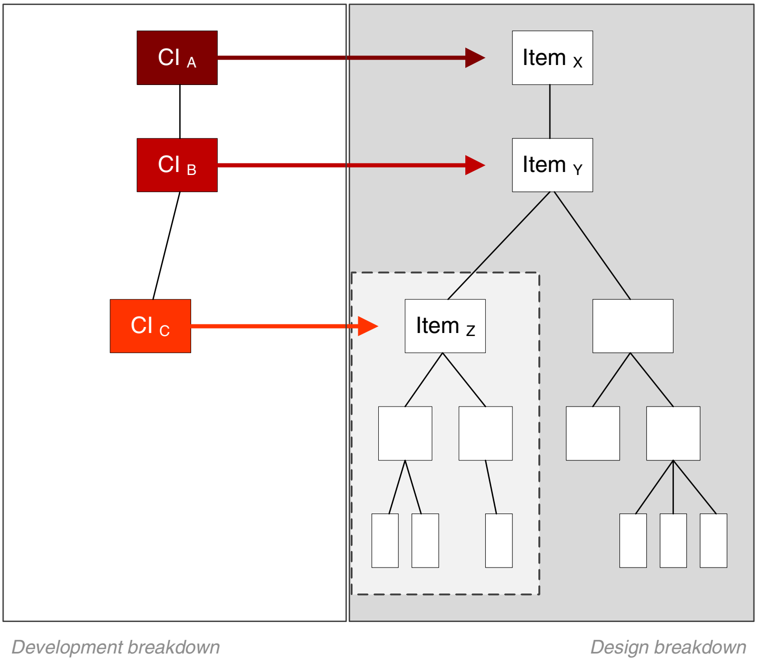
In the design breakdown, the question is raised of exchanging a component and the impact of a change on the rest of the entire system. These items are both exchangeable and serviceable by the customer by swapping their constituents, and are distinguished by the level at which this is done:

- Shop Replaceable Unit (SRU),
- Line Replaceable Unit (LRU).

These logistic compounds will not necessarily be CIs, but they may appear as items tracked in the various configuration states. The logistic constituents likely to be configuration items must have a certain level of complexity.

5.4 Illustration of a Design Breakdown

Building a design breakdown starts out from the development breakdown and applies criteria for selecting configuration items (CI).



Visibility between development and design breakdowns

During production phase, the PBS keeps on leading decisions to make and to support the complete system. Its role has to be reinforced in order to keep coherence between all the phases of a project (from Bid to Disposal).

6 Conclusion

As a conclusion,

The PBS is a major enabler for Systems Engineering. During Bid, Project, Production phases, the PBS represents a possible lighthouse that pulls every orientation, that lightens dark situations and that federates efforts of all the various disciplines involved in Complex Systems Design & Management.

Chapter 7
Complex Systems Architecture Framework: Extension to Multi-objective Optimization

Abdelkrim Doufene, Hugo G. Chalé-Góngora, and Daniel Krob

Abstract. This paper shows the utility to follow an architecture framework in order to design complex systems with a holistic approach. Multi-objective Optimization techniques extend and complete the architecture framework to support trade-off analysis and decision making in the Systems Engineering design process. The merging and combination of these two approaches, decision making and systems engineering, contribute to the efficient design of systems by helping to meet needs and constraints stemming mainly from the system analysis.

To support this assertion, we present a case study for an Electric Vehicle Powertrain. The decision problem is modeled as a Pareto model, in order to find a solution for the Electric Vehicle Powertrain that maximizes its autonomy and minimizes its total cost of ownership.

Keywords: complex systems, architecture framework, trade-off analysis, multi-objective optimization, electric vehicle.

1 Introduction

Trade-off analyses during complex systems design are inevitable. They are useful and necessary to make almost all the decisions during system design. Different kinds of decisions are involved: the choice of a safe and not too expensive architecture, the balance between the most reliable but still available architecture

Abdelkrim Doufene
RENAULT & École polytechnique Laboratoire d'informatique (LIX)
e-mail: Abdelkrim.doufene@renault.com,
doufene@lix.polytechnique.fr

Hugo G. Chalé-Góngora
RENAULT, 1 avenue du Golf, 78288 Guyancourt, France
e-mail: hugo.chale-gongora@renault.com

Daniel Krob
École polytechnique Laboratoire d'informatique (LIX) 91128 Palaiseau Cedex, France
e-mail: dk@lix.polytechnique.fr

and so on. We can imagine a lot of possible tradeoffs taking into consideration the stakeholders' needs, the total cost of ownership of the system we want to design, its lifecycle properties such as quality, reliability, safety, flexibility, robustness, durability, scalability, sustainability… In addition, economical, technological, societal and regulatory feasibility studies are indispensable in order to satisfy all the stakeholders' needs around the system of interest.

Indeed, in industrial practice, in order to design a complex system, there are several multidisciplinary objectives and constraints. Making analyses, defining the right criteria and evaluating the possible alternatives are hard tasks. This difficulty is due in particular to the fact that the separation between the problem definition and the solution design is often blurry.

Using an architecture framework could considerably contribute to bridge this gap, and to clarify the link between design constraints and design variables, often mixed in practice. An architecture framework provides guidance and rules for structuring and organizing system architectures. The several viewpoints that allow covering all the scope of the system architecture and the different abstraction levels add clarity to the problem definition and solution design processes.

To explain this assertion, we present in this paper the architecture framework explained in (Krob, 2009 and 2010), and we show the contribution of multi-objective optimization models in the decision-making process. We illustrate these contributions on a simplified case study. Our system of interest is the Electric Vehicle Powertrain (EVP) whose mission is to propel an Electric Vehicle (EV). The three global measures of effectiveness related to EVs are the protection of the environment by the reduction of carbon dioxide emissions[1], a TCO (total cost of ownership) that must at least be equivalent to internal combustion engine vehicles and a driving autonomy.

This paper is organized as follows. We begin with an overview of the background of our study, the architecture framework we use and the multi-objective optimization problem formulation. We then explain the context of the optimization problem studied in this paper. We present the requirements analysis and the objectives to reach, we define the parameters and variables to be used, we represent the optimization problem as a multi-objective optimization mathematical formulation. Finally, before concluding, we discuss some tests and results.

Note that this work is a part of current initiatives at Renault aiming at improving the efficiency of product development and mastering the complexity of automotive systems as reported in (Chalé Góngora et al. 2012).

2 Background

The decision problem model presented in this article is based on a systems architecture model. In this section, we briefly present the systems architecture framework we use, explained in (Krob 2009 and 2010). Then we present some

[1] The same way for other pollutants as: Nitrogen oxide (NOx), Hydrocarbon (HC) and Carbon monoxide (CO). Source: Presentation of Rémi Bastien, Director of the Department of Research, Advanced Engineering, and Materials at Renault: The Electric Vehicle Program of the Renault-Nissan Alliance, in Automotive Electronics and Systems Congress CESA, Electric and Hybrid Vehicles, Paris, November 2010.

related works on multi-objective optimization in the context of complex systems engineering.

2.1 Systems Architecture Framework

In order to design our System of Interest (SOI), we follow a framework that provides guidance and rules for structuring and organizing system architectures. The framework provides several viewpoints that allow covering the whole scope of the system architecture. It is inspired from the SAGACE method originally proposed in (Penalva 1997), which revolves around three main principles: a modeling approach, a representation of views (a matrix of nine points of view) and a graphical modeling language. In this method, the SOI is defined in an iterative manner. First, elements like issues and system environment, project purpose and missions, stakeholders are identified. Then, we can design the system and describe it with a matrix of nine points of view: Operational, Functional and Structural views, refined by three temporal perspectives (Benkhannouchel 1993; Chatel 2004; Meinadier 1998 and 2002).

In our study, we also use these three main views but we refine them by behavioral - and not temporal, as in SAGACE - perspectives in order to be able to use the SysML modeling language, as explained in (Krob 2009 and 2010). This Framework was adapted for the automotive context as explained in (Chalé Góngora et al. 2012). Figure 1 gives an overview of the analysis and modeling steps used in our study.

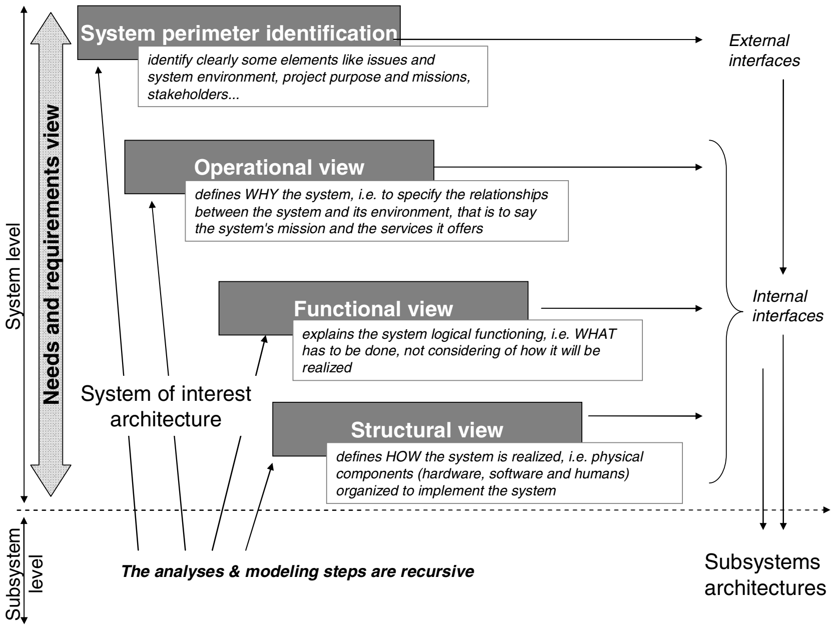

System perimeter identification
identify clearly some elements like issues and system environment, project purpose and missions, stakeholders...
External interfaces
Operational view
defines WHY the system, i.e. to specify the relationships between the system and its environment, that is to say the system's mission and the services it offers
Functional view
explains the system logical functioning, i.e. WHAT has to be done, not considering of how it will be realized
Internal interfaces
Structural view
defines HOW the system is realized, i.e. physical components (hardware, software and humans) organized to implement the system
System level
Needs and requirements view
System of interest architecture
Subsystem level
The analyses & modeling steps are recursive
Subsystems architectures

Fig. 1 Analysis and modeling process (adapted from (Krob 2009))

The system perimeter identification and the operational view, followed by a needs analysis, define clearly the problem, and the SOI missions and issues. It is during these steps that the Measures Of Effectiveness (MOEs) of the SOI are identified. *The MOEs are the "operational" measures of success that are closely related to the achievement of the mission or operational objective being evaluated, in the intended operational environment under a specified set of conditions (i.e., how well the solution achieves the intended purpose)* (INCOSE 2010).

Additionally, we highlight some *Measures Of Performances (MOPs) that define the key performance characteristics the system should have when fielded and operated in its intended operating environment. MOPs are used to assess whether the system meets design or performance requirements that are necessary to satisfy the MOEs. MOPs should be derived from or provide insight for MOEs or other user needs.* (INCOSE 2010). Performances may be economic, technical, user-perceived and so on. The differentiation between these two key concepts (MOEs and MOPs) is important in order to resolve multi-objective optimization problems.

2.2 Multi-objective Optimization

Multi-objective optimization (MOO) in the context of the engineering and design of complex systems, is very useful in trade-off analysis and decision-making. It allows taking into consideration multidisciplinary objectives such as performance, cost, schedule and risk (cf. Maier and Rechtin, 2000 in (de Weck 2006)). A brief review of the history of MOO and the most popular methods are presented in (de Weck 2006). We reproduce the same formulation of a MOO problem in the following equation (1).

$$
\begin{aligned}
& \min J(x,p) && \text{Where} \\
& \text{s.t. } g(x,p) \leq 0 && J = \left[J_1(x) \ldots J_z(x)\right]^T \\
& \quad h(x,p) = 0 && x = \left[x_1 \ldots . x_i \ldots x_n\right]^T \\
& x_{i,LB} \leq x_i \leq x_{i,UB} \quad \boldsymbol{(i=1,\ldots,n)} && g = \left[g_1(x) \ldots g_{m_1}(x)\right]^T \\
& x \in S && h = \left[h_1(x) \ldots h_{m_2}(x)\right]^T
\end{aligned}
\tag{1}
$$

Here, $\boldsymbol{J}$ is a column vector of $\boldsymbol{z}$ objectives, whereby $\boldsymbol{J_i} \in \boldsymbol{R}$. The individual objectives are dependent on a vector $\boldsymbol{x}$ of n design variables as well a vector of fixed parameters, $\boldsymbol{p}$. The individual design variables are assumed continuous and can be changed independently by a designer within upper and lower bounds, $\boldsymbol{x_{UB}}$ and $\boldsymbol{x_{LB}}$, respectively. In order for a particular design $\boldsymbol{x}$ to be in the feasible domain $\boldsymbol{S}$, both a vector $\boldsymbol{g}$ of $\boldsymbol{m_1}$ inequality constraints, and a vector $\boldsymbol{h}$ of $\boldsymbol{m_2}$ equality constraints have to be satisfied. The problem is to minimize – simultaneously – all the elements of the objective vector (de Weck 2006).

The survey of MOO concepts and methods presented in (Marler and Arora 2004), classifies methods into four major categories: methods with a priori articulation of preferences, methods with a posteriori articulation of preferences, methods with no articulation of preferences and genetic algorithms. The reader can refer to the surveys on MOO methods in (Andersson 2000) and (Coello 2010), for further references.

In a MOO problem, there is no single global solution, and it is often necessary to determine a set of points that all fit a predetermined definition for an optimum. The predominant concept in defining an optimal point is that of Pareto optimality (Marler and Arora 2004). Works have been done using Pareto modeling in the context of systems architecture. We can cite, for example, (Kim and de Weck 2005a, 2005b), (Smaling and de Weck 2004) and (Smaling 2005).

The difficulty in industrial practice is the identification of the adequate objectives, design constraints, design variables and mathematical relations among them. Which are the necessary and useful criteria in order to evaluate the design alternatives? Are there priorities? Questions we often ask.

This case study shows an example how to deal with these questions. Our SOI is the Electric Vehicle Powertrain (EVP). The main challenges that the EVP has to face are autonomy (the driving capacity, given the embedded electric energy in the vehicle, measured by distance or time) and Total Cost of Ownership (TCO). However, the EVP has to meet a set of other needs and constraints stemming mainly from its environment and from the different technologies that are involved. We model a decision problem as a Pareto model with no articulation of preferences, in order to find adequate and satisfactory values of dimensional parameters of a solution that maximizes the EVP autonomy and minimizes its TCO, and that best meets all the needs and constraints.

3 Context of Our Optimization Problem

As explained previously, our study is based on the utilization of an architecture framework in order to design the SOI. In one word, all the information useful for the optimization problem stems from this architecture framework. The first steps highlight the SOI missions and its environment.

Our SOI is the EVP whose mission is to propel an EV. Given that we use an electric powertrain, the EV MOE related to the CO2 reduction is achieved. Therefore, we consider that the EVP MOEs are the driving autonomy (or range, that we define as the driving capacity, given the embedded electric energy in the vehicle, measured by distance or time) and total cost of ownership.

The EVP must be well integrated in its environment in order to achieve its mission. The environment modeling identifies the external systems that interact with the EVP. In this study, we consider the following significant external systems: the users, standards and regulations, roads, environmental conditions (weather), charging stations, and other subsystems of the vehicle (e.g. the charge management system).

The operational analysis of the EVP, following the architecture framework, highlights its lifecycle phases (Design, Production, Vehicle integration, Distribution/ Commercialization, Use/ Exploitation, Maintenance / Reparation and

Recycling). In the present study, we will focus on the Use / Exploitation phase, for which Use Cases and Scenarios analyses help to discover the SOI functions.

Still following the architecture framework, we show in figure 2 a first stage of the functional and structural architectures design. The circles represent the macro-functions organized in a Functional Breakdown Structure (FBS) and the rectangles represent the components organized in a Product Breakdown Structure (PBS). The arrows <Allocated to> serve to show the Functions to Components allocation process.

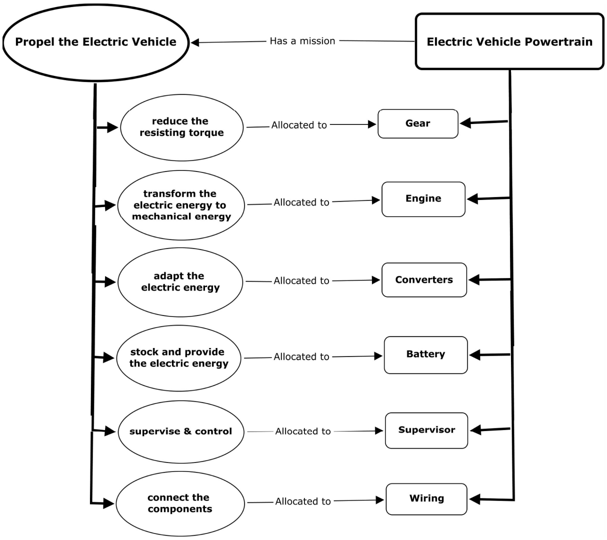

Propel the Electric Vehicle
Has a mission
Electric Vehicle Powertrain
reduce the resisting torque
Allocated to
Gear
transform the electric energy to mechanical energy
Allocated to
Engine
adapt the electric energy
Allocated to
Converters
stock and provide the electric energy
Allocated to
Battery
supervise & control
Allocated to
Supervisor
connect the components
Allocated to
Wiring

Fig. 2 Macro FBS and PBS of the EVP

Globally, in order to provide the sufficient torque, the EVP transforms an electric energy into mechanical energy. A battery serves to stock and provide the electric energy. An engine transforms the electric energy to a mechanical energy. Converters are used in order to adapt the electric energy from the battery to the engine. A gear serves to reduce the resisting torque. A supervisor (actually a set of electronic control units) supervises and manages the EVP. Electric wiring connects all these components.

We summarize in figure 3 the EVP components and its environment. It is clear that the external systems are more numerous and diverse depending on the life cycle phases of the EVP. For the sake of clarity, we focus on those that are significant in our context. We aim at pre-sizing the EVP that best meets all the needs and constraints, in order to optimize its autonomy and TCO. In the next sections, we present the inputs of the problem, the analysis of these inputs and the optimization objectives we have to reach.

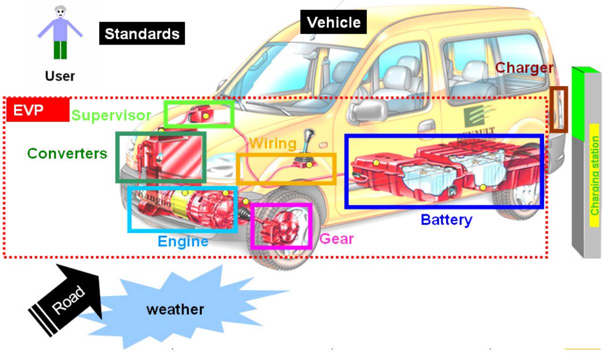


Fig. 3 Overview of the environment and the components of an EVP

4 Requirement Analysis

4.1 Inputs of the Problem

The main inputs of the problem are a set of needs and constraints as stated by the stakeholders of the SOI, listed in the table 1 and 2 (stemming from (Dauron et al. 2011)), as well as other information resulting from market studies. These data are illustrative and do not reflect the actual Renault EVs characteristics.

Table 1 Macro needs around the EV

Need ID	Need description
N1	The user expects that the EV's purpose is zero emission.
N2	The user needs to be reassured by usage support and assistance.
N3	The user wants powerful and safe EVs.
N4	The user wants to be able to supply the vehicle with energy.
N5	The automotive manufacturer wants the EV to be an innovation for a large number of customers.
N6	The automotive manufacturer wants to offer an environment-friendly performance to the customers, combined with economic attractiveness and unique services.
N7	Green norms and automotive standards impose their constraints.

Table 2 Refinement of the macro need (N3)

Need ID	Need description
N1-3	The user wants good acceleration level when fully pressing the accelerator pedal.
N3-2	The user does not want to feel the physical limitations of the EV autonomy.
N3-3	The user wants the maximum speed not to exceed a defined safety threshold.
N3-4	The user wants a completely safe behavior in respect to electric risk

There are other inputs to be taken into consideration resulting from market studies. Indeed, the target market for our example is the European market. The business model takes into account the TCO that we define as sum of the acquisition cost, the costs of use (focusing only on the energy consumption by the vehicle) and maintenance costs. The battery, converters, wiring and supervisor technologies are imposed.

Additionally, to move the vehicle at a given speed, the EVP must provide sufficient wheel torque to overcome all resisting forces (rolling resistance, wind speed, road slope and vehicle acceleration), as represented in Figure 4. The torque represents a MOP.

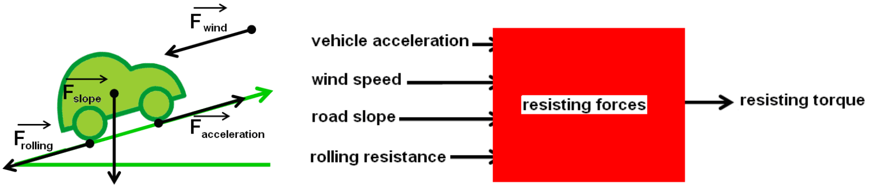

$\overrightarrow{F}_{wind}$
$\overrightarrow{F}_{slope}$
$\overrightarrow{F}_{acceleration}$
$\overrightarrow{F}_{rolling}$
vehicle acceleration
wind speed
road slope
rolling resistance
resisting forces
resisting torque

Fig. 4 Summary of the resisting forces, adapted from (Janiaud 2011)

4.2 Analysis of Inputs of the Problem

The analysis of the inputs of the problem aims at identifying useful and significant information (parameters, variables and constraints) for modeling our optimization problem. The abbreviations used in this section are defined in table 3.

The needs presented in the previous paragraph can be used to derive some requirements that the vehicle shall satisfy. First, concerning the type of vehicle, need (N1) implies that the EV shall not emit CO2. Need (N6) implies that the EV shall not be expensive at purchasing and that it shall not be expensive at use. Therefore, the EV shall consume as little energy as possible. In addition, and following market studies, the EV is defined as an urban and semi-urban vehicle, which can carry up to four passengers (driver included), for a daily utilization that does not exceed an average distance of 60km and at a maximum speed of 120km/h. To design the EVP, we must furthermore consider the characteristics of this type of vehicle that constrain the EVP, namely: aerodynamic coefficient of the vehicle, wheel diameter, weight of empty vehicle (without passengers and luggage), weight of passengers and luggage and the coefficient of rolling resistance.

Concerning the user expectations, needs (N3-1), (N3-2), (N3-3) in Table 2 respectively imply that the EV shall allow a maximum acceleration, have the highest autonomy as possible and limit the vehicle to maximum speed.

On the other hand, the vehicle has to face environmental factors such as resisting forces of the road, wind, vehicle weight, road friction coefficient and weather (temperature, rain, ...). For demonstration purposes, we consider in this study the following parameters in order to calculate the resisting torque: Road slope, Acceleration of gravity, Air density, Temperature, Air pressure.

In order to satisfy need (N7), the EV shall be tested using normalized driving cycles. This allows, for instance, the estimation of the energy consumption by the vehicle. We will use predefined profiles to estimate the energy consumption per vehicle: The New European driving cycle (NEDC), Artemis Urban, Extra-urban and Highway.

Finally, we have to take into consideration the imposed technologies on some components of the EV. These concern mainly the battery technology, as summarized in the Table 3.

Table 3 List of parameters

Parameter	Explications	Measure units	Values
RP	**Road and weather constraints**		
RP1	Road slope	[0 1]	0
RP2	Acceleration of gravity	m/s²	9,81
RP3	Air density	Kg/m^3	1,28
RP4	Temperature	° celsius	25
RP5	Air pressure	bar	
VP	**Vehicle parameters**		
VP1	Aerodynamic coefficient of vehicle.	--	0,7
VP2	Wheel diameter	Meter	0,6
VP3	Weight of empty vehicle (without passengers/luggage)	Kg	1000
VP4	Weight of passengers and luggage	Kg	350
VP5	Coefficient of rolling resistance	--	0,015
STP	**Standards parameters**		
STP1	Driving cycle		NEDC/Artemis
UP	**User parameters**		
UP1	Maximum acceleration	m/s²	0-100km/h(20s)
UP2	Maximum speed	km/h	120
CHP	**Electric charger parameters**		
CHP1	Charger performance	--	~1
CHP2	Weight of the Charger	Kg	10
CHP3	Charger load power	kw	
BP	**Battery parameters**		
BP1	Weight of the battery	Kg	250
BP2	Energy of the battery	Kwh	24
BP3	Battery voltage	Volt	400
BP4	Battery storage Performance	--	~1
BP5	Battery cost	Euros	
COP	**Converters parameters**		
COP1	Weight of the Converters	Kg	25
COP2	Converters cost	euros	
COP3	Performance of the Converters	--	~1

Table 3 (*continued*)

WP	**Wiring parameters**		
WP1	Weight of the wiring	Kg	3
WP2	Wiring cost	euros	
WP3	Electric resistance	ohm	~0
SP	**Supervisor parameters**		
SP1	Weight of the Supervisor	Kg	2
SP2	Supervisor cost	euros	

4.3 Optimization Objectives

On a first approach, some of the system requirements become optimization objectives. In our context, for instance, the requirement *« The EV shall not be expensive»* yields an objective that is *Minimize (the vehicle TCO).* Another requirement *« The EV shall not be expensive at use »* yields another objective: *Minimize (Energy consumption cost).* Finally, the requirement *« The EV shall have as long autonomy as possible»* yields the objective *Maximize (Vehicle autonomy).* Note that in order to minimize the total cost of ownership of the EV, it is clear that we can study other factors such as maintenance costs, the battery rental costs, insurance costs,... In the present case, we focus only on the energy

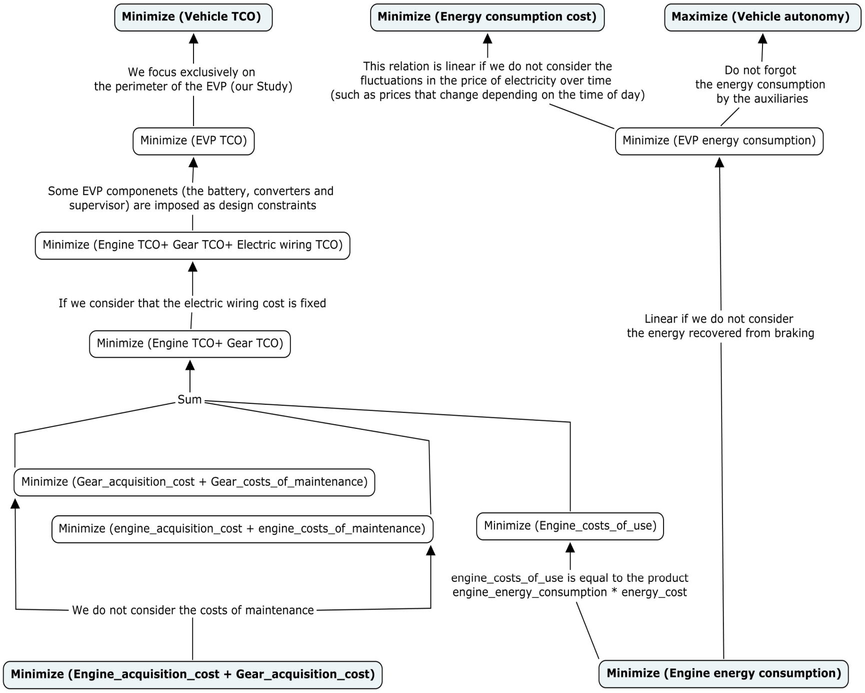

Minimize (Vehicle TCO)
Minimize (Energy consumption cost)
Maximize (Vehicle autonomy)
We focus exclusively on
the perimeter of the EVP (our Study)
This relation is linear if we do not consider the
fluctuations in the price of electricity over time
(such as prices that change depending on the time of day)
Do not forgot
the energy consumption
by the auxiliaries
Minimize (EVP TCO)
Minimize (EVP energy consumption)
Some EVP componenets (the battery, converters and
supervisor) are imposed as design constraints
Minimize (Engine TCO+ Gear TCO+ Electric wiring TCO)
If we consider that the electric wiring cost is fixed
Linear if we do not consider
the energy recovered from braking
Minimize (Engine TCO+ Gear TCO)
Sum
Minimize (Gear_acquisition_cost + Gear_costs_of_maintenance)
Minimize (engine_acquisition_cost + engine_costs_of_maintenance)
Minimize (Engine_costs_of_use)
engine_costs_of_use is equal to the product
engine_energy_consumption * energy_cost
We do not consider the costs of maintenance
Minimize (Engine_acquisition_cost + Gear_acquisition_cost)
Minimize (Engine energy consumption)

Fig. 5 The objectives simplification process

consumption cost for demonstration purposes. We explain in the following paragraphs how we simplify these objectives on the EVP. We summarize objectives simplification process in figure 5.

5 Requirement Derivation and Simplification of Objective Functions

5.1 Requirement Derivation on the EVP

The vehicle requirements are derived on all of its subsystems. Similarly, other requirements on the EVP are derived from enabling systems that interact with the EVP. As some macro-needs defined previously, some requirements are outside the scope of this study. We will address only the requirements that are relevant to our decision problem. Consequently, we take into account the requirements listed on Table 4.

Table 4 The requirements of the EVP

Requirement ID	Requirement description
Req1	The EVP shall be able to propel a vehicle having the following parameters: VP1, VP2, VP3, VP4, VP5.
Req2	The EVP shall be able to propel a vehicle given the following road parameters: RP1, RP2, RP3, RP4, RP5.
Req3	The EVP shall interact with an electric charger that has the following parameters: CHP1, CHP2, CHP3.
Req4	The EVP shall use the electric energy provided by the battery that has the following parameters: BP1, BP2, BP3, BP4, BP5.
Req5	The EVP shall use the Converters that have the following parameters: COP1, COP2, COP3.
Req6	The EVP shall use the Electric Wiring that has the following parameters: WP1, WP2, WP3.
Req7	The EVP shall use the Supervisor that has the following parameters: SP1, SP2.
Req8	The EVP shall meet the user requirements: UP1, UP2.
Req9	The EVP shall be tested with driving cycles NEDC and Artemis: STP1.

5.2 Simplification of the Objective Minimize (Vehicle TCO) on the EVP

If we focus exclusively on the perimeter of the EVP, the objective *Minimize (Vehicle TCO)* becomes *Minimize (EVP TCO).* Moreover, given that the battery, converters and supervisor are imposed as design constraints, we can say that

Minimize (EVP TCO) becomes *Minimize (Engine TCO+ Gear TCO+ Electric wiring TCO).* If we consider that the electric wiring TCO is the cost of its acquisition and it is fixed, the function becomes *Minimize (Engine TCO+ Gear TCO).*

We defined previousely the TCO as the sum of the costs of acquisition, use and maintenance. Therefore, we have to *Minimize(engine_acquisition_cost + engine_costs_of_use + engine_costs_of_maintenance)* and *Minimize(gear_acquisition_cost + gear_costs_of_maintenance).* Given that the engine_costs_of_use is equal to: engine_energy_consumption * energy_cost, and we have to *Minimize (Engine energy consumption),* explained previously, this will *Minimize (Engine_costs_of_use).* Therefore, it is sufficient to *Minimize (engine_acquisition_cost + engine_costs_of_maintenance)* in order to *Minimize (Engine_cost).*

In this study, we do not consider the costs of maintenance of the engine and the gear. The objective *Minimize (Engine_TCO + Gear_TCO)* becomes *Minimize (Engine_acquisition_cost + Gear_acquisition_cost).*

This should not be confused with the total cost of EV use. If we focus only on the electric energy consumption (EV_energy_consumption), the cost of EV use is the product of the energy cost (that may change during day and season) multiplied by EV_energy_consumption. The latter is the sum of the energy consumed by all the components of the EVP, taking into account the performance of all the component. To this, we would have to add the energy consumption by the auxiliaries, such as lights, air conditioning system and radio.

5.3 Simplification of the Objective Maximize (Vehicle Autonomy) on the EVP

There exists a dependency between the two functions: *Minimize (Energy consumption cost)* with *Maximize (Vehicle autonomy).* This relationship is linear if we do not take into account fluctuations in the price of electricity over time (such as prices that change depending on the time of day). Therefore, it is sufficient to *Maximize (Vehicle autonomy)* in order to *Minimize (Energy consumption cost).* Given that the battery technology is imposed, the function *Maximize (Vehicle autonomy)* becomes *Minimize (EVP energy consumption).*

The EVP energy consumption is the sum of the energy consumption by all its components. Since in our context the Converters are imposed and the consumption of the other components is negligible, then *Minimize (EVP energy consumption)* becomes *Minimize (Engine energy consumption).* This implication is completely true if we do not consider the energy that may be recovered during braking.

To calculate the electric energy consumption by the engine, we will adapt the solution proposed in (Janiaud 2011), as follows:

<u>Given:</u>
t= time
UP[t] is the vehicle speed at t on a given driving cycle.

$$M = VP3 + VP4 + CHP2 + BP1 + COP1 + WP1 + SP1 + Var4 + Var7 \tag{2}$$

Result, Gear_torque, Engine_torque, Engine_omega : intermediate variables.

The estimation of the resisting forces is performed as follows.

- *Force related to vehicle acceleration:*

$$F1 = \left(VP2\big/2\right) \times \left(M \times {dt\left(STP1(t) \times \left(1\big/3.6\right)\right)}\big/{dt} \right) \tag{3}$$

- *Force related to wind speed:*

$$F2 = \left(VP2\big/2\right) \times \left(1\big/2 \times RP3 \times VP1 \times \left(STP1(t) \times \left(1\big/3.6\right)\right)^2\right) \tag{4}$$

- *Force related to road slope:*

$$F3 = \left(VP2\big/2\right) \times \left(M \times RP2 \times RP1\right) \tag{5}$$

- *Force related to rolling resistance:*

$$F4 = \left(VP2\big/2\right) \times \left(M \times RP2 \times VP5\right) \tag{6}$$

With

$$\mathrm{Re}\,sult = F1 + F2 + F3 + F4 \tag{7}$$

$$Gear\_torque = \{\ \mathrm{Re}\,sult \text{ if } |\,\mathrm{STP1(t)} \times \left(1\big/3.6\right)| \geq 0.1 \tag{8}$$

$$Gear\_torque = \{ 0 \text{ if } |\,\mathrm{STP1(t)} \times \left(1\big/3.6\right)| < 0.1 \tag{9}$$

Where Gear_torque represents the resisting torque which the EVP has to face in order to move the vehicle. It represents all the resistant forces, given the vehicle speed STP1 (t) with which it must advance (see Figure 2).

Then we reduce the resisting forces, as follows.

$$Engine\_torque = \left(Gear\_torque\big/Var9 \right) \tag{10}$$

$$Engine\_omega = \left(STP1(t) \times Var9 \times \left(1\Big/\left(\left(VP2\big/2\right)\big/3.6\right) \right) \right) \tag{11}$$

Where

- Engine_torque represents the torque against which the engine has to face in order to move the vehicle. Engine_torque corresponds to Gear_torque divided by Gear_ratio.
- Engine_omega represents the rotational speed of the engine in order to move the vehicle with the speed required by the user.

The energy consumption by the engine « Engine_energy_consumption » depends on the Engine_torque, Engine_omega, Engine_map and t. Indeed, we calculate the energy losses as a function of the engine torque and the engine speed:

$$E = \left(Pelect \times t\right) \quad \text{and} \qquad Pelect = \left(Pmeca\big/\mu \right) \tag{12}$$

Where

- E is Engine_energy_consumption
- Pelect is the electric power
- μ is the engine power according to a functioning point on the engine map.
- Pmeca is the mechanic power given by the relation:

$$Pmeca = \left(\text{Engine\_torque} \times \text{Engine\_omega}\right) \tag{13}$$

The engine map gives the correspondence between Engine_omega and Engine_torque. It serves to calculate the electric power given a functioning point. The Engine_energy_consumption represents the energy consumed by the engine over a given time interval. Indeed, the electrical energy « E » consumed by an electric component is equal to the product of the electric power « Pelect » by the duration « t » of its operation. Note that when we choose an engine, we must take into account the battery voltage. An example of engine map is shown on Figure 6.

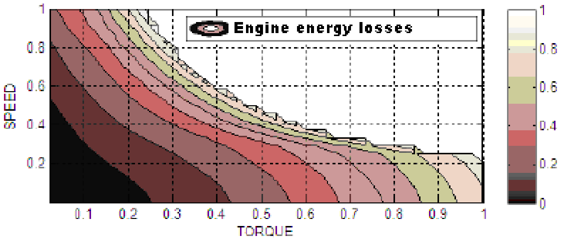


Fig. 6 Example of an engine map (Adapted from (Janiaud et al. 2009))

NB. The maximum speed and acceleration (UP1, UP2) required by the user must be realizable by the engine. In order to verify this, we create a deriving cycle from these two data and make the same calculation as a driving cycle such as NEDC.

Finally, given a driving cycle, an estimation of the total EV autonomy is given by these equations:

$$\text{EV\_autonomy\_by\_distance} = \left\| \left({}^{\text{BP2}} \!\big/ \!{}_{\text{EV\_energy\_consumption}} \right) \right\| \times cycle\_distance \tag{14}$$

$$\text{EV\_autonomy\_by\_duration} = \left\| \left({}^{\text{BP2}} \!\big/ \!{}_{\text{EV\_energy\_consumption}} \right) \right\| \times cycle\_duration \tag{15}$$

For example, in NEDC, Cycle_disance~11km and Cycle_duration~20minutes.

6 Synthesis of Simplified Functions to Optimize

Coming back to our original problem, we have to choose an engine and a gear in order to reach two objectives:

- *Minimize (Engine acquisition cost+ Gear acquisition cost).*
- *Minimize (Engine energy consumption).*

taking into consideration the parameters and the different variables related to the engine and to the gear summarized respectively in Table 3 and Table 5.

Table 5 List of the variables and their domain value constraints

Variables	Explications	Measure units
Engine variables		
• Engine map	that contains:	
• Var1	• Engine torque	N*m
• Var2	• Engine rotation speed	Rad/s
• Var3	• Engine power	Watt
• Var0	• Engine cost	Euros
• Var4	• Weight of the engine	Kg
• Var5	• Engine voltage	volt
Gear variables		
• Var6	• Gear performance	--
• Var7	• Weight of the gear	Kg
• Var8	• Gear cost	Euros
• Var9	• Gear ratio	--

7 Mathematical Representation

In this section, we represent our optimization model following the equation (1) formulation:

Parameters "p":

P (VP, EP, CHP, RP, BP, COP, WP, SP, UP,STP) is a vector of parameters , where :

- *VP (VP1, VP2, VP3, VP4, VP5)* is the vector of the vehicle parameters.
- *EP (EP1, EP2)* is the vector of the charging station parameters.
- *RP (RP1, RP2, RP3, RP4, RP5)* is the vector of the road/weather parameters.
- *CHP (CHP1, CHP2, CHP3)* is the vector of the charger parameters.
- *UP (UP1, UP2)* is the vector of the user parameters.
- *STP (STP1)* is the vector of the user/standard parameters.
- *BP (BP1, BP2, BP3, BP4, BP5)* is the vector of the battery parameters.
- *COP (COP1, COP2, COP3)* is the vector of the converters' parameters.
- *WP (WP1, WP2, WP3)* is the vector of the wiring parameters.
- *SP (SP1, SP2)* is the vector of the supervisor parameters.
- *t* is the time.

Variables « x » :

V (VE, VG) is the vector of the variables where:

- *VE (Var0, Var1, Var2, Var3, Var4, Var5)* is the vector of the engine variables.
- *VG (Var6, Var7, Var8, Var9)* is the vector of the gear variables.

Constraints « g » and « h »: (examples)

$g_1(BP1, COP1, WP1, SP1, Var4, Var7) = (BP1 + COP1 + WP1 + SP1 + Var4 + Var7) < 400$

$h_1(BP3, Var5) = BP3 - Var5 = 0$

Functions « J » to minimize :

J1: *engine_energy_consumption = J1(VP1, VP2, VP3, VP4, VP5, BP1, RP2, RP3, COP1, WP1, SP1, RP1,STP1,Var9,Var4,Var7, engine_map, t)*

= equa(Engine_torque, Engine_omega , engine_map, t)[2]

Given the equations (2-3-4-5-6-7-8-9-10-11)

J2: *engine and gear acquisition costs = J2 (Var0, Var8) =Var0 + Var8*

For the sake of clarity, we do not list all the constraints.

8 Tests, Results and Brief Discussion

In this section, we show simulation results of our example, simplified to compare two different engines and one same gear. The choice of the engine will be made according to the two criteria defined above (the acquisition cost of the engine and its energy consumption). The parameters values and measure units are shown in the Tables 3 and 5. The values of the variables of the gear we use are:

- Var9 = 10.4
- Var6 = ~1
- Var7 = 10
- Var8 =200

Figure 7 shows the simulation results: the acquisition cost of the two engines (in euros) and their energy consumptions (kWh) on the four driving cycles.

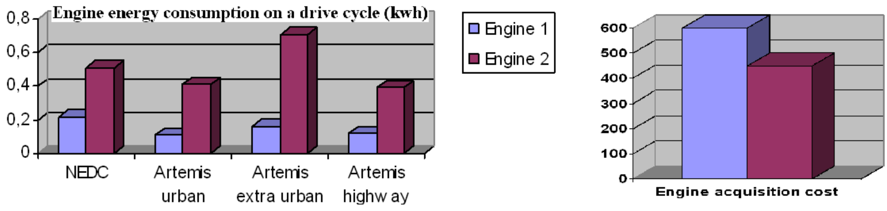


Fig. 7 Simulation results

Table 6 gives an estimation of engine costs of use (focused on the energy consumption by the engine on the driving cycle is NEDC) given the cost of 1kwh = 0,1209 Euros [3].

[2] This equations system is resolved with a matlab-simulink model.

Table 6 Engine costs of use estimation

Engine	Engine acquisition cost (euros)	Engine energy consumption on NEDC, distance= 11 km (kwh)	Engine energy consumption on NEDC, distance= 1000 km (kwh)	Engine cost of use – distance = 1000 km (euros)
Engine 1	600	0,2162	19,6545	2,376
Engine 2	450	0,5098	46,3454	5,6031

These results show that Engine 1 would be the better choice according to the criteria defined above. Furthermore, according to this short study, it appears that it would be beneficial to broaden the optimization problem by a parametric sensitivity analysis to the scope of the overall vehicle. Indeed, other parameters can affect the overall vehicle autonomy. Some parameters can be varied and see the impact on the energy consumption. Some interesting results have been presented in (Janiaud 2011). In the context of its study, the results show that the most sensitive parameters (on the overall vehicle) are the Aerodynamic coefficient of vehicle, Coefficient of rolling resistance and the Weight of the vehicle.

Continuing on the same example, given the battery capacity, we calculate the EV autonomy (14) and its cost of use per 100km. We use the same driving cycles. Also, we focus only on the electric energy consumption without other fees such as insurance, battery rent, highway toll,... Note, as we said previously, the EV cost of use is the product of the energy cost multiplied by the EV energy consumption. The latter represents the sum of the energy consumed by all the components of the EVP (considering the performance of all its components) and by the auxiliaries, such as lights, air conditioning system and radio…). We present the results in the following figure 8.

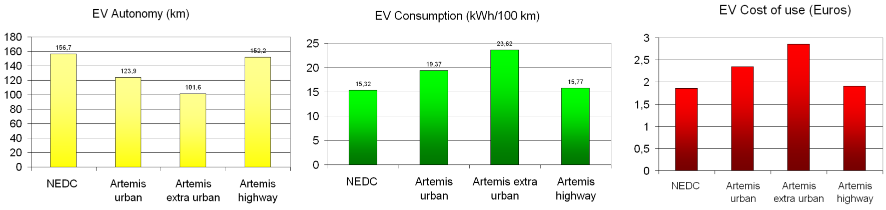


Fig. 8 The EV autonomy, energy consumption and cost of use estimations

Widening the study scope by considering all the fees related to the whole Electric Vehicle TCO (such as subscription and taxes, insurance, battery rent, highway toll,....) is challenging. To study such a complex system, the assessment of all its lifecycle would yield other promising results.

[3] Cost of 1 kwh, without subscription and taxes fees, France, august 1st, 2011. Source `http://bleuciel.edf.com`

9 Conclusion

This paper shows the importance and utility to use an architecture framework in order to organize and structure all the views that allow studying the system of interest with a holistic approach. Additionally, multi-objective optimization models in a complex systems engineering context help considerably in trade-off analysis and decisions making. The data used in the optimization problem stem from the system descriptions contained in this architecture framework, which separates the solution design process from the problem definition. Indeed, the design constrains are easily distinguishable from the design variables. On the other hand, this paper shows the importance of taking into account the lifecycle of the system of interest in the optimization problems modeling. The assessment of the properties of the lifecycle would yield some parameters that might contribute efficiently in the integration of the system in its environment (See an exhaustive list of lifecycle properties in (de Weck et al. 2011)).

Concerning the case study we presented, it is important to note that some criteria that might contribute for the success of electric vehicles depend not only on their characteristics but also on other parameters derived from their environment. Also, for example, studying some properties such as scalability (millions of EVs), flexibility (if new needs will appear), modularity (easy to adapt for different target markets) and sustainability (given European policies) is challenging but important. Some design constraints such as regulation laws could be considered as design variables in a continuously and increasingly uncertain complex environment.

References

Andersson, J.: A survey of multiobjective optimization in engineering design. Technical Report LiTH-IKP-R-1097, Department of Mechanical Engineering, Linkoping University, Linkoping, Sweden (2000)

Benkhannouchel, S., Penalva, J.M.: Systemic Approach for Supervision Systems Design. In: Conference Proceedings Systems, Man and Cybernetics, International Conference on 'Systems Engineering in the Service of Humans', vol. 1, pp. 395–400 (October 1993)

Chalé Góngora Hugo, G., Dauron, A., Gaudré, T.: A Commonsense-Driven Architecture Framework. Part 1: A Car Manufacturer's (naïve) Take on MBSE. Accepted for Publication in Proceedings of the INCOSE International Symposium, Rome. INCOSE (2012)

Chatel, V., Feliot, C.: Functional analysis for safe and available system design. In: IEEE International Conference Systems, Man and Cybernetics (2004)

Coello Coello, C.A.: Multiobjective Optimization Website and Archive (last update February 23, 2010), http://www.lania.mx/~ccoello/EMOO/

Dauron, A., Doufene, A., Krob, D.: Complex systems operational analysis - A case study for electric vehicles. In: International Conference CSD&M, Poster session, Paris (2011)

De Weck, O.L.: Multiobjective optimization: History and promise. In: China-Japan-Korea Joint Symposium on Optimization of Structural and Mechanical Systems (2006)

De Weck, O.L., Roos, D., Magee, C.L.: Engineering Systems: Meeting Human Needs in a Complex Technological World. The MIT Press editions (October 2011)

INCOSE, Systems Engineering Handbook. A guide for system lifecycle processes and activities. International Council on Systems Engineering (INCOSE), San Diego, CA (January 2010)

Janiaud, N., Vallet, F.X., Petit, M., Sandou, G.: Electric Vehicle Powertrain Architecture and Control Global Optimization. In: Electric Vehicle Symposium, Stavanger, Norway. Publication de l'article dans WEVA Journal (World Electric Vehicle Association), vol. 3 (2009) ISSN 2032-6653

Janiaud, N.: Modélisation du système de puissance du véhicule électrique en régime transitoire en vue de l'optimisation de l'autonomie, des performances et des coûts associés. PhD thesis, SUPELEC, France (2011)

Kim, I.Y., De Weck, O.L.: Adaptive weighted sum method for multiobjective optimization: a new method for Pareto front generation. Structural and Multidisciplinary Optimization 31, 105–116 (2005)

Kim, I.Y., De Weck, O.L.: Adaptive weighted-sum method for bi-objective optimization: Pareto front generation. Structural and Multidisciplinary Optimization 29(2), 149–158 (2005)

Krob, D.: 'Eléments d'architecture des systèmes complexes. In: Appriou, A. (ed.) Gestion de la Complexité et de l'Information Dans Les Grands Systèmes Critiques, pp. 179–207. CNRS Editions (2009)

Krob, D.: Enterprise Architecture, Modules 1-10, Ecole Polytechnique. Personal communication (2009-2010)

Meinadier, J.P.: Ingénierie et intégration de systèmes. Editions Hermès (1998)

Meinadier, J.P.: Le métier d'intégration de systèmes. Editions Hermès-Lavoisier (2002)

Marler, R.T., Arora, J.S.: Survey of multi-objective optimization methods for engineering. Structural and Multidisciplinary Optimization 26(6), 369–395 (2004)

Penalva, J.M.: La modélisation par les systèmes en situations complexes. PhD Thesis, Université de Paris 11, Orsay, France (1997)

Smaling, R.M.: System architecture analysis and selection under uncertainty. PhD thesis, Massachusetts Institute of Technology (2005)

Smaling, R.M., De Weck, O.L.: Fuzzy pareto frontiers in multidisciplinary system architecture analysis. AIAA Paper 4553 1–18 (2004)

Chapter 8
Requirement Management for Complex Systems, a Critical Element of the Integration Process

Nicolas Chapron, Michel Luttmann, and Christian Blanchet

Abstract. The Laser MégaJoule (LMJ) is a laser beam facility dedicated to the inertial confinement fusion program. Its construction started in 2003 at the French Atomic Energy Commission CESTA centre located near Bordeaux. This facility is an extremely complex system, designed using a concurrent engineering process, with state of the art performances. Requirement management is an essential part of the project management of such a complex system, characterized by the integration of multiple heteroclite subsystems and products. This activity ensures the mapping between the system requirements of the project's end user and the lower-level specifications for subsystems and product suppliers. Using adequate methodology and tools, requirement management acts as a guide throughout the whole integration process, from the first steps of the definition phase to the last part of the project qualification and delivery. As an illustration, this paper presents how the project team (CEA/DAM as the prime contractor and Areva TA as its support) handles alignment performance engineering for the LMJ project using a specific tool based on software package DOORS made by the Telelogic company.

Keywords: Requirement management, Complex systems, Specifications, Methodology, Integration, Performance modeling, Doors.

Nicolas Chapron
AREVA TA
Site LMJ, Route de Marcheprime, CS 80480, F-33116 Le Barp Cedex,
France
e-mail: nicolas.chapron@projet-lmj.org

Michel Luttmann
CEA
CEA DAM, CESTA, BP N°2, F-33114 Le Barp Cedex,
France
e-mail: michel.luttmann@projet-lmj.org

Christian Blanchet
AREVA TA
BP 17, F-91192 Gif-Sur-Yvette,
France
e-mail: christian.blanchet@areva.com

1 About Requirement Management

Requirement management is one of the major concerns in project management. Its main objective is to ensure that the requirements desired by the end user are fully realized by the prime contractor. Therefore, managing the requirements means not waiting until the qualification phase to discover conformity problems. Of particular concern are system performances (quantitative technical requirements) whose realization is accomplished using different techniques, different trades and different suppliers, whose global monitoring is hard to achieve and whose qualification can only be done at the very end of the project.

The concept of requirement management is based on constant monitoring throughout the entire project's lifecycle, strengthened by different assessment tools. The whole concept is based on describing the process step-by-step with a breakdown tree whose objectives are:

1. To link the system requirements to the subsystems/products specifications,
2. To explain the impact of each subsystem/product specification on the different system requirements.

In reality, it means one has to master an ascendant calculation of system performance from the low-level specification values. We will describe the ascendant calculation part later because it represents the quality of the integrated model.

Considering this as a V-lifecycle model, this activity can be separated into four distinct steps (Figure 1):

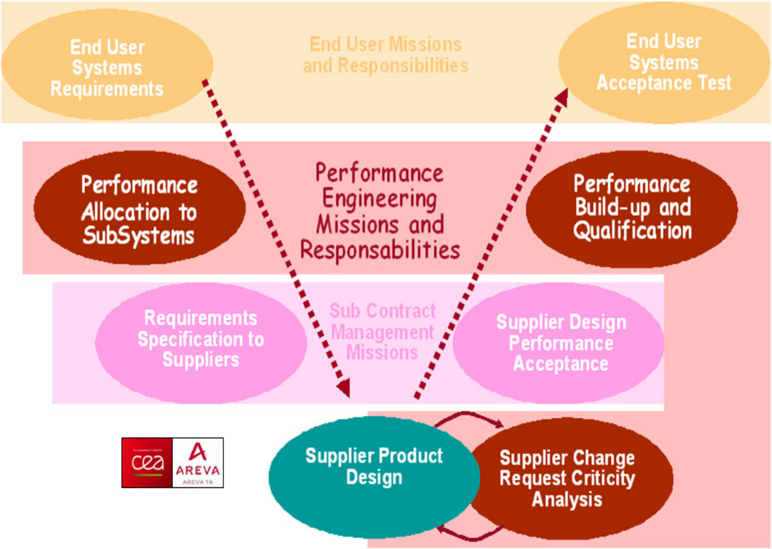

End User Systems Requirements
End User Missions and Responsibilities
End User Systems Acceptance Test
Performance Allocation to SubSystems
Performance Engineering Missions and Responsabilities
Performance Build-up and Qualification
Requirements Specification to Suppliers
Sub Contract Management Missions
Supplier Design Performance Acceptance
cea
AREVA
AREVA TA
Supplier Product Design
Supplier Change Request Criticity Analysis

Fig. 1 Performance V-lifecycle model

- During the specification phase:
  - The end user has to identify and to rank the system requirements.
  - At this stage, a label and a number has to be given to each requirement to avoid any misunderstanding.
- During the design phase:
  - The requirement breakdown structure has to be completed during the progression of each design. Each requirement needs to be allocated to different contributors, contributors may then have sub-contributors one level down. This breakdown operation should be pursued until the contributors' level is coherent with the subsystems or the products' specifications. To calculate performances, the ascendant calculation model has to be established during this phase.
  - One important constraint is to respect a lower level contributors' coherence with the products' specifications. Strictness is important: each lower level contributor must have its own separate and corresponding product specification.
  - A useful study to do at this stage is to synchronize the breakdown steps with each step of the technical studies and their documentation. If this synchronization is done properly, the requirement breakdown structure can become the justification map of the project.
- During the supplier product design phase:
  - The ascendant calculation model permits to repeatedly update the calculation of system performances. This work requires a follow-up of the different contributors' progress during the product design lifecycle. These supplied milestone values will be carried across into the calculation model.
  - The ascendant calculation model enhances the change request analysis process. Non-conformities can be analyzed with a view of the global effect on the top-level requirements and taking into account the status of the other products involved in the requirement (sometimes enabling trade-offs).
  - A sensitivity analysis can give first order coefficients in order to quickly estimate the effect of a non-conformity or to focus efforts on critical contributors.
- During the system qualification phase:
  - The breakdown structure is a powerful tool to define and organize the system qualification plan.
  - The ranking of the system performances and the sensitivity analysis enables a focus on the major qualification tests.

Such requirement management methodology is quite difficult to put into practice. Small projects generally don't have time and money for this kind of subject and large projects are too detailed to permit such work on every requirement. Two variables permit to adjust requirement management work to the size and budget of the project:

1. The ranking of the requirements (cf. specification phase) helps to focus the breakdown work only on the major requirements.
2. The complexity of the ascendant calculation model (for performances) can be adapted according to the rank, flexibility and state of the art of the required performance.

The following presents an illustration of this methodology applied to a large scientific instrument project: the Laser Mégajoule (LMJ) Project.

2 About the Laser Megajoule Project

The Laser Mégajoule (LMJ) is one of the major tools of the French Simulation Program. The main objective is to ensure the relevance and control of computing software that simulates the operation of a nuclear weapon. The final user and prime contractor is the CEA (French Atomic Energy Commission). This project is entirely funded by the French military budget for about 3 Billion Euro (complete cost) for 15 years but it will also be available to the French and European scientific community. Indeed, the LMJ and its prototype (the LIL) are unique ways to simulate the phenomena that occur under extreme conditions of temperature, pressure and density, such as those encountered in stars and planetary cores.

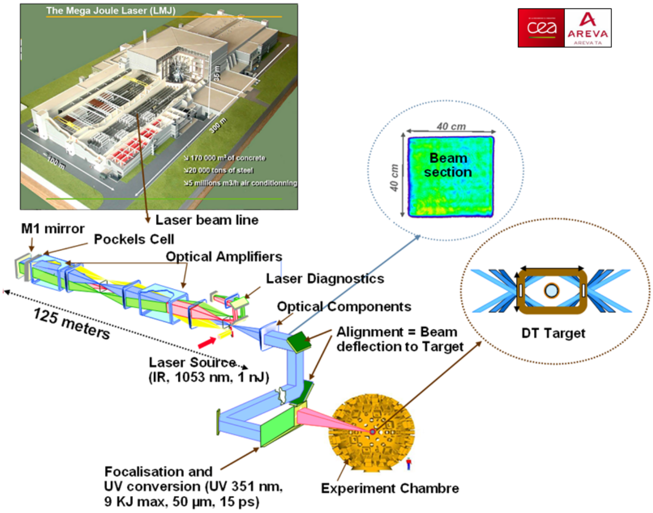


Fig. 2 The Laser MégaJoule concept

Technically, the LMJ is an inertial confinement fusion facility designed to focus an UV (351 nm) laser short pulse of 1.8 MJ on about 1 cm size deuterium-tritium (DT) target located in an experiment chamber. The laser UV beams are obtained using a four-pass infrared amplifier and a laser beam wavelength conversion. The experiment chamber is a 10 meter diameter aluminum sphere under high vacuum (10-6 bars). The whole experiment is monitored with IR and UV laser diagnostics for the laser properties and with plasma diagnostics for the fusion measurements.

Physically, the LMJ is composed of 5 main halls:

- The Experiment Hall (the central one) contains the experiment chamber, the experimental diagnostics and the final stages of the laser process (mirrors deviating beams towards target and UV conversion and focusing).
- The 4 Laser Halls are designed to include 7 or 8 laser lines of 8 laser beams each, leading to a total of 240 laser beams grouped in 30 bundles. The current plan is to only install 22 bundles.

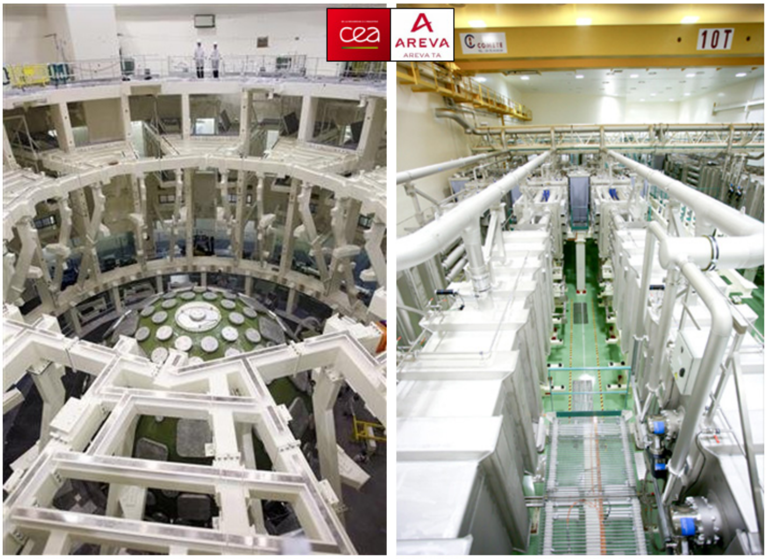


Fig. 3 LMJ views: Experiment Hall and the chamber (left); one of the 4 Laser Halls (right)

Temporally, the program is organized in 2 steps:

1. The LIL is a prototype equivalent to 1/30 of LMJ. It is operational since March 2002.
2. The LMJ is the main facility. The building construction started in 2003 for 5 years and the laser installation and assembly is in progress. The first laser alignment is expected in 2012, the operation will start in 2014 and the first fusion will be in 2016.

The LMJ concept requires numerous equipments, each one involving very different techniques: optics, lasers, neutronics, mechanics, radiation shielding, vacuum, cryogenics, electronics, HVAC, I&C... Because a lot of the LMJ requirements are state of the art, many competent suppliers are necessary (hence, more than 35 industrial contracts are identified).

To manage all these industrial contracts, the LMJ project is organized using a double classification:

1. Four subsystems: building and utilities, lasers, experimental equipments and I&C.
2. Three different engineering levels: the system engineering whose responsibility is to define the system requirements for the subsystems, then the subsystems engineering have to pursue the allocation of work for suppliers' specifications.

In this paper we will focus on the alignment performance: the LMJ facility has to focus the laser beams, after 450 meters of propagation, on a target to a precision under 50 microns. Indeed the alignment performance is representative of the project's complexity and it fully illustrates the ascendant calculation modeling.

To obtain such a performance, it is important to master different error sources: optical design, actuator accuracy, image processing, thermal drift and mechanical stability. All these errors must be identified during the breakdown of the alignment performance.

We can identify 490 product specifications in accordance with the 14 system performances for the alignment subject alone. The resulting breakdown structure is described completely using a specific software (DOORS, by Telelogic) which is a relational database commonly used for requirement management in the aeronautical or automotive sectors.

As shown in Figure 4, where red arrows indicate the system alignment entity's outputs, there are three kinds of outputs:

1. Specifications towards a product, which are to be included in the product specification documents,
2. Specifications towards subsystems entities which are to be allocated one level down with a design and justification study towards product specifications (blue arrows in Figure 4),
3. Specifications towards complex models which are to be allocated with specific technical model software (NASTRAN in this example for studying the building's stability) in order to add some specifications towards the different LMJ products (green arrows in Figure 4).

Finally, another constraint has to be considered, different industrial calendars are not synchronized, some suppliers are in the testing phase when others are just beginning design studies.

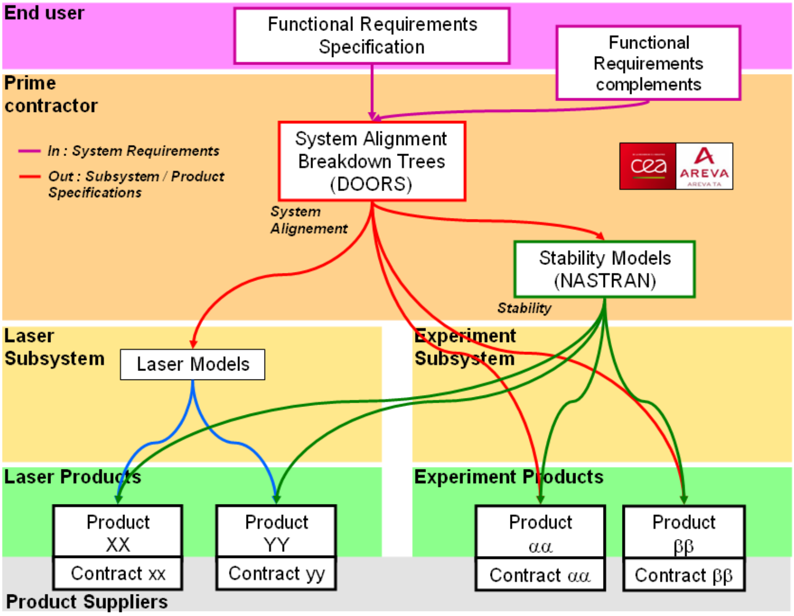


Fig. 4 System alignment performances organization

For all these reasons, simply using a requirement management tool (such as Doors) was not sufficient to handle the LMJ project's objectives. Thus, different contributions were made to add to Doors capacities. The three major contributions described in the next paragraphs were:

1. The addition of a simple marker to directly link suppliers specifications with systems requirements,
2. The encoding of the performances ascendant calculation model,
3. The use of specific milestone values on the follow-up of suppliers' specifications.

3 First LMJ Contribution: The Gene Marker

Alignment performances can be separated into the following different objectives: the laser alignment, the target alignment and the plasma diagnostics alignment. The alignment breakdown structure allocates these values between different contributors and sub-contributors towards the lower level specifications, as illustrated below.

With such a complex organization of the contributors, one of the LMJ difficulties is to find a way to quickly link the products' specifications (the ones who are discussed everyday with the suppliers) with the system performances impacted. As the whole structure is contained in the Doors model which can not be shared with

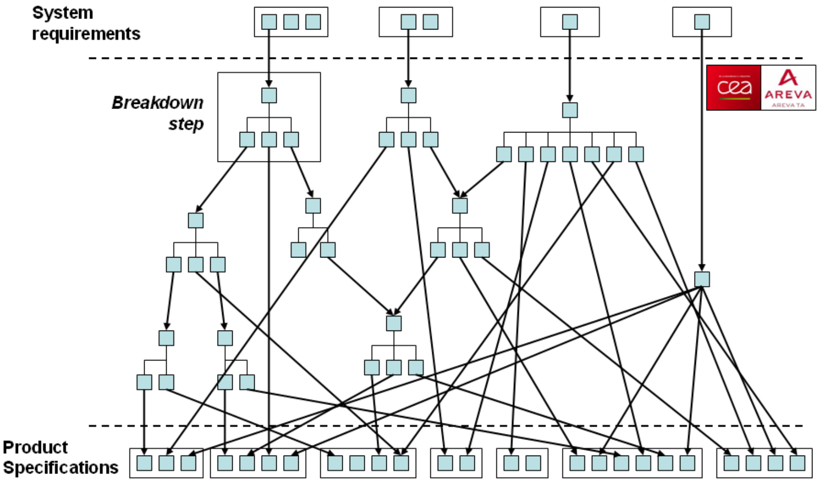


Fig. 5 Breakdown trees with traceability

everybody involved in the project, one useful tool to implement is the traceability matrix that indicates which product specification impacts which system requirement.

In order to simplify the software process generating this matrix, we introduced to the LMJ project a marker (a ‘gene’) for each system requirement and all its contributors and sub-contributors. These genes are defined for each alignment system performance and then reported towards each first rank contributors of the performance. When a contributor impacts multiple performances, its genome contains all the genes associated with the performances impacted. The operation is repeated at every breakdown step: each contributor transmits its own genome towards its sub-contributors.

In Figure 6, the genes are represented by colors:

- Product specifications with one color will only impact the associated colored system requirement.
- Product specifications with multiple colors will impact multiple system requirements.

One of the greatest advantages of this genetic transcription is to directly see from the product specification the exact list of system performances impacted.

To implement such a marker, specific DXL code has been implemented in the Doors software. Moreover, due to specific project management needs, other DXL codes based on the genome have been implemented in the Doors software to create the following functions:

- Change request analysis: One can extract from a product specification the list of system performances impacted.

- System performance appraisal: One can extract from a system performance the whole list of product specifications to check and number their conformity status or their progression level.
- Criticality analysis: One can extract from a system performance the whole list of product specifications to check and number their criticality level.

One must be aware that the gene marker only indicates a link with the system requirements. In order to quantify the impact of a contributor on a system performance, an ascendant calculating model has to be implemented.

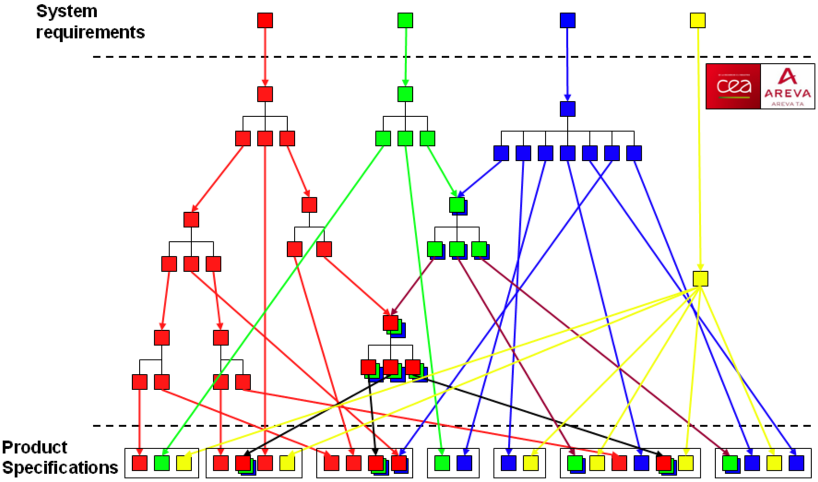


Fig. 6 Gene markers

4 Second LMJ Contribution: The Ascendant Calculation

The ascendant calculation model is established during the design phase in order to allocate lower level contributors in accordance with the system performances objectives. This calculation represents the performance modeling quality. Whichever the software tools one may choose, from off-the-shelf software to specific tailor-made developments, it will have to be capable of the ascendant calculation.

This ability to easily build-up the system performances from the product specifications involves requirement management as a major actor in the change request analysis process. Indeed, by changing the lower-level specification value, the calculation can report the impact throughout the structure and give the final impact on the system performances as illustrated in Figure 7.

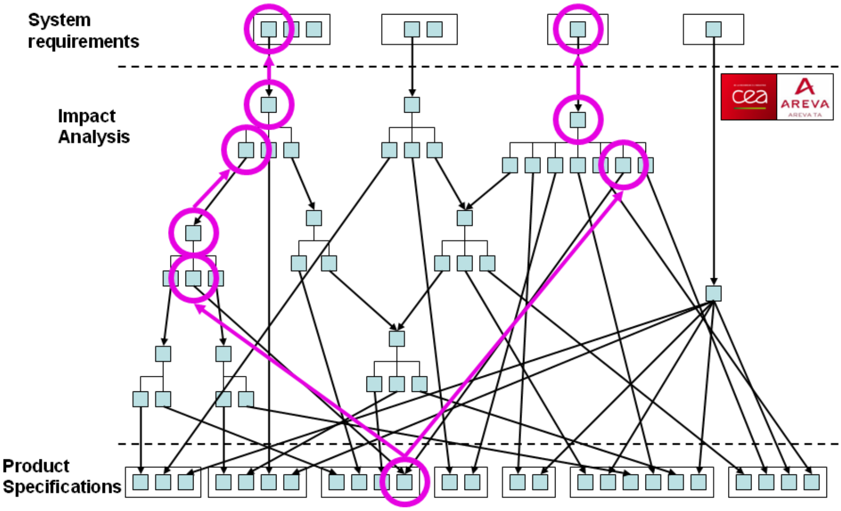


Fig. 7 Change request analysis

The ascendant calculation helps to enable a sensitivity analysis which gives:

- First order coefficients (between the product specification and the impacted system performances) which are useful in quickly giving the effects of non-conformities without the need for calculation software.
- Criticality ranking of the lower level specifications: the calculation part can provide impact analysis for a generic specification evolution (+10% for instance) and rank specifications by considering the percentage impact on the performances.

Moreover, the ascendant calculation allows definition of some intermediate levels of system qualification. This helps to manage a step-by-step integration of the whole system, as shown in Figure 8.

Sometimes, when the top-level requirement is difficult to qualify, the ascendant calculation can provide an adequate justification. The criticality ranking enables a focus on the major qualification tests and allows time optimization for the entire qualification phase.

The global ascendant calculation is created by collating the local models used during the design phase and connecting them together for their path through the whole breakdown tree.

Three kinds of local models can be differentiated:

1. Basic models: The calculation involves basic operations like addition, multiplication, unit conversion, quadratic sums and requirement software can easily handle this calculation. These operations are implemented directly into the requirement software (DXL source code in our case).

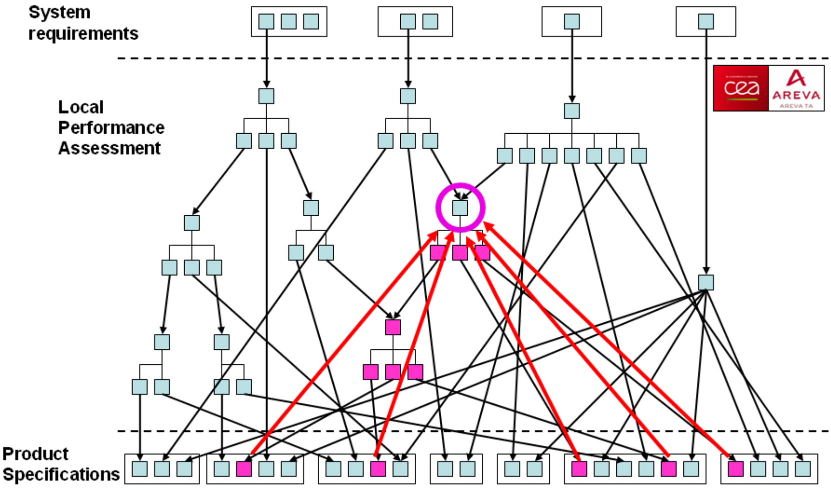


Fig. 8 Intermediary performance qualification

2. Complex linear models: The definition calculation can only be made by a specific modeling tool (optical design software) but once this has been done evolutions can be modeled in the requirement software by using first-order equations.
3. Specific complex models: The calculation has to be modeled for every product change with a specific tool and requirement software will only record this calculation's output. One has to develop gateways with these specific tools.

The LMJ laser beam alignment performance is a typical example of these multiple calculation models. In order to target the DT capsule, laser beam alignment has to fulfill:

- Alignment process requirements: about 1/3 of the global alignment performance depends on the alignment process (sensors, actuators and image treatment errors), managed by the laser subsystem teams (Figure 4).
- Stability requirements: nearly 2/3 of the global alignment performance depends on the building's stability (ground vibration and thermal drift), modeled with powerful finite element analysis such as NASTRAN.

The NASTRAN model manages the whole facility's structure (building, equipment, optics). It has already been qualified with the LMJ's prototype 'LIL' and is frequently updated with environmental data taken within the LMJ. Figure 9 shows an example of one laser focusing module as modeled by NASTRAN.

In order to translate these NASTRAN results into laser pointing errors on the DT target, we need to put the NASTRAN result through an optical model of the laser (made with ZEMAX software) and then enter these results into the alignment breakdown structure. In practice, NASTRAN is reiterated whereas the ZEMAX model is approximated to the first order (available at this scale of vibrations) with

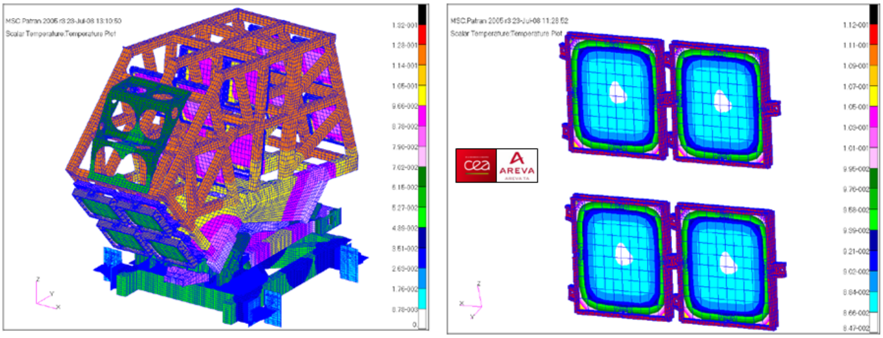

MSC Patran 2005 r3 23-Jul-08 13:10:50
Scalar Temperature:Temperature Plot
MSC Patran 2005 r3 23-Jul-08 11:28:52
Scalar Temperature:Temperature Plot
cea
AREVA

Fig. 9 NASTRAN modeling of the focusing and UV conversion module

in the DOORS alignment breakdown structure and the remaining calculations are made within DOORS (specific calculations implemented with DXL source code) as described in Figure 10 where red indicates the NASTRAN results, blue represents the external data (ZEMAX) memorized in the Doors model and black highlights Doors calculations.

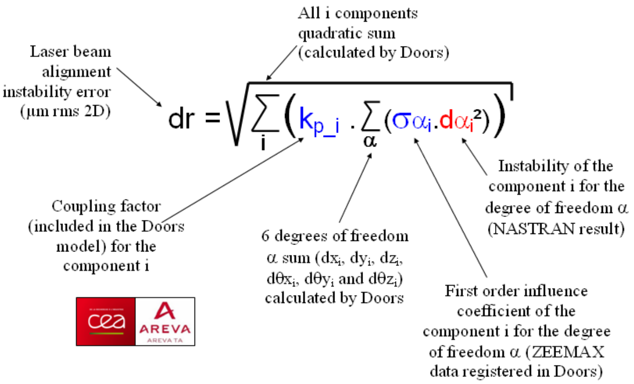

All i components quadratic sum (calculated by Doors)
Laser beam alignment instability error (µm rms 2D)
$dr = \sqrt{\sum_{i}\left(k_{p\_i} \cdot \sum_{\alpha}(\sigma\alpha_i . d\alpha_i^2)\right)}$
Instability of the component i for the degree of freedom α (NASTRAN result)
Coupling factor (included in the Doors model) for the component i
6 degrees of freedom α sum (dx$_i$, dy$_i$, dz$_i$, dθx$_i$, dθy$_i$ and dθz$_i$) calculated by Doors
First order influence coefficient of the component i for the degree of freedom α (ZEEMAX data registered in Doors)
cea
AREVA
AREVA TA

Fig. 10 DOORS calculation example

5 Third LMJ Contribution: Milestone Value Monitoring

As the ascendant calculation of the system performances is one of the keys to ensure the products' integration success, it should be updated frequently. For the LMJ project, frequent status reports are made to update the evolution of the system alignment performance.

In order to make these reports, we need to follow up the contributors' status during their lifecycle and to report corresponding milestone values. The more advanced the milestone is, the more accurate the assessment of the final value will be. The DOORS requirement database built for the LMJ project allows data input for each contributor and its use (especially the latest value) in the ascendant calculation of the system performances.

Due to different contributors using a variety of state of the art techniques, every supplier has its own schedule and some products are in the feasibility stage when others are nearly qualified. Then, the question arises of how to describe in these reports the maturity of the performance (its confidence level)?

For the alignment status reports at the LMJ, specific states of advancement were considered:

- The specification value: what you need,
- The contract value: what you may get,
- The model/tests value: what you are likely to have,
- The qualified value: what you have.

The percentage of contributors at each different state, used for the calculation of the system performance in the status report, gives important information about the confidence level.

6 Conclusion

Requirement management has been actively employed to develop a range of LMJ project specific tools to ensure integration success. The entire requirement breakdown structure, coupled with the ascendant calculation, is a major tool for change request analysis, for integration process support and for performance validation.

Requirement management needs a real calculation part and it needs to register the complete breakdown structure and milestone values. Whilst the choice was made for the LMJ project to use DOORS together with specific code developed for calculations, one should choose his own software solution based on his own priorities.

Moreover, the whole requirement management methodology can be extended to a ‘‘design to cost’’ activity: there is indeed a strong link between these two methods as their shared concept lies in linking the end user’s system needs with the equipment manufactured. So the requirement breakdown structure can be used for design to cost interpretation.

References

[1] Blanchet, C., Benso, E., Chiocchio, S.: Experience from Laser Mega Joule applied to the ITER System Engineering and Configuration Management. Dissertation. SOFT 2009, ‘ITER from design to procurement’ session (2009)

[2] Blanchet, C., Fleurot, N., Montasheri, C.: Performance engineering methods and tools on the Laser Mégajoule Project. Dissertation, AAAF CS2E (2005)

[3] Luttman, M., Denis, V., Lanternier, C., Péalat, M., Compain, E.: Laser Mégajoule alignment to target chamber center. In: Proc. SPIE, vol. 7916, p. 79160N (2011)

[4] Pascal, C., Blanchet, C., Ollivier, J.M.: The Value of Modern Decision-Making Support Service to Fusion Projects. Dissertation, SOFT 2006 (2006)

[5] General recommendation for the program management specifications, RG AERO 00040 BNAE (1999)

Chapter 9
Modelling Languages for Functional Analysis Put to the Test of Real Life

Jean-Luc Voirin

Abstract. ARCADIA is a system & software architecture engineering method, based on architecture-centric and model-driven engineering activities. It targets systems whose architecture is largely constrained by issues such as performance, safety, security. This paper focuses on functional analysis concerns in ARCADIA, and return of real life experiments around use of existing standards such as Architecture frameworks, SysML/UML, for this purpose.

1 Introduction

System Engineering of aerospace, transportation or security electronic devices and systems (e.g. avionics, flight or aircraft or train systems control, mission computers,…) is submitted to high constraints regarding safety, security, performance, environment, human factors and more; all of these are under responsibility of different stakeholders, yet deeply influence systems architecture design and development; thus they are to be reconciled in a relevant system architecture.

The approach partly described in this paper, named ARCADIA (ARChitecture Analysis & Design Integrated Approach) ([1][2]), based on ***architecture-centric and model-driven engineering*** activities, aims at:

- securing system definition and verification,
- supporting collaborative work on architecture,
- and easing fluid technical relationships with both customers and suppliers.

ARCADIA definition was completed in 2008, and the method is in operational use since then. Initially, it was supported by use of “standard” languages such as architecture Frameworks (e.g. NATO AF or NAF [4]), and SysML/UML ([5][6]). But

Jean-Luc Voirin
THALES
Centre Amiral Nomy - 10, avenue de la 1ère DFL - CS93801
Brest, Cedex 3 29238
France
e-mail: jean-luc.voirin@fr.thalesgroup.com

first return of experiments coming from early operational users in real life, dealing with large scale models and complex systems/software, raised significant difficulties with these languages given ARCADIA purpose; this resulted into modifications in the method support language (meta-model), especially in the field of functional analysis definition and use. This is the subject of this paper.

2 Why a Functional Analysis?

Current practices in system engineering are often based on « requirement to box allocation »:

- customer need is captured through (usually textual) requirements;
- these requirements are allocated to components of a system breakdown whose outline and justification are rarely formalized and capitalized,
- then interfaces are built and "justified" by means of (manual) traceability with requirements,
- IVVQ defines test campaigns based on a set of requirements, whose precise allocation on components to be integrated often appears fragile (because manual, here again)…

These practices show their limits, especially in large and complex systems/software engineering, and when reactivity, agility are required. In contrast to them, ARCADIA promotes driving engineering, no more by requirements only, but mainly by **functional need analysis** (and operational analysis as well). Of course, non formalized requirements are still used, but not used as a major driver for engineering.

Comprehensive need requirements are thus captured that way, by "translating" each textual requirement into functions, exchanges and data flows between functions, conveyed data, modes & states, time-related scenarios…; this leads to a **formalized, checkable need model description**, including expected behaviour, exchanged data, etc.

Quantified and non functional requirements such as latency, reaction time, feared events, security threats, cost target, and more, can be expressed on this model by defining non-functional properties on model elements. Traceability with textual input requirements is of course maintained for each functional element contributing to a requirement.

User explicit requirements are often not sufficient in order to describe end users expectations, the context of their work, conditions and constraints… Thus, before or in parallel with the functional analysis, an operational analysis is performed to capture user missions, activities, organization, constraints, tasks and way of working… Usually, some elements of the functional analysis should contribute to each element of this operational need analysis in order to fulfil operational need (similar to Architecture Frameworks). Confronting textual requirements to operational analysis and functional/non functional analysis usually greatly helps in securing and strengthening need definition (figure 1).

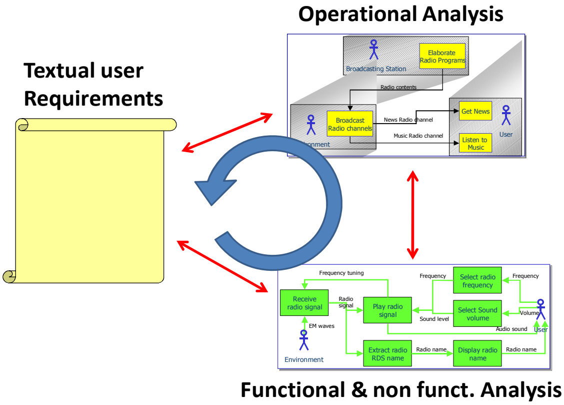


Fig. 1 Securing need definition by confrontation

Once this need analysis is done, architecture is built by grouping or segregating these functions into components, under engineering constraints such as performance, interface simplification, safety, security, cost, reuse… this results in an architecture still related to - and “documented” by - the need description, and that can be justified with regards to its expectations.

Many uses can be made of this model, thanks to multi-viewpoint analysis ([3]); among others, some uses are:

- Defining and justifying components interfaces based on functional dataflow between functions implemented by each component
- Describing and validating dynamic interactions of components based on time-ordered functional sequence diagrams
- And also:
    - Mastering safety and security analyses,
    - planning and mastering Integration Verification Validation (IVV),
    - optimizing non regression testing,
    - estimating cost and risks,
    - securing sub-contracting,
    - securing evolutions,
    - optimizing product line definition, management, and value analysis,

… and much more. Multi-viewpoint analysis is unfortunately out of the scope of this paper, and will not be detailed here.

3 How to Elaborate Functional Analysis?

As mentioned previously, functional analysis is used by ARCADIA as *the way to express and formalize customer need and expectations – and detailed behaviour* *expectations as well* (in architecture definition steps); before all, this formalization is expected to allow an early check of requirements, in terms of completeness, consistency, coherency, what is hard to obtain when considering textual requirements only.

This approach is different from just defining a requirement specific language, using ontology and appropriate syntax to guide and constrain expression of requirements. It is more an *added value transformation* into the description of the expected behaviour and properties of the final product based on - and justified by - the initial requirements.

The process to elaborate functional analysis depends on the context, and when defining ARCADIA, a special effort has been put on its ability to deal with non conventional waterfall or V cycle: ARCADIA is compatible (and tested) with top-down, bottom-up, middle-out, iterative, incremental, agile and mixed approaches. But for sake of simplification, several major ways can be (simplistically) described here, depending on the available sources of requirements:

- When an operational need analysis exists (such as those promoted by TOGAF, DODAF, NAF Architecture Frameworks), then for actors dealing with system, each owned operational activity shall be translated into some functions and exchanges between them, these functions being then allocated to system and users/actors (under performance and workload constraints, a major human factors concern); traceability is maintained with operational analysis for justification;
- When operational scenarios between users, or interaction scenarios between users and system can be identified, then a "use case based" approach can also be applied, starting from operational scenarios, and identifying functions contributing to scenarios interactions;
- When user requirements exist, then each of them shall be "translated" in terms of functions (e.g. expressing the "verbs" of the requirement), exchanges between these functions (expressing dataflow and information exchanges), and non functional properties on them (e.g. performance expectations, security & safety constraints, human factors…)
  this is a "bottom-up" process, creating low level, fine grain, leaf functions, that are later grouped into coarse grain mother functions;
- When a product is to be defined from scratch or without sufficiently detailed requirements, then, added to operational analysis, a "top-down" process can be used, decomposing the product in terms of major expected capabilities, and defining functions fulfilling each capability, in a recursive, hierarchical manner.

Of course, these different ways can be mixed to adapt to each context, *and tooling should be able to support this.*

Note that during this process, each function is progressively defined, by its capability to deliver services, information, data, flows…, and its need for inputs of the same kind. This is an important feature, as explained below.

4 What Kind of Functional Analysis?

When functional analysis becomes complex (several hundreds or even thousands of functions in complex systems), a function usually appears in several contexts and diagrams; it can be reused several times in this single analysis, in order to simplify and secure definition; thus care has to be taken to ensure consistency between its different uses.

The first functional analyses that we experimented in the large, used architecture frameworks support tools; each function was therefore only defined by name, and exchanges with its neighbourhood - in the context of a given diagram. Inconsistencies in the understanding of a function inputs/outputs where often detected very late in the modelling phase: each functional analyst modified function use by changing/adding exchanges according to his/her needs, with no means to be warned against resulting inconsistency with other uses of the function.

We concluded from this that *function inputs/outputs have to be formalized independently from its specific use in one given context*; thus the notion of "function port" characterizing these inputs/outputs. Thanks to this, each time the function is considered for reuse in the elaboration process above, the compatibility of its inputs/outputs with the reuse context is checked, resulting in better definition and consistency of functions and functional exchanges. Ports of a function describe a kind of "direction for use" of the function.

Since we introduced function ports, we decided to follow SysML standard by considering port delegation for nested functions (in the same way as for blocks or activities/actions in SysML). Functions where intended to be hierarchically defined; due to SysML rules, only top level functions were allowed to be connected to each other by functional exchanges; delegation links between mother ports and children ports ensured precise exchange allocation.

Unfortunately, when applied to large models by early users, this led to a huge work each time a modification had to be done: moving a function from one mother to another, changing an exchange source or destination… meant dealing with several unnecessary delegation links, for example. Even in diagram management, focusing on leaf function interaction was complex and difficult to tool or to allow.

Furthermore, in the bottom-up approach above ("translating requirements"), leave functions are defined first, and then they are grouped into more abstract mother functions for ease of understanding (abstraction).Transforming a simple exchange between two leave functions into a set of delegation links and exchange between top level functions appeared tedious and inefficient, hard to modify and not relevant. Tooling could have been a (complex) solution, but with little added value, especially because considering functions, hierarchy deals more with organization or abstraction than with structuring: our users often change top-level functions meaning and outline when moving from need analysis to architecture (see later in this paper).

Figure 2 compares two ways of representing interactions between nested functions, in order to illustrate consequences on diagram complexity:

- the upper one (in green) is extracted from Thales modelling tool supporting ARCADIA;
- the lower one (light yellow) comes from an open source SysML modeller; in order to represent a similar situation in SysML, you may need up to four diagrams (yet this may depend on the kind of modeller, of course).

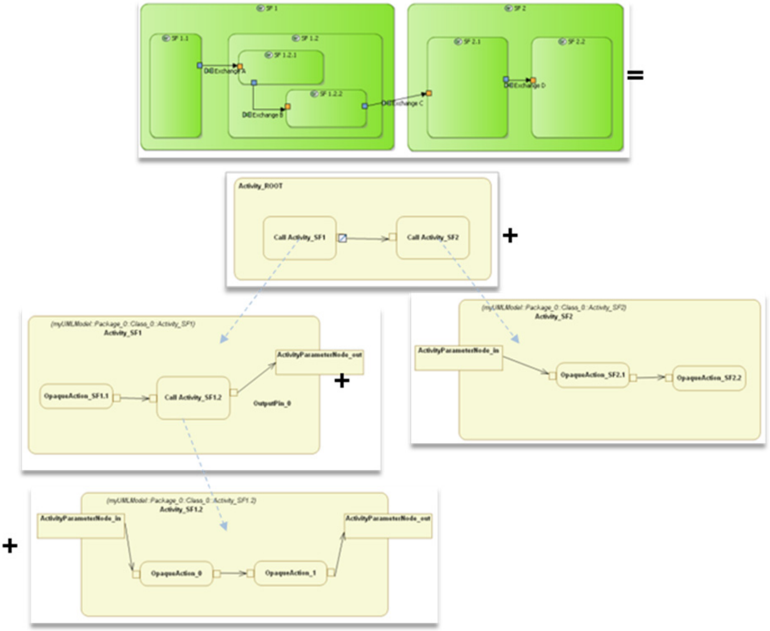


Fig. 2 Ports delegation in SysML can make simple representations hard to obtain

A similar problem arose when allocating functions to components: when dealing with direct exchanges between leaf functions, defining components exchanges and interfaces from them is straightforward, as well as traceability; when dealing with delegation links, it is much more difficult, especially when a mother function has children allocated to several components. And here again, any change in model is complex, as well as model exploitation/analysis.

For all these reasons, we had to change our mind and relax some SysML-originated constraints:

Any function can be linked to any other, no matter whether they are top-level or nested, child functions; no delegation links are defined between mother and child functions. This leads to mainly defining links (functional exchanges) between leaf functions.

In a top-down approach, once a function has been broken down into child functions, *functional exchanges of the mother are just moved towards the relevant child function* in charge of managing the exchange. In bottom-up approach, mother functions can be defined after leaf functions, to group them, and there is no need for extra allocation or delegation work.

Of course, this is not sufficient to manage abstraction: when defining a model diagram with only mother functions, users would like to see which functional exchanges it deals with (even if allocated to its sub-functions). This is automatically done by tooling (functional exchanges allocated to child functions are displayed as if allocated to the mother function, if it is the only one visible in the diagram), but this is only done at diagram level ("computed links"), instead of in the model itself, resulting in a much easier modelling and model management and use.

Another side effect of the top-down approach is that once refinement of functions and exchanges are done, high level diagrams using mother functions appear cluttered with very numerous exchanges; in order to solve this point and give the same abstraction capability to exchanges as compared to functions, we introduced the notion of "exchange categories"; *when several exchanges between two functions belong to the same category, they can be displayed as a single "composite" exchange.*

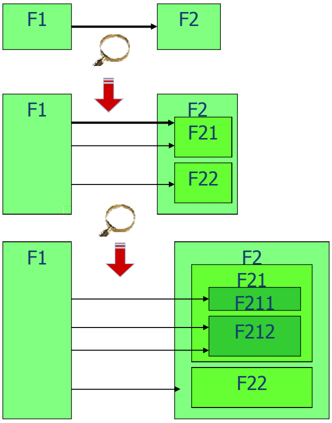


Fig. 3 Categories: abstraction means for exchanges

This emphasizes the fact that *functional breakdown is not structural, but rather documentary*, or in other words just a way of *dealing with abstractions* to hide complexity at some level of detail.

It is to be noted, however, that this is just a modelling option: ARCADIA functional approach can be supported in SysML, by reconstructing delegation links and moving exchanges to mother functions, e.g. if required in a SysML model or for model interoperability.

5 Data Flows or Activity Diagrams?

How to specify the expected interactions between functions? The first need is to specify data flows (or information, or signals, or fluid flows, energy, heat…) to be exchanged between these functions; these "functional exchanges" are related to ports describing function capabilities, as sources or destinations (they may be associated to a data model describing the exchanged contents in more details).

This just describes which inputs a function needs to execute, and which outputs it can deliver (thus the necessity of function ports, as mentioned above). But it also describes **functional dependencies** between functions, which are of prime importance in architecture engineering: these dependencies drive architecture versus need analysis, for most engineering concerns (to be described in a later paper): Definition and justification of interfaces, Performance, Safety, testability, availability, Security, Constraints for integration verification validation, Constraints on startup and reconfiguration procedures…

This is why the major functional interaction modelling means used in Arcadia only deals with functional exchanges as defined here (while excluding pure control flows). To summarize quickly, what we call “dataflow” describes:

- Functions definition in terms of “direction for use” through function ports, and function contents through sub-functions (children),
- Dependencies between functions, as expressed by (oriented) functional exchanges linking their ports,
- Nature of data, information, signals, and flows… exchanged between functions.

Considering SysML or UML activity diagrams to support these concepts, the OMG specifies:

“Activity modelling emphasizes the sequence and conditions for coordinating lower-level behaviours, rather than which classifiers own those behaviours. These are commonly called control flow and object flow models. The actions coordinated by activity models can be initiated because other actions finish executing, because objects and data become available, or because events occur external to the flow.” ([6] page 303)

The mix of control and data (or object) flows in UML/SysML activity diagrams, although well adapted to describe algorithms, led to some difficulties for our system engineering purpose:

Even though data-oriented flows can be used between activities in SysML activity diagrams, the way to secure and control “direction for use” of these activities (e.g. by ports) is not easy: control flows do not contribute to the “direction for use” of an activity, and do not express an intrinsic required dependency between functions that are executed as a sequence. Therefore, control flows should be at least separated from data flows, and should not use same ports as a means to describe direction for use.

This raises the question of how to express control flows. Many of them are or may be related to “underlying” data flows: e.g. events, commands, service requests… In these cases, corresponding data flows can be qualified as carrying control. Note that *this covers a very large part of needs if we restrict to expressing expected behaviour* (and not final [software] design): when describing what functional behaviour is to be done, you don’t necessarily need to explicitly define which function should be executed before which other, but just which precedence constraints should be respected – this is adequately described by functional data flows dependencies.

In some cases, however, describing an order between existing data flows can be useful, either to describe some time-related operational scenarios, or to specify non functional constraints such as latency, safety criticality…

In ARCADIA, time-related operational scenarios are described by using sequence diagrams, whose arcs are functional exchanges described earlier. This allows *describing the possible "functional graph" by means of the data flow, and different ways of running along it by means of different scenarios.*

Currently in the method and its applications, these means are most of the time sufficient. In some rare cases, triggering functions are defined; order of sequence itself is described by means of a sequence diagram or functional chain. Note that this preserves the properties described above in terms of direction of use, because it only adds a trigger entry to the function, which is not thus linked to others for execution order concerns.

However, of course, the opportunity to add specific sequence arcs is still under study, but if so they would be fully distinguished from functional exchanges, and not part of the direction for use definition (function ports).

Furthermore, pure control flows raise some problems still to be solved when allocating functions to architecture components: what would be the meaning of two function s linked by such a link, if allocated on two separate processors, without any supporting event (or command) exchange?

6 One or Several Functional Analyses?

As seen above in this paper, ARCADIA recommends capturing need through a functional and non functional need analysis (linked to requirements and operational analysis from which it is built). This is done in System need Analysis (SA). This part of the model has a lifecycle related to need definition only. It is sometimes called "external functional analysis" ([7][8]).

When entering design, the approach is to allocate functions on components, by grouping or segregating them according to architectural constraints.

Here, the question that we had to address was: can we only use functions coming from need capture, in order to describe each component contents (expected behaviour)? Or should we need to enrich, detail, group, and reuse... some functions, as a design decision?

Most of our users replies meet the same orientation: when defining the architecture, they need to make design and optimization decisions, that lead to modify the former functional analysis; thus they define an enriched one, sometimes called "internal functional analysis": starting from the SA external functional analysis, it details it, adding design choices, reuse concerns, etc. This part of the model has a lifecycle related to design & development rather than only need.

This is done once at Logical Architecture (LA) level, and is likely to be done once again in Physical Architecture (PA), where final technological choices may appear (e.g. adding technical daemons, database storage, middleware or network routing functions, depending on technology implementation).

Note that this is just the way to detail the expected behaviour of each component at the end of system design; this level of detail is likely to be greater than need expectations, so not surprising to have two different functional descriptions. To be short : *In ARCADIA the different levels of functional analysis are useful for* *the same reason as the different levels of requirements for one engineering level* (customer requirements, system requirements, allocated to sub-systems...).

Furthermore, the need for functional abstraction (top-level functions) is different from need analysis: architects often group leaf functions under considerations such as Human System interactions Vs processing Vs data management Vs real-time processing…, thus the logical or physical function tree can be significantly different from the need analysis one, which is usually fully user and functional need-oriented.

Of course, each new level of functional analysis is initialized from the former one, and traceability links should be maintained between them, even if one of them evolves (see Figure 4).

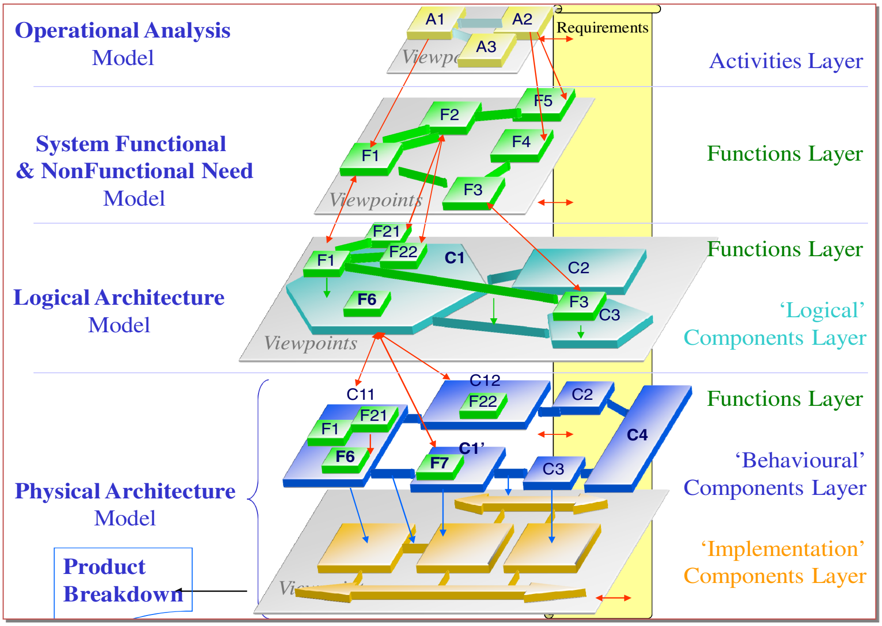


Fig. 4 Major ARCADIA concepts & relationships

This is recommended practice in ARCADIA, and looks beneficial to most pilot projects, that appreciate not mixing need and solution; but of course it can be adapted and tailored.

Here is an example of adjustments:

- In order to ease building and maintaining traceability between functional analyses of different levels, some can choose to only enrich the upper level by *refining* upper level0000.0. functional tree leaves – with limited expressivity in some cases;

- If one considers that separating need and solution is not relevant for them, then they can use the same SA functional analysis at each level - and accepting resulting limits.

7 Conclusion

At the end of this return of experiment, our major convictions were clearly reinforced:

Method is of prime importance: focus first on defining your natural way of thinking; then support it, secure it and share it among stakeholders by means of the method.

A major challenge in adoption of such new practices is to reduce the complexity for engineers to understand new concepts and their representations; for so, Domain Specific Languages tailored to their own context are best suited.

These languages should cover the full scope of required engineering notions, in a seamless and transparent way.

Modelling languages targeting system and software architecture are useful because they give great versatility and broad implementation range, along with a basic common “inspiring” vocabulary; they are of great value also as an interoperability means between organizations and domain specific languages (DSL).

But such languages are today limited to only part of the global engineering need: they cover only parts of the required engineering notions; and – more important – not really founding or structuring engineering enough, to strengthen and secure engineering.

Furthermore, in some cases, these languages still need to be improved to address real life situations such as large scale modelling, model maintenance capacity, functional (or service) based engineering, non functional constraints support… and above all, evolutionary, agile and multi-users team modelling.

And last but not least, a tailored, field proven modelling approach such as ARCADIA definitely appears to be a major lever for transforming our engineering and addressing new challenges and complexity of emerging systems.

References

[1] Voirin, J.-L.: Method and tools to secure and support collaborative architecting of constrained systems. In: ICAS 2010, 27th Congress of the International Council of the Aeronautical Science (2010)

[2] Voirin, J.-L.: Method & Tools for constrained System Architecting. In: INCOSE 2008 Symposium (2008)

[3] ISO/IEC 42010:2007 (also known as IEEE Std 1471–2000) Systems and software engineering - Recommended practice for architectural description of software-intensive systems

[4] NATO C3 System Architecture Framework (NAF), AC/322-D (2004)0041, NATO C3 Board (2004)

[5] Systems Modelling Language (SysML) Specification, OMG document: ad/2006-03-01 (2006)

[6] OMG Unified Modelling LanguageTM (OMG UML), Superstructure – version 2.4 (January 2011)

[7] `http://www.k-inside.com/web/en/produits-et-services/produits/pack-essentials/la-specification-systeme/`

[8] Advances in Integrated Design and Manufacturing in Mechanical Engineering II, Serge Tichkiewitch, p. 316

Chapter 10
Modeling the Impact of Requirements Change in the Design of Complex Systems

João Fernandes, Arlindo Silva, and Elsa Henriques

Abstract. Managing changes in requirements more effectively than others can become a source of competitive advantage for companies designing and developing complex systems. In this paper, we report that decision-makers in these firms feel that understanding the effects of requirements change in the design process is a difficult task in practice. Such knowledge is however crucial to organize the design process in ways that mitigate for the impact of requirement change. Considering this need, we propose here a modeling framework to determine the impact of requirements change in the design process of complex systems. We view the design process as a requirements-driven process containing steps performed in a mechanical way, but ultimately controlled and steered by the human agents involved in it. Therefore, we develop a technique which allows the modeler to capture both the mechanical relationships and the decision-making behavior of design agents. We introduce attributes such as requirements availability, stability, difficulty and margin, which act as process variables driving the design process. Task properties and the agents' behavior are then modeled as functions of these process variables, which are dynamically updated as the design process progresses and decisions are taken. The potential of this modeling technique is illustrated on the design process of a turbine blade cooling system. Discrete-event Monte Carlo simulations are used to assess the impact of requirements change during design.

Keywords: requirements management, requirements change, impact analysis, design process modeling, discrete-event simulation.

1 Introduction

Requirements change has concerned academia and industry during the last couple of decades, due to the importance of requirements in the design and development

João Fernandes · Arlindo Silva · Elsa Henriques
ICEMS, Instituto Superior Técnico, TULisbon, Portugal
e-mail: {joao.ventura.fernandes,arlindo.silva,elsa.h}@ist.utl.pt

of complex systems. Change management processes have been designed to identify, analyze, cost and ensure that changes in requirements are consistently implemented into the system under development [1]. A central activity to all requirements change management processes is change impact assessment [2, 3]. In this paper, we begin by presenting results showing that there are many causes of requirements change during the design of complex systems, but a significant proportion is related to how the design process itself is organized. Finding how can the design process of complex systems be structured to mitigate for requirements change is of interest to decision-makers, but requires understanding *a priori* the impact of changes in design . The main part of this paper is thus the presentation of a modeling framework to assess the impact of requirements change during design. We finalize with an illustrative example based on the design process of a turbine blade cooling system, showing the potential of the proposed modeling technique in requirements change impact assessment.

2 Findings from Exploratory Research

Our starting point on the topic of requirements change was an exploratory research conducted at a large UK aerospace company. This organization develops complex products and is currently employing state-of-the-art systems engineering methodologies [3]. Our belief was that managing changes in requirements during the development of complex systems is a multi-faceted issue and therefore we begun with a qualitative research approach. A set of 14 semi-structured interviews were performed on an heterogeneous sample of engineers and managers, which is shown in Figure 1. The interviews followed the structure *Introduction*, *Warm-up*, *Main Body*, *Cool-off* recommended in literature [4] and lasted between 45 minutes and 1 hour with each interviewee. Sample selection was purposive but following the method of maximum variation [5]. The objective was to gain insight while minimizing sample selection bias through comparison of answers coming from people with very different expertise, experience and job responsibilities in the company. The interview guide consisted of 30 pre-defined *open questions* organized around the requirements process, uncertainty, response to uncertainty and risk, causes and impact of change and current challenges in requirements change management. Despite of the predefined sequence of questions, considerable freedom was given to the interviewees and questions were often suppressed, added or re-sequenced, depending on the path taken by the interviewee. From the analysis and comparison of the interviewee's answers, a set of key findings were retrieved which are subsequently summarized.

2.1 Causes of Requirements Change

The causes of requirements change were found to be diverse. The group generally divided them into *externally* or *internally-driven* changes. Among the first type, it was often mentioned that customer needs may change during the development

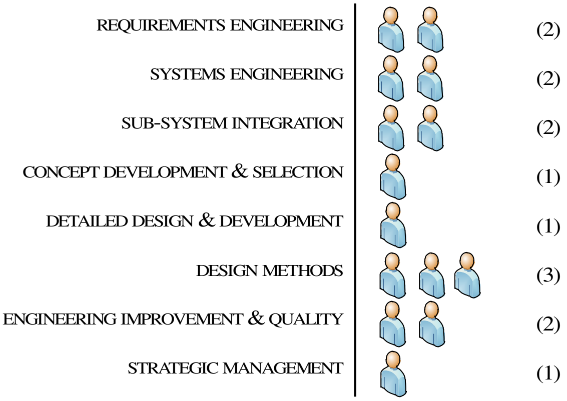


Fig. 1 Overview of the sample of interviewees investigated at a large UK aerospace company

process and regulations evolve cyclically. Both were seen as causes of change. Internally-driven changes included a larger number of causes. Lack of compliance in the initial design solutions was pointed out frequently as a source of change, since the inability of meeting requirements triggers technical trade-offs that often lead to new requirements. Incorrect requirements capture or flow down were two identified causes of change which relate to the requirements management process. On the other hand, lack of communication in design (such as ”unawareness about the requirements status”) and incorrect design planning (such as ”triggering activities too early or too late”) were mentioned causes of change which relate to the organization of the design process itself. Figure 2 lists all the causes of change mentioned by the interviewees according to the number of answers found. It shows that the response to unpredicted system behaviors, lack of confidence in requirements validation or verification activities and change propagation effects were additional causes identified in the interviews, although with lower frequencies of response. There was a general agreement between the interviewees that most changes in requirements are *self-generated*, which is an important finding also suggested by the proportion of answers in Figure 2 that can be related to problems in the design solution, in requirements management or in design management.

2.2 Impact of Requirements Change

Because ”development time is shortening” as one of the interviewees stated, requirements change impact was found to be of great importance. The impact of changes was thought to be considerable in terms of the amount of scrap and rework generated during the design process and, at times, even ”massive”. Having established the context, we probed for which changes ”hurt the most” to establish which types of requirements change cause the highest impact. We found that interviewees associated a high change impact to the following cases or events:

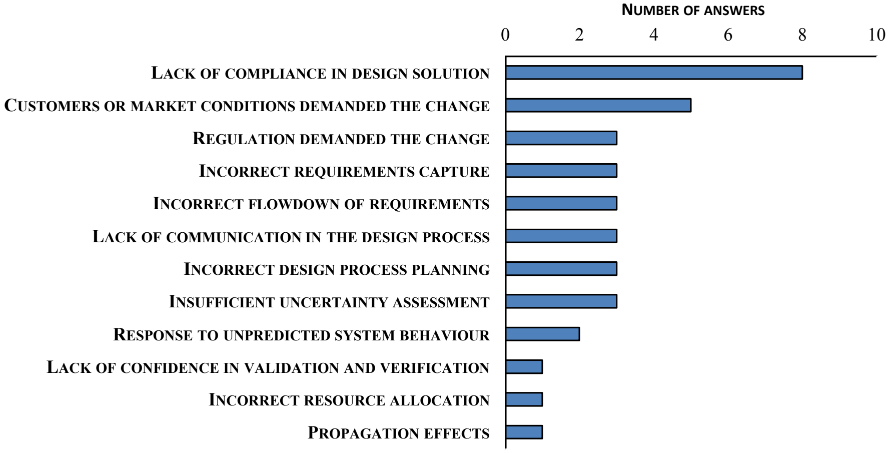


Fig. 2 Causes of requirements change found in the interviews, according to the number of answers

- *Changes in high level product requirements*, which typically trigger design changes in a larger set of sub-systems and thus originate large amounts of scrap and rework.
- *Changes arising during functional sub-system integration*, which typically include changes in difficult to predict interface requirements. Although the impact is significant, changes are typically contained in one or in a small number of sub-systems.
- *Changes arising from product systems requirements*, which include changes in the requirements of systems in the product interfacing with *many* sub-systems. Since the large number of interdependencies can cause networks of design changes, the impact of this type of change was considered high.
- *Late changes*, which typically originates high impact since earlier design activities need to be revisited and trade-offs are harder to achieve because the design is much more constrained in later development stages.

2.3 Conclusions from Exploratory Research

Requirements change is a complex phenomena, as illustrated by the wide range of causes mentioned in Figure 2. Finding an universal approach capable of mitigating all causes of change seems to be an unsurmountable challenge. However, we found in the interviews a connection between requirements change and design process management. The design of complex systems requires design activities to start, usually concurrently, long before requirements are final. Therefore, design teams are forced to work with *uncertain* requirements. This uncertainty is gradually reduced as the design progresses and new knowledge is gained but changes are ”inevitable”, as one of the interviewees stated. Moreover, although interviewees mentioned that change is expected, we concluded that organizing the design process in ways that mitigate for the impact of change is difficult in practice. How can decision-makers structure the design process of complex systems to mitigate for the impact of changes in requirements? This industrial need is our research motivation and supports the development of modeling techniques suited to quantify its impact in the design process.

3 Modeling the Impact of Requirements Change during Design

Modeling always starts from the analysis of the main mechanisms governing a reality and the same principle applies to design modeling. Literature on this subject [6] shows there are two main trends among design process modelers.

One trend sees design essentially as a mechanical process, where the process's outcome is determined by a well defined sequence of interactions between tasks containing individual sources of uncertainty, such as task durations. This has been called the *mechanistic* vision of design [6]. Models based on precedence relationships, such as PERT/GERT [7, 8], signal flow graphs [9] or the Applied Signposting approach [10] are highly efficient representing and simulating well defined sequences of tasks. The same strength is found on models based on dependency relationships, such as the activity-based Design Structure Matrix [11] and its subsequent extensions, which possess the extra benefit of allowing decision-makers to understand which tasks should be made in series or in parallel and which are highly coupled [12].

Conversely, the second trend sees design as a dynamical and adaptive process essentially controlled by the human agents intervening in it, who take *in-situ* decisions [6] about the next tasks to be performed depending upon past events or future predictions. Therefore, design agents actively shape the design process itself and determine its outcome. Adaptive models such Signposting [13] or the Adaptive Test Process [14] rely on adaptive task selection schemes, based for instance on the level of confidence in parameters, while agent-based approaches [15] rely on custom-made computer codes describing the design processes and the behavior of human intervenients.

We argue that *real* design processes contain, with variable proportion, a mixture of both mechanisms. This proportion certainly depends on multiple factors, such as the type of system or the degree of innovation contained in the design. We can foresee that the design process of an entirely new system is quite dynamical, while a redesign process is much more mechanical. It thus varies from case to case. Nevertheless, our belief is that any model capable of assessing the impact of requirements change must contain both of the previous modeling capabilities.

3.1 A Requirements-Driven Process Model

In addition, we view the design of complex systems as a *requirements-driven process*, due to the importance of the *status* of requirements during design. Deciding when to start a set of design activities or to what level of accuracy they should be performed certainly depends upon the availability and the maturity of the corresponding requirements. The technical difficulty associated with the target values embedded in requirements certainly affects the time spent in design activities, searching for better concepts, technologies, materials and validated results. Additionally, margins introduced in requirements influence decisions from design agents, such as whether to continue iterating or to perform trade-offs. There are thus numerous examples supporting the idea that the design of complex systems is *requirements-driven*.

To capture such mechanisms in a model, we propose that a set of *attributes*, characterizing the *status* of requirements, drive the design process during time. The concept of requirement attributes has been suggested by several authors in requirements engineering literature [1, 2]. Volatility, for instance, is commonly used in requirements management in software development [2]. However, the concept's potential has not been fully explored in design modeling. In this paper, we propose (but we are not limited to) four requirement attributes to model design:

1. **Availability** - is seen as an attribute conditioning the existence of dependent entities during design, such as activities or other requirements. For instance, an unavailable requirement status can prevent dependent activities to begin.
2. **Stability** - is seen as an attribute shaping the probability that the requirement will change during a particular design stage and conditioning task properties. We view a stable requirement status as a condition determining a small probability of change (e.g. 2%) while an unstable status is a condition leading to a large probability (e.g. 50%). In addition, an unstable status also determines lower task durations since teams typically reduce effort and accuracy when input data is more uncertain.
3. **Difficulty** - is perceived as an attribute shaping the probability of the design solution complying with a requirement, as a result of a particular sequence of activities and considering the current technologies employed in that solution. A difficult status is a condition determining a large probability of non-compliance, for instance.
4. **Margin** - is realized as the tolerance of the requirement to non-compliance, acting as a threshold that conditions the outcome of non-compliance situations. A low margin, for instance, is seen as a status making the process very vulnerable to non-compliance and thus increasing the probability of reworking certain activities.

3.2 Assessing the Impact of Change through Simulation

Requirement attributes are used to drive the design process over time through simulation. Furthermore, we transform these attributes into *process variables* in our

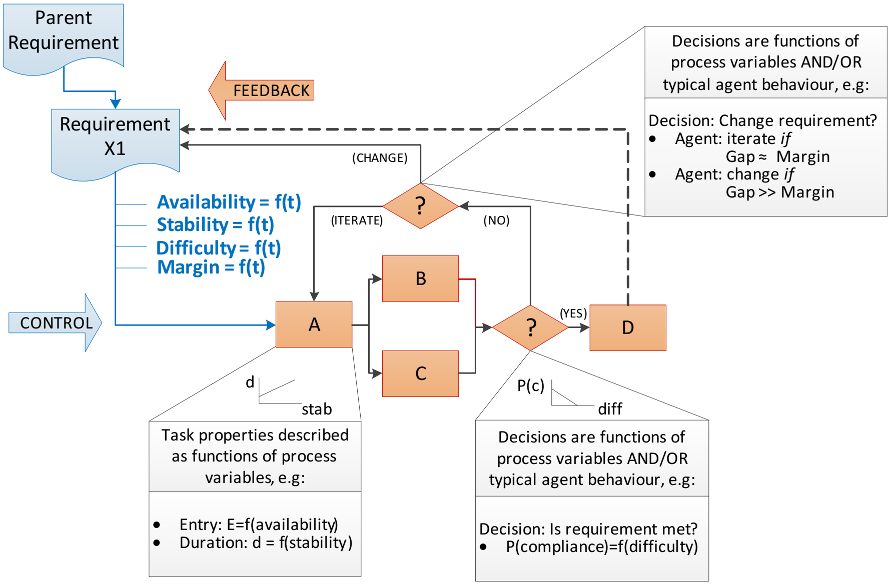

Parent Requirement
FEEDBACK
Requirement X1
Decisions are functions of process variables AND/OR typical agent behaviour, e.g:
Decision: Change requirement?
• Agent: iterate *if* Gap ≈ Margin
• Agent: change *if* Gap >> Margin
(CHANGE)
Availability = f(t)
Stability = f(t)
Difficulty = f(t)
Margin = f(t)
(ITERATE)
?
(NO)
B
CONTROL
A
?
(YES)
D
C
d
stab
P(c)
diff
Task properties described as functions of process variables, e.g:
• Entry: E=f(availability)
• Duration: d = f(stability)
Decisions are functions of process variables AND/OR typical agent behaviour, e.g:
Decision: Is requirement met?
• P(compliance)=f(difficulty)

Fig. 3 Overview and main features of the requirement-driven process modeling framework

modeling framework. Figure 3 illustrates the definition of requirements availability, stability, difficulty and margin as functions of design process time. This transformation augments the model's capabilities during simulation. In fact, defining attributes as process variables allows requirements to *control* the design process and, simultaneously, be affected by it since results from activities or decisions made during the course of the process are *fed back* to them. This intends to capture the *complex dynamics* of design processes and it is illustrated in Figure 3 by the arrows representing control and feedback loops. Figure 3 also depicts, in greater detail, the main features of the requirements-driven process modeling framework, which include:

1. *Requirements traceability data*, i.e, the hierarchy of requirements and their interdependencies. This information is needed to allow changes in requirements to be propagated to connected entities, such as activities or other requirements.
2. *Attributes as functions of design process time*, i.e, the status of requirements is dynamically updated as design progresses.
3. *Task properties as functions of process variables*, i.e., activities depend upon the status of requirements. For instance, as shown in Figure 3, entering the subprocess A-B-C-D depends upon the availability of requirement X1 and the duration of activity A is a function of the associated requirement's stability.
4. *Decisions as functions of process variables and the typical decision-making behavior of agents*, i.e., the process's evolution is influenced dynamically by the requirements status and the encoded behavior of design agents. A couple of examples are depicted in Figure 3. For example, the probability of compliance is function of X1's difficulty. In addition, the agent's typical behavior in the event

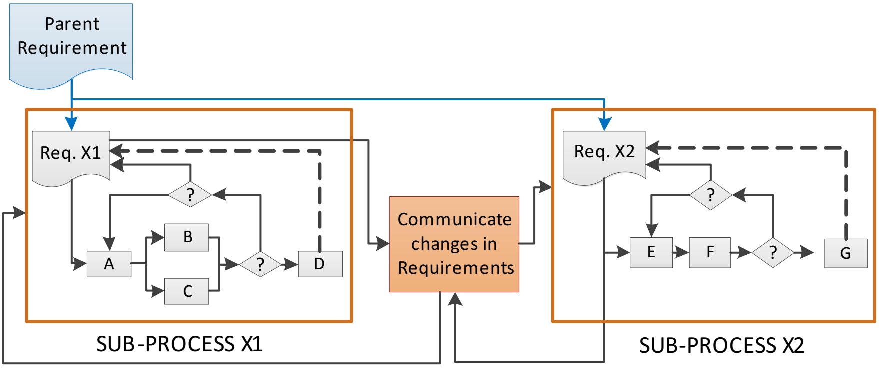


Fig. 4 Scalability of the proposed requirements-driven process modeling framework

of non-compliance is captured through the relationship between the compliance gap (the difference between the current and target values) and requirement X1's margin status, which dictates whether to change the requirement or to iterate again.

We illustrate in Figure 4 that the proposed modeling framework is completely scalable and it is up to the modeler to decide the appropriate level of detail for the impact analysis. However, according to the findings presented in subsection 2.2, we can anticipate that changes in requirements and constraints affecting the design of related sub-systems or components will be of interest to most models. This rich representation can then be used to assess the impact of requirements change during the design of complex systems through discrete-event Monte Carlo simulations [16]. The result of these requirement-driven simulations is thus a *probabilistic assessment* of impact. This paper follows with an illustrative example. We argue that, despite its probabilistic nature, such impact assessment is useful to decision-makers and can subsequently lead to design process improvements that mitigate for the effects of requirements change.

4 An Illustrative Example

The purpose of this illustrative example is to demonstrate the potential of the proposed framework. A simplified version of a turbine blade cooling system design process was retrieved from literature [17] to perform this demonstration, since it integrates the design of a complex system: the jet engine. Figure 5 depicts this design process. In addition, we propose in Table 1 requirement-based relationships to model properties and agent behaviors in this illustrative example. We will naturally need to validate any relationships in a case-study with a real design team later on. But for demonstration purposes, we claim that the relationships shown in Table 1 are straightforward models for task durations, probabilities of compliance and

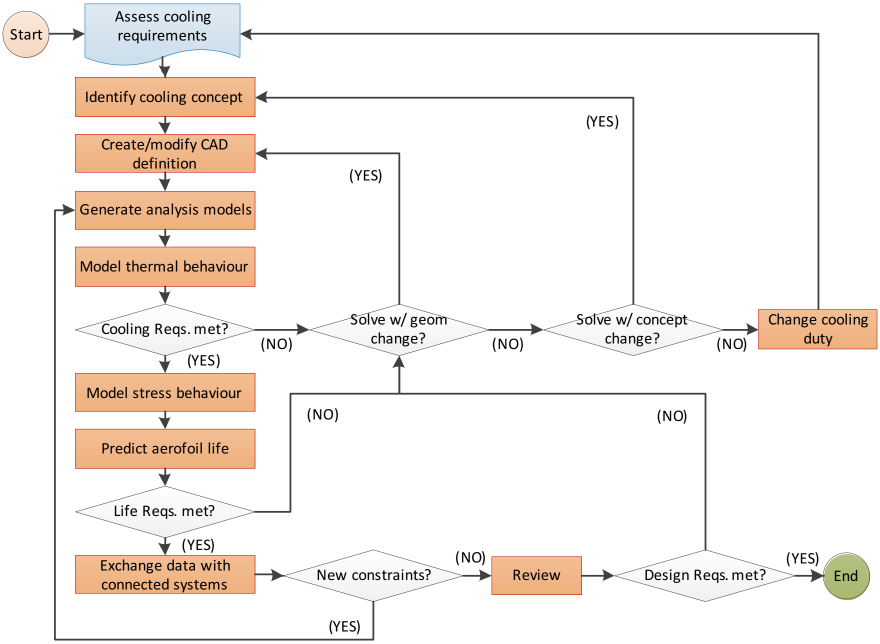


Fig. 5 The turbine blade cooling system design process used in this example, simplified from [17]

requirements change, compliance gap and the decision-making behavior of agents when non-compliance occurs. These are dependent variables in our requirements-driven model. The attributes *Avail, Stab, Diff, Mar* are the independent variables driving the model, taking numeric values between 0 and 1. In this illustrative case, the actual numbers were sampled from *assumed* triangular probability distributions. It also means that an unstable requirement would get $Stab = 0$ in this example, while a stable would take $Stab = 1$. Similar reasoning applies to the other attributes. T_i^{max}, T_i^{min} and e_i are data inputs. We consider that tasks have minimum and maximum durations and different *effectiveness*, e_i, which represents the capability of making the design progress, i.e., an *improvement rate* towards compliance. The previous effectiveness concept is similar to the one used in ATP [14]. We have assumed minimum and maximum task durations ranging from 4 hours to 3 days. Relatively to the effectiveness values, we assumed that concept and geometry generation are the main tasks in this process capable of making the design solution progress and these were assigned 10% and 20% improvement rates, respectively. This means that each time such tasks were repeated, the compliance gap, g, decreased according to the task's effectiveness. K_a was set to 1. The relationships proposed in Table 1 have then been assigned to this particular design process. This assignment is shown in Table 2. The model for this example has been constructed in Applied Signposting [10], but advanced DSM models could also be employed [18, 19].

Table 1 Definition of the relationships used to model properties and behaviors in this example

Modeled properties/behavior	Definition
The duration of a task, d_i, is proportional to the stability, $Stab$, of the requirements driving the task and decreases with increasing number of iterations of that task, N_i. T_i^{max} and T_i^{min} represent maximum and minimum task durations and bound the expression.	$d_i = \frac{[Stab \times (T_i^{max} - T_i^{min})]}{N_i} + T_i^{min}$ (1)
Following a sequence of tasks, the probability of the design solution complying with the requirements decreases with increasing requirements difficulty, $Diff$, and increases with increasing number of iterations of that sequence, N_s.	$P(compliance) = 1 - Diff/N_s$ (2)
During the course of the design process, the probability of changes in requirements decreases with increasing requirements stability, $Stab$.	$P(change) = 1 - Stab$ (3)
The gap between solution and requirement, g, is related to the requirements difficulty, $Diff$, and inversely related to the effectiveness, e_i, and number of iterations, N_s, of the m tasks affecting the solution's progress during design.	$g = Diff/(1 + \sum_{i=1}^{m} e_i)^{N_s}$ (4)
If the difference between the compliance gap, g, and the requirement margin, Mar, is close to the effectiveness of a particular task, e_i, the agent decides to continue iterating that task. K_a is a constant adjusted according to the agent's typical tolerance.	$\lvert g - Mar \rvert \leq K_a \times e_i$ (5)

Attributes $Avail$, $Stab$, $Diff$ and Mar are independent variables of each requirement, taking numeric values between 0 and 1. T_i^{max}, T_i^{min} and e_i are data inputs retrieved from the actual process. K_a is set according to the agent's tolerance. d_i, $P(compliance)$, $P(change)$ and g are dependent variables. All other variables are internal.

With assumed inputs and the previous modeling framework, the turbine blade cooling system design process was simulated using discrete-event Monte Carlo simulations. The probability distribution for the overall process duration, which represents a basic performance measure of the design process, is shown in Figure 6. The impact of requirements change in the design process is then depicted in Figure 7, which shows the probability distribution for the *duration of rework triggered by requirements change*. Figure 7 shows that, in this illustrative example, the cumulative probability of a rework duration between 2 and 5 weeks due to requirements change is 61%. These results characterize the performance of the design process, in its *current state*, relatively to requirements change. In addition, we aim that the proposed framework becomes an useful tool to investigate the effects of change mitigation

Table 2 Assignment of modeled properties/behaviors to the design process shown in Figure 5

Activity	Pre-process	Properties/Behavior	Post-process
Assess cooling reqs.		$d = constant$	Avail, Stab, Diff, Mar
Identify concept		Eq. (1)	
Create/modify CAD		Eq. (1)	
Generate models		Eq. (1)	
Model thermal behavior		Eq. (1)	
Cooling reqs. met?		Eq. (2)	*if* No, then Eq. (4)
Solve w/ geom change?		Yes *if* Eq. (5) is true	
Solve w/ concept change?		Yes *if* Eq. (5) is true	
Change cooling duty		$d = constant$	
Model stress behavior		Eq. (1)	
Predict aerofoil life		Eq. (1)	
Life reqs. met?	Avail, Stab, Diff, Mar	Eq. (2)	*if* No, then Eq. (4)
Exch. data w/ conn. sys.		$d = constant$	
New constraints?	Avail, Stab, Diff, Mar	Eq. (3)	Diff, Cooling Req.
Review		$d = constant$	
Design reqs. met?		Eq. (2)[1]	*if* No, then Eq. (4)

[1] Assuming independence in the probability of compliance of multiple requirements.

strategies and to search for process improvements to the *current state*. Considering the design process used in this example, several mitigation strategies could be investigated with this framework. For instance, the modeler could determine whether to start concept generation earlier or later as a function of the stability and difficulty levels in the requirements cascaded to the turbine blade component. Or to determine if the cooling concepts should be created and assessed in series, as shown in the *current state*, or in parallel, depending on the status of the requirements attributes. Or even to determine if the data exchange task with connected systems should be performed earlier or more frequently to reduce the likelihood of long design iteration loops due to changed constraints. Deriving a *future state* for the design process, which is likely to deliver a lower duration of rework triggered by requirements change and thus lower development costs is the ultimate goal, as illustrated in Figure 8. This illustrative example therefore demonstrates the potential of the proposed framework to assess the impact of requirements change in the design of complex systems.

Fig. 6 Probability distribution for the duration of a turbine blade cooling system design process [17], with assumed data inputs

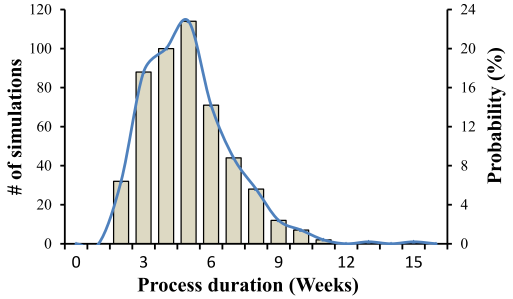


Fig. 7 Probability distribution for the duration of rework triggered by requirements change in the design process simulations presented in Figure 6

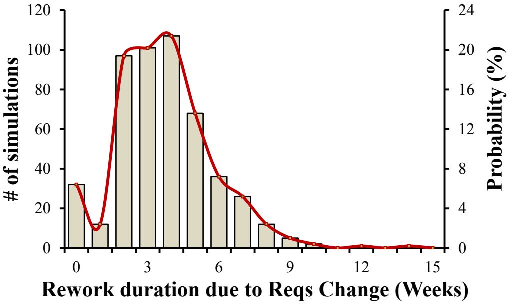
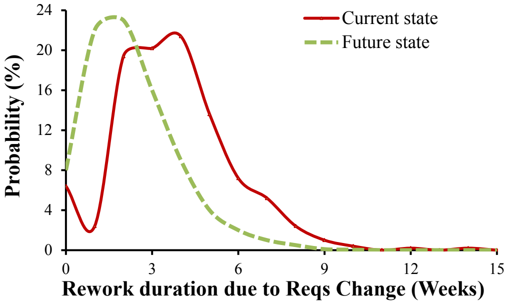


Fig. 8 Example of the envisioned probability distribution for the duration of rework triggered by requirements change, after the study of change mitigation strategies and the implementation of process improvements



5 Conclusions and Outlook

In conclusion, this paper has proposed and described a requirements-driven modeling framework to assess the impact of requirements change during the design of complex systems. We have been inspired by findings obtained at a large UK aerospace company demonstrating there is the need to understand how can the design process be structured to mitigate for requirements change. This paper proposes requirement attributes to model the complex dynamics of design, such as task properties or the decision-making behavior of agents. Within a simulation environment, these relationships allow a probabilistic assessment of requirements change impact. Developing this modeling framework and validating it with design teams is planned through further research at the same aerospace company.

Acknowledgments. Funding from the Portuguese Foundation of Science and Technology under the doctoral grant SFRH/BD/51107/2010 is greatly acknowledged.

References

[1] Kotonya, G., Sommerville, I.: Requirements Engineering: Processes and Techniques, 1st edn. Wiley, Chichester (1998)

[2] Pohl, K.: Requirements Engineering: Fundamentals, Principles and Techniques, 1st edn. Springer, Heidelberg (2010)

[3] NASA. Systems Engineering Handbook, 1st edn. National Aeronautics and Space Agency, Washington D.C. (2007)

[4] Robson, C.: Real World Research - A Resource for Social Scientists and Practitioners, 2nd edn. Blackwell Publishing (2002)

[5] Kuzel, A.: Sampling in Qualitative Inquiry. Sage Publisher (1999)

[6] Wynn, D.C., Eckert, C.M., Clarkson, P.J.: Modelling iteration in engineering design. In: 16th International Conference on Engineering Design, Paris, France (August 2007)

[7] Wiest, J.D., Levy, F.K.: A Management Guide to PERT/CPM, 2nd edn. Prentice Hall (1977)

[8] Pritsker, A.A.B.: Gert: Graphical evaluation and review technique. Technical report, The RAND Corporation, RM-4973-NAS (April 1966)

[9] Eppinger, S.D., Nukala, M.V., Whitney, D.E.: Generalised models of design interation using signal flow graphs. Research in Engineering Design 9(2), 1121–1123 (1997)

[10] Wynn, D.C., Eckert, C.M., Clarkson, P.J.: Applied signposting: a modeling framework to support design process improvement. In: Proceedings of ASME Design Engineering Technical Conferences, Philadelphia, Pennsylvania, USA (September 2006)

[11] Browning, T.R.: Applying the design structure matrix to system decomposition and integration problems: a review and new directions. IEEE Transactions on Engineering Management 48(3), 292–306 (2001)

[12] Eppinger, S.D., Whitney, D.E., Smith, R.P., Gebala, D.A.: A model-based method for organizing tasks in product development. Research in Engineering Design 6(1), 1–13 (1994)

[13] Clarkson, P.J., Hamilton, J.R.: Signposting, a parameter-driven task-based model of the design process. Research in Engineering Design 12(1), 18–38 (2000)

[14] Lévàrdy, V., Browning, T.R.: Adaptive test process designing a project plan that adapts to the state of a project. Technical report, INCOSE (2005)

[15] Olson, J., Cagan, J., Kotovsky, K.: Unlocking organizational potential: A computational platform for investigating structural interdependence in design. Journal of Mechanical Design 131(3), 1–13 (2009)

[16] Leemis, L.M., Park, S.K.: Discrete-Event Simulation: A First Course, 1st edn. Pearson Prentice Hall (2006)

[17] Bell, C.P., Dawes, W.N., Jarrett, J.P., Clarkson, P.J.: Improving the conceptual design of turbine rotor blade cooling systems. In: 15th International Conference on Engineering Design, Melbourne, Australia (August 2005)

[18] Carrascosa, M., Eppinger, S.D., Whitney, D.E.: Using the design structure matrix to estimate product development time. In: Proceedings of the ASME Design Engineering Technical Conferences, Design Automation Conference, Atlanta, Georgia, USA (September 1998)

[19] Cho, S., Eppinger, S.D.: Product development process modeling using advanced simulation. In: Proceedings of ASME Design Engineering Technical Conferences, Pittsburgh, Pennsylvania, USA (September 2001)

Chapter 11
Model-Driven Development of Logistic Systems Using Domain-Specific Tooling

Jacques Verriet, Hsuan Lorraine Liang,
Roelof Hamberg, and Bruno van Wijngaarden

Abstract. The development of complex systems involves many people from different disciplines, each communicating with his own jargon. These different languages may lead to misunderstandings between stakeholders that cause a significant increase of development costs. This paper addresses this communication gap by proposing the usage of domain-specific tooling, which is shared by all stakeholders. We argue that logistic systems are well suited for the usage of such tooling. This is illustrated by the application of domain-specific tooling in the warehousing domain. We present a warehouse-specific graphical configuration tool built on top of a warehouse-specific language and apply it to an industrial automated case picking warehouse. This application shows that the communication gap between specification and implementation can be reduced by introducing parameterised components and behaviours and local optimisation rules with well-defined interfaces.

1 Introduction

There are many stakeholders in the development of complex systems. Since these stakeholders have different backgrounds, they generally use different languages. For instance, a system architect uses a different jargon than the software engineers who have to implement the system's control software. These differences cause

Jacques Verriet · Roelof Hamberg
Embedded Systems Institute, P.O. Box 513, 5600 MB Eindhoven, The Netherlands
e-mail: {jacques.verriet,roelof.hamberg}@esi.nl

Hsuan Lorraine Liang
Sioux Embedded Systems, Esp 405, 5633 AJ Eindhoven, The Netherlands
e-mail: lorraine.liang@sioux.eu

Bruno van Wijngaarden
Vanderlande Industries, Vanderlandelaan 2, 5466 RB Veghel, The Netherlands
e-mail: bruno.van.wijngaarden@vanderlande.com

misunderstandings between stakeholders, for instance due to ambiguity of the language used or incomplete information. These misunderstandings may be very costly, because they result in errors that need to be repaired later in the development process. Especially in the final stages of development, the repair of these engineering errors is very expensive [16].

The number of misinterpretations between stakeholders will greatly decrease if they use a language that is understandable to all stakeholders. This is the main purpose of a domain-specific language (DSL). A DSL is a language specific for the underlying solution domain that is shared by all system development stakeholders. Because of its commonality, a DSL leaves little room for misunderstanding. Therefore it reduces the chance of misinterpretations, the resulting engineering errors, and the corresponding rework. A DSL can simplify system development even further if it is linked to a reference architecture, which defines the general system organisation. A DSL can, for example, be a (formal) description of a product family to which the system under development belongs. In that case, the DSL describes the common components of the reference architecture and how they can be configured using their parameters.

A DSL can be used for more than just reducing communication ambiguity. A formal DSL can be used to develop domain-specific tooling. Using such tooling, a system developer can specify a system by appropriately configuring the system parameters defined by the DSL. Moreover, the domain-specific tooling may allow parts of a specified system to be generated directly from the domain-specific configuration tooling, thereby reducing development efforts even further.

Hvam et al. [6] address the benefits of domain-specific tooling. They present a literature study of successful applications of system configuration in a variety of industries. For four companies, they quantify the following benefits of domain-specific system configuration: reduction of lead times, more frequent on-time delivery, reduced resource consumption, higher quality of specifications, and optimisation of products. Other (qualitative) benefits that they observed include increased sales, reduction of production/engineering costs, formalisation of engineering knowledge, and reduction of item numbers.

1.1 Logistic Systems

Logistic systems are well suited for the usage of domain-specific tooling. This is because such systems are highly modular and their constituting parts are loosely coupled. Logistic systems can be seen as a collection of workstations connected by a transportation system. Each workstation implements one step of the overall logistic process. Logistic systems are organised in such a way that the functions performed by the different workstations are largely independent of each other.

Because a logistic system involves many independent processes, a decentralised controller is well suited for the corresponding process control. In the literature, there are many examples of decentralised controllers for logistic systems. Many of these controllers have a structure that resembles the structure of the underlying physical hardware. Van Brussel et al. [13] have defined the PROSA reference architecture for holonic manufacturing systems. PROSA distinguishes three common roles: order, product, and resource. These roles are implemented by three standardised control components, called *holons*. From these holons, one can construct a hierarchical decentralised controller where each manufacturing workstation is represented by instantiations of the holons. ADACOR [7] is another reference architecture for decentralised manufacturing control systems. ADACOR is very similar to PROSA; the main difference is the addition of a supervisor holon.

Moneva et al. [9] have taken PROSA [13] as an inspiration for the control of warehouses. They define a hierarchical reference architecture that distinguishes three roles enacted by holons: order, logic, and resource. The logic holon provides a service directory; it is similar to PROSA's product holon, albeit more general. Using their reference architecture, Moneva et al. [9] are able to develop warehouse control software from an XML specification of the underlying warehouse equipment.

Hallenborg and Demazeau [5] have proposed an agent-based controller model for airport baggage handling systems. Their controller model consists of default agents for all standard elements of such a logistic system; they distinguish top loaders, diverters, mergers, straight segments, and dischargers. These element agents are supported by a routing agent, which is responsible for calculating a suitcase's route taking into account its urgency.

The fact that logistic systems can be controlled using software of which the structure resembles the underlying physical hardware is useful for model-driven development of logistic systems. One can use a single layout model, which forms the basis for both the physical hardware and the control software.

1.2 Outline

In this paper, we will demonstrate how domain-specific tooling can be used to support the development of logistic systems. We have selected the warehousing domain as a carrier for this demonstration. In particular, we will present domain-specific tooling for the development of warehouse management and control systems. This tooling involves a graphical configuration tool built on top of a reference architecture for warehouse management and control systems. Although we demonstrate the usage of domain-specific tooling in the warehousing domain, the underlying methods are also applicable to other logistic systems.

The warehousing domain and the corresponding domain-specific tooling are described in Sect. 2. We have applied the developed tooling to an automated case picking warehouse. The application of our tooling to this warehouse is presented in Sect. 3. In Sect. 4, we evaluate our tooling and its application to the automated case picking warehouse. Section 5 provides a summary of this paper.

2 Warehouse-Specific Tooling

Warehouses, or distribution centres, are facilities that receive and store goods of many suppliers. These goods are shipped to a large diversity of different customers. Thereby warehouses enable the efficient distribution of goods. The operations in a warehouse are controlled by a *warehouse management and control system* (WMCS). To achieve a high warehouse performance, a WMCS needs to use the warehouse's scarce resources efficiently.

Two important stakeholders in warehouse development are warehouse architects and WMCS developers. The former are specialists in the design of logistic processes. They are responsible for the selection of the required warehouse equipment and the operations to be carried out using this equipment. The specification of the warehouse's logistic process is transferred to WMCS developers, who are responsible for the implementation of the WMCS code that realises the specified logistic processes. The backgrounds of the warehouse architects and WMCS developers are quite different: warehouse architects have great knowledge of logistic processes, and generally little experience with the WMCS developers' expertise, i.e. software development. The communication between warehouse architects and WMCS developers often has not been formalised, making it highly ambiguous. This ambiguity causes misinterpretations resulting in rework by the WMCS developers and hence involves a larger WMCS development effort.

This section describes domain-specific tooling which allows a large reduction of the WMCS development effort by addressing the communication gap between warehouse architects and WMCS developers. The basis of the prototype tools is a WMCS reference architecture (or meta-model) that was developed by Verriet et al. [14, 15]. Section 2.1 gives a short description of this reference architecture. The reference architecture has been used to develop a WMCS configuration tool with which warehouse architects are able to configure a WMCS using warehouse-specific terminology. This tooling is presented in Sect. 2.2.

2.1 Warehouse Reference Architecture

Our reference architecture distinguishes five layers of WMCS functionality. The top layer is the *Enterprise Resource Planning* (ERP) layer, which is responsible for the high-level management of orders and stock. The second layer, the *planning* layer, handles the assignment of orders to resources, i.e. equipment and stock. The *scheduling* layer is responsible for balancing the system to obtain an optimal system performance. Scheduling involves selecting tasks to be executed and forwarding them to the *Material Flow Controller* (MFC) layer, which controls the warehouse equipment. The MFC layer provides an interface to the *Material Handling System* (MHS), which contains the warehouse equipment. In this paper, we will limit ourselves to the three top layers: ERP, planning, and scheduling. The MFC and MHS layers will be emulated in the scheduling layer.

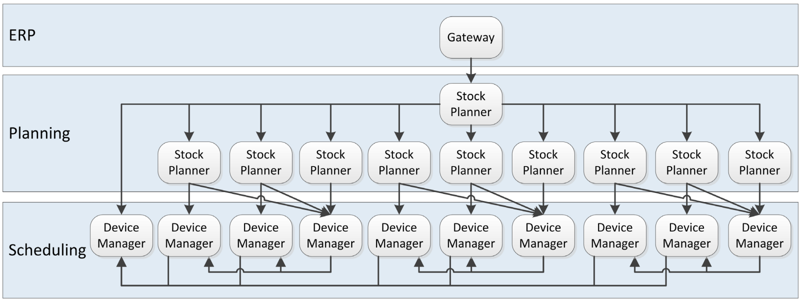


Fig. 1 Standard agent types: gateway, stock planner, and device manager

In each layer, there is one type of standardised agent. This is illustrated by the schematic WMCS organisation in Fig. 1. In the ERP layer, there is only one agent, a *gateway* agent, which plays the role of the ERP system. This agent has a list of customer orders and forwards these to the stock planners in the planning layer.

The planning layer's *stock planner* agents are responsible fulfilling incoming orders. Efficient order fulfilment is achieved by maintaining a certain level of stock. If its stock level drops below a specific minimum, a stock planner issues a replenishment order to other stock planners. Together the stock planners form a tree. Each stock planner, starting at the root stock planner, divides incoming work over its underlying agents, either other stock planners or device managers, taking into account their capabilities and work-in-process (WIP) level.

The *device manager* agents in the scheduling layer have a one-to-one correspondence to a warehouse's workstations. As illustrated in Fig. 1, the device managers form a network of producers and consumers: goods flow along the directed arcs from producers to consumers. This network corresponds to the physical transportation system connecting the warehouse workstations. Device managers are responsible for the sequencing and execution of work; hereby they take into account the (sequencing) requirements of their consumer device managers.

The gateway, stock planner, and device manager agents are standardised components that can be configured using their (structural) parameters. The main parameters of the stock planners are their initial stock level, their target WIP level, and their stock replenishment strategy. The device managers' parameters include their (nominal) throughput, i.e. the number of operations per hour, and the packaging types they can handle. The gateway agent is a specialisation of the stock planner agent. Compared to the stock planner agents, it has one additional parameter, a collection of orders to be forwarded. Apart from these attributes, the agents' parameters also include their connections to other agents. There are *parent-child* relationships between stock planners, *producer-consumer* relationships between device managers, and *associations* between stock planners and device managers.

Besides the structural parameters, the standardised agents also have behavioural parameters: each agent has a number of *behaviours*. The reference architecture includes a library of behavioural components. These are generic agent behaviours that

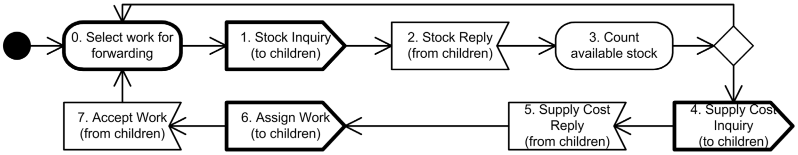

0. Select work for forwarding
1. Stock Inquiry (to children)
2. Stock Reply (from children)
3. Count available stock
4. Supply Cost Inquiry (to children)
5. Supply Cost Reply (from children)
6. Assign Work (to children)
7. Accept Work (from children)

Fig. 2 State machine of ForwardWork behaviour

have to be made application specific in order to obtain the desired overall system behaviour. The implementation of the reference architecture involves 30 generic behaviours that are implemented as state machines. These state machines define the standardised agent *interaction protocols*. These protocols are triggered by the reception of a message or the change of an agent's internal status.

An example is the state machine of the ForwardWork behaviour in Fig. 2. This behaviour is responsible for forwarding a stock planner's orders to its children. The behaviour starts by selecting an order and checking whether all stock needed is available in its children's local stock. The latter is achieved by sending them a StockInquiry message and receiving the corresponding StockReply answers. If the child stock planners have sufficient stock, the ForwardWork behaviour sends a Supply-CostInquiry to its children, which answer with a SupplyCostReply specifying the cost of supplying (parts of) the order. Based on the received replies, the Forward-Work behaviour assigns (parts of) the order to some of its children using an AssignWork message. The ForwardWork behaviour continues until it has completely forwarded the order and then selects the next order to be forwarded.

Note that the state machine in Fig. 2 has two types of states. The bold states have associated *business rules*. These are local optimisation rules, with which a behaviour can be configured. The ForwardWork behaviour in Fig. 2 has four business rules. The business rule of state 0 selects the next order to be forwarded to the child stock planners. The business rule of state 1 decides which children to ask for the availability of the stock for the selected order. State 4 has a business rule that decides which children to ask for the cost of supplying stock and the packaging in which this stock is to be delivered. State 6's business rule decides which parts of the order to assign to which child stock planners; this includes the packaging.

The reference architecture was implemented in Java using Jade [1], the Java Agent Development Framework. The implementation involves 30 *default behaviours* implemented as state machines. Together, these behaviours have 172 states and 73 business rules. The business rules in the default behaviours are called *default business rules*. These are application-independent business rule implementations. In the ForwardWork behaviour shown in Fig. 2, the default business rule of state 0 selects the oldest order for forwarding. State 1's default business rule asks all children for their available stock. The default business rule of state 4 asks all children for supply costs without changing any delivery requirements. Finally, state 6's default business rule assigns work to the child with the shortest delivery time.

To obtain the desired overall system behaviour, some of the default behaviours need to be made application specific. For instance, delivery packaging may need to

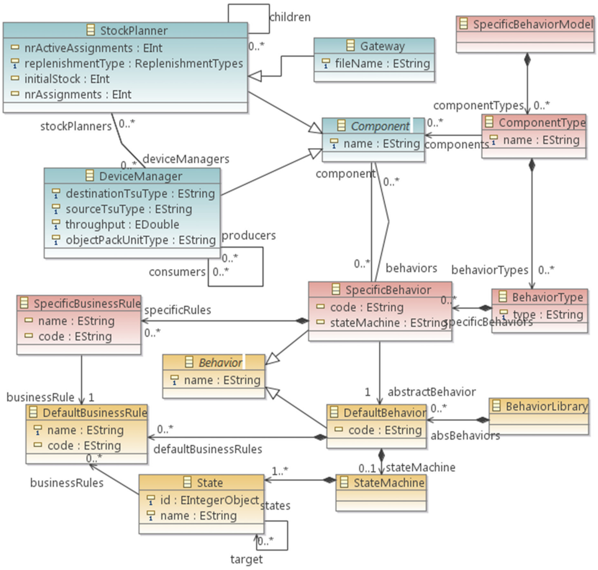


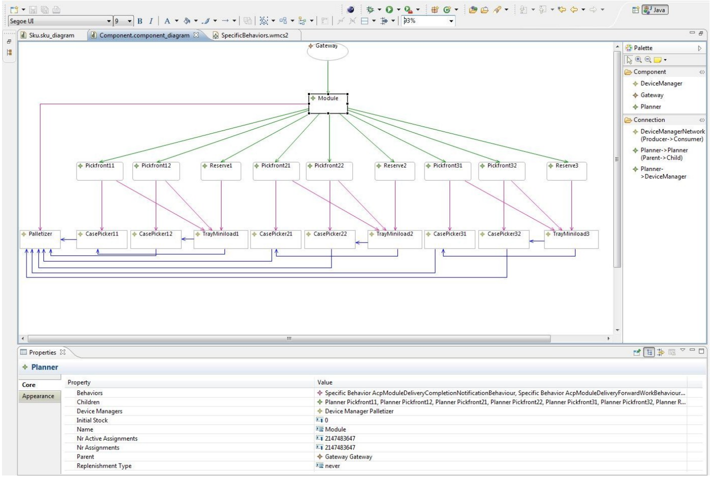
Fig. 3 Warehouse Control Specification Language (WCSL)

be changed due to the capabilities of the warehouse's equipment. A behaviour can be made specific by specialisation of a default behaviour, i.e. by overwriting some of a default behaviour's business rules. This is similar to the techniques proposed by Praehofer [11] to reuse simulation code.

2.2 WMCS Configuration Tool

The usage of the reference architecture described in the previous section requires software engineering skills that a warehouse architect may not have. To make the reference architecture accessible for warehouse architects, Liang [8] has developed a WMCS configuration tool. The elements of this tool are similar to those of the tooling of Trask et al. [12] used for a software product line for software-defined radio: it consists of a domain-specific language, a domain-specific graphical language, a domain-specific constraint language, and a domain-specific code generator. These elements will be discussed in this section.

The goal of the WMCS configuration tool is to offer a means to configure WMCSs with as little manual programming as possible. The basis of the tool is a DSL, called *Warehouse Control Specification Language* (WCSL). An Ecore [3] model of WCSL is depicted in Fig. 3. Note that WCSL contains all the elements of the reference architecture described in Sect. 2.1: it shows the WMCS components with their configuration parameters, the components' behaviours, and these behaviours' business rules.

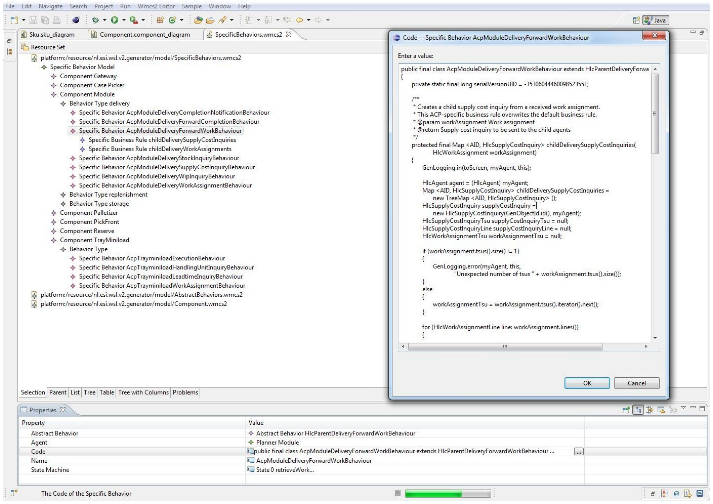


Property	Value
Behaviors	Specific Behavior AcpModuleDeliveryCompletionNotificationBehaviour, Specific Behavior AcpModuleDeliveryForwardWorkBehaviour...
Children	Planner Pickfront11, Planner Pickfront12, Planner Pickfront21, Planner Pickfront22, Planner Pickfront31, Planner Pickfront32, Planner R...
Device Managers	Device Manager Palletizer
Initial Stock	0
Name	Module
Nr Active Assignments	2147483647
Nr Assignments	2147483647
Parent	Gateway Gateway
Replenishment Type	never

Fig. 4 WST component diagram

The second main ingredient of our model-driven WMCS development approach is the configuration tool, called *Warehouse Specification Tool* (WST). This WMCS configuration tool is generated from the domain-specific language WSCL using EMF [3] and GMF [4]. Its main components include two diagrams, a component diagram and a behaviour diagram. The *component diagram* is shown in Fig. 4. With this diagram, a warehouse architect can create WMCS components by selecting the appropriate agent type from the palette on the right and placing it on the main canvas. Next, he can specify the configuration parameters of the newly created agent using the interface on the bottom. The warehouse architect can define relationships between agents using the different agent relations on the palette. These relations can be selected from the palette and drawn between agents on the canvas. To extend the support of the warehouse architect, WST also performs consistency checks using its *constraint language*. For instance, it checks whether all agent names are unique, relationships are drawn between the correct agent types, the stock planners form a tree, and there is at most one gateway agent.

With the other main diagram, the *behaviour diagram* (see Fig. 5), a warehouse architect can assign behaviours to agents. The specification of a behaviour starts with the selection of a default behaviour from the default behaviour library and the specification of the new behaviour's name. The corresponding Java code will directly be shown in the behaviour's properties (bottom part of Fig. 5). If needed, the warehouse architect can select some of the behaviour's business rules and, possibly with the help of a WMCS developer, overwrite them in a special code window (right part of Fig. 5). The newly written code will be merged with the default behaviour's code and updated in the behaviour's properties. After a new behaviour has been defined, it can be assigned to the agents that have been defined earlier. The latter can be done using the component diagram (see Fig. 4) or the behaviour diagram (see Fig. 5).



```
Resource Set
  platform:/resource/nl.esi.wsl.v2.generator/model/SpecificBehaviors.wmcs2
    Specific Behavior Model
      Component Gateway
      Component Case Picker
      Component Module
        Behavior Type delivery
          Specific Behavior AcpModuleDeliveryCompletionNotificationBehaviour
          Specific Behavior AcpModuleDeliveryForwardCompletionBehaviour
          Specific Behavior AcpModuleDeliveryForwardWorkBehaviour
            Specific Business Rule childDeliverySupplyCostInquiries
            Specific Business Rule childDeliveryWorkAssignments
          Specific Behavior AcpModuleDeliveryStockInquiryBehaviour
          Specific Behavior AcpModuleDeliverySupplyCostInquiryBehaviour
          Specific Behavior AcpModuleDeliveryWipInquiryBehaviour
          Specific Behavior AcpModuleDeliveryWorkAssignmentBehaviour
        Behavior Type replenishment
        Behavior Type storage
      Component Palletizer
      Component PickFront
      Component Reserve
      Component TrayMiniload
        Behavior Type
          Specific Behavior AcpTrayminiloadExecutionBehaviour
          Specific Behavior AcpTrayminiloadHandlingUnitInquiryBehaviour
          Specific Behavior AcpTrayminiloadLeadtimeInquiryBehaviour
          Specific Behavior AcpTrayminiloadWorkAssignmentBehaviour
  platform:/resource/nl.esi.wsl.v2.generator/model/AbstractBehaviors.wmcs2
  platform:/resource/nl.esi.wsl.v2.generator/model/Component.wmcs2
```

Code -- Specific Behavior AcpModuleDeliveryForwardWorkBehaviour

Enter a value:

```
public final class AcpModuleDeliveryForwardWorkBehaviour extends HlcParentDeliveryForwa
{
    private static final long serialVersionUID = -3530604446009852355L;

    /**
     * Creates a child supply cost inquiry from a received work assignment.
     * This ACP-specific business rule overwrites the default business rule.
     * @param workAssignment Work assignment
     * @return Supply cost inquiry to be sent to the child agents
     */
    protected final Map <AID, HlcSupplyCostInquiry> childDeliverySupplyCostInquiries(
            HlcWorkAssignment workAssignment)
    {
        GenLogging.in(toScreen, myAgent, this);

        HlcAgent agent = (HlcAgent) myAgent;
        Map <AID, HlcSupplyCostInquiry> childDeliverySupplyCostInquiries =
            new TreeMap <AID, HlcSupplyCostInquiry> ();
        HlcSupplyCostInquiry supplyCostInquiry =
            new HlcSupplyCostInquiry(GenObjectId.id(), myAgent);
        HlcSupplyCostInquiryTsu supplyCostInquiryTsu = null;
        HlcSupplyCostInquiryLine supplyCostInquiryLine = null;
        HlcWorkAssignmentTsu workAssignmentTsu = null;

        if (workAssignment.tsus().size() != 1)
        {
            GenLogging.error(myAgent, this,
                    "Unexpected number of tsus " + workAssignment.tsus().size());
        }
        else
        {
            workAssignmentTsu = workAssignment.tsus().iterator().next();
        }

        for (HlcWorkAssignmentLine line: workAssignment.lines())
        {
```

Property	Value
Abstract Behavior	Abstract Behavior HlcParentDeliveryForwardWorkBehaviour
Agent	Planner Module
Code	public final class AcpModuleDeliveryForwardWorkBehaviour extends HlcParentDeliveryForwardWorkBehaviour ...
Name	AcpModuleDeliveryForwardWorkBehaviour
State Machine	State 0 retrieveWork...

Fig. 5 WST behaviour diagram

After all WMCS components and their behaviours have been specified, WST allows the corresponding WMCS code to be generated. For this, WST uses Acceleo [2]: Acceleo performs model-to-text transformations using OMG's Model to Text Transformation Language (MTL). It takes an MTL specification and a specified WMCS configuration as input and generates WMCS code.

WST also provides the possibility to run the generated WMCS code as a software-in-the-loop simulation. The output of this simulation can be visualised using WST's Gantt chart tool. This tool, which is built using JFreeChart [10], takes the logging written during the execution of the generated WMCS and visualises an execution chart. These Gantt charts provide quick feedback to the warehouse architect: e.g. it shows whether the desired system throughput can be achieved using the specified warehouse equipment and the generated WMCS software. With this, the warehouse architect can change his specification until the desired system performance is achieved.

3 Automated Case Picking Case Study

We will use the system shown in Fig. 6 to demonstrate an application of the model-driven WMCS development approach described in Sect. 2. It shows an Automated Case Picking (ACP) module with a palletiser and three case picking cells. Each of a module's cells consists of two pick fronts, a reserve, two case pickers, and a tray miniload. The pick fronts and the reserves are storage racks containing trays with cases. Case pickers are devices that can pick cases from a pick front's trays; tray miniloads are capable of handling the trays of a reserve.

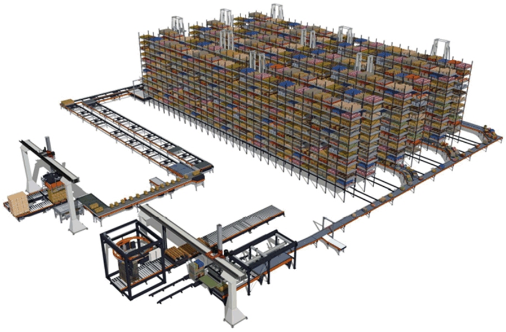


Fig. 6 Automated case picking module

The ACP module in Fig. 6 is responsible for delivering mixed pallets, i.e. pallets built from a collection of different cases. If a module is asked to deliver a pallet, it will ask the pick fronts to deliver the required cases using the corresponding case picker. If a pick front cannot deliver a case of certain product, it can be replenished from a reserve: the corresponding tray miniload can pick entire trays and place them in the pick front's local stock.

The ACP module's component structure is shown in the WST component diagram in Fig. 4. The top agent is a gateway agent, which represents the ERP system. It assigns a collection of orders to a tree of stock planners consisting of the module and its children: six pick fronts and three reserves. The palletiser, case pickers, and tray miniloads are device managers that reside in the scheduling layer. The palletiser is associated to the module, each case picker to one pick front, and each tray miniload to one reserve and two pick fronts. The device managers' producer-consumer network shows that a tray miniload delivers goods (i.e. trays) to two case pickers and that all case pickers deliver goods (i.e. cases) to the palletiser.

The bottom part of Fig. 4 shows the properties of the module stock planner. It shows the module's behaviours, its parent, its child stock planners, its connected device managers, its initial stock level, and a few other parameters.

The total number of different ACP-specific behaviours is 49. If the instances of the different agent types are counted, there are 21 different agents with 173 behaviours. Although these behaviours have many business rules, few have been made ACP-specific: only 15 business rules had to be changed in order to obtain the desired ACP system behaviour. For instance, for the module stock planner, one of the default business rules of the ForwardWork behaviour in Fig. 2 had to be overwritten. The business rule of state 4 needs to do a *packaging transformation*: it takes an order for pallets and translates this into orders for the contained cases.

The small number of overwritten business rules is due to the fact that all agents of one type share the same behaviours and there are only three types of packaging, i.e. case, tray, and pallet. The code of the overwritten business rules is the only code that was written by hand; all other code was generated from WST. Only 310 out of nearly 9,000 lines of code had to be written by hand. This is just 3 percent of the WMCS code; the remaining 97 percent of the code was generated automatically.

4 Reflection

In the previous sections, we have introduced domain-specific tooling for WMCS development and have applied this tooling to an ACP module. Using our prototype tools, we are able to specify and generate a WMCS. For the ACP module, this proved to be very successful, since only 3 percent of the WMCS code needed to be written by hand. This low percentage is partially due to the characteristics of ACP. There are only three types of packaging (case, tray, and pallet) and only two packaging transformations (from pallet to case for the module's pallet delivery and from case to tray for pick front replenishment). Moreover, the ACP stock planners and device managers work according to an oldest-work-first strategy. Since this strategy is implemented by the behaviours' default business rules and the number of packaging transformations is small, only few business rules needed to be overwritten to obtain the desired WMCS configuration. For a different system or another type of logistic process the percentage of hand-written code may be larger.

However, there is more to the WMCS development effort than the amount of manual coding. Currently, warehouse architects define a warehouse architecture and communicate this architecture to the WMCS developers. This architectural specification involves the selected warehouse equipment and a description of the logistic process to be implemented using this equipment. The communication has not been formalised meaning that it is highly ambiguous. The misinterpretations caused by the ambiguity necessitate rework by the WMCS developers and hence involves a larger WMCS development effort. Our domain-specific tooling can eliminate a large part of this communication ambiguity, as it provides a clear structure for the WMCS architecture, the logistic process in particular. This structure will leave little room for misinterpretation, because the code to be implemented by WMCS developers only involves local business rules, which have a well-defined scope and interface.

5 Conclusion

In this paper, we have shown that domain-specific tooling can improve the development of logistic systems by bridging the communication gap between specification and implementation. Logistic systems are characterised by a highly modular structure and a loose coupling of its components and the corresponding processes. We have used the warehousing domain to illustrate the benefits of domain-specific tooling, but the underlying methods apply to other types of logistic systems as well. We have presented a warehouse-specific specification language and used it to create a graphical configuration tool for WMCSs. The warehouse-specific tooling exploits the modular structure and loose coupling by using a library of structural and behavioural components. Using these parameterised components a warehouse architect can configure a WMCS. The tooling was demonstrated for an automated case picking warehouse. It was shown that the communication gap between warehouse architects and WMCS developers can be reduced by the parameterised components and behaviours and local optimisation rules of our domain-specific tooling. As a result, the probability of misunderstandings between development stakeholders is reduced greatly.

Acknowledgements. This work has been carried out as part of the FALCON project under the responsibility of the Embedded Systems Institute with Vanderlande Industries as the carrying industrial partner. This project is partially supported by the Netherlands Ministry of Economic Affairs under the Embedded Systems Institute (BSIK03021) program.

References

1. Bellifemine, F., Caire, G., Greenwood, D.: Developing Multi-Agent Systems with JADE. John Wiley & Sons, Ltd., Chichester (2007)
2. Eclipse Foundation: Acceleo (2012), http://www.eclipse.org/acceleo/ (viewed April 2012)
3. Eclipse Foundation: Eclipse Modeling Framework Project, EMF (2012), http://www.eclipse.org/modeling/emf/ (viewed April 2012)
4. Eclipse Foundation: Graphical Modeling Project, GMP (2012), http://www.eclipse.org/modeling/gmp/ (viewed April 2012)
5. Hallenborg, K., Demazeau, Y.: DECIDE: Applying Multi-agent Design and Decision Logic to a Baggage Handling System. In: Weyns, D., Brueckner, S.A., Demazeau, Y. (eds.) EEMMAS 2007. LNCS (LNAI), vol. 5049, pp. 148–165. Springer, Heidelberg (2008)
6. Hvam, L., Haug, A., Mortensen, N.H.: Assessment of benefits from product configuration systems. In: ECAI 2010 Workshop on Configuration (2010)
7. Leitão, P., Restivo, F.: ADACOR: A holonic architecture for agile and adaptive manufacturing control. Comput. Ind. 57, 121–130 (2006)
8. Liang, H.L.: A graphical specification tool for decentralized warehouse control systems. SAI technical report, Eindhoven University of Technology (2011)
9. Moneva, H., Caarls, J., Verriet, J.: A Holonic Approach to Warehouse Control. In: Demazeau, Y., Pavón, J., Corchado, J.M., Bajo, J. (eds.) 7th International Conference on PAAMS 2009. AISC, vol. 55, pp. 1–10. Springer, Heidelberg (2009)
10. Object Refinery Limited: Jfreechart (2012), http://www.jfree.org/jfreechart/ (viewed April 2012)
11. Praehofer, H.: Object oriented, modular hierarchical simulation modeling: Towards reuse of simulation code. Simulat. Pract. Theor. 4, 5–8 (1996)
12. Trask, B., Paniscotti, D., Roman, A., Bhanot, V.: Using model-driven engineering to complement software product line engineering in developing software defined radio components and applications. In: 21st ACM SIGPLAN Symposium on Object-oriented Programming Systems, Languages, and Applications, pp. 846–853 (2006)
13. Van Brussel, H., Wyns, J., Valckenaers, P., Bongaerts, L., Peeters, P.: Reference architecture for holonic manufacturing systems: PROSA. Comput. Ind. 37, 255–274 (1998)
14. Verriet, J., van Wijngaarden, B.: A reference architecture capturing structure and behaviour of warehouse control. In: Hamberg, R., Verriet, J. (eds.) Automation in Warehouse Development, pp. 17–32. Springer, London (2012)
15. Verriet, J., van Wijngaarden, B., van Heusden, E., Hamberg, R.: Automating the Development of Agent-Based Warehouse Control Systems. In: Corchado, J.M., Pérez, J.B., Hallenborg, K., Golinska, P., Corchuelo, R. (eds.) Trends in PAAMS. AISC, vol. 90, pp. 59–66. Springer, Heidelberg (2011)
16. Westland, J.C.: The cost of errors in software development: evidence from industry. J. Syst. Software 62, 1–9 (2002)

Chapter 12
Smart Grid: Constructing a System of Systems Model Using Both Qualitative and Quantitative Assessments

Michael Z. Miller, Satya S. Pogaru, and Dimitri N. Mavris

Abstract. When constructing a model of a system or system of systems, one usually decides between utilizing a qualitative model or a quantitative model. There exists a desire to leverage both modeling approaches by finding a way to compare and contrast both types of models. In this paper the authors make attempts to leverage these modeling techniques by exploring the use of a newly formulated methodology, referred to as Relational Oriented Systems Engineering and Technology Tradeoff Analysis (ROSETTA) as a way to compare qualitative assessment and quantitative analysis. This was applied to a sample problem of constructing a model of the Smart Grid. As a proof of concept, instead of examining the entire Smart Grid, only Demand Response and day-ahead load prediction were assessed. Qualitatively, the Quality Function Deployment methodology was used as a representative means to capture Subject Matter Expert (SME) opinion. On the quantitative side, an Agent-Based Modeling approach augmented with elements of Discrete Event Simulation was employed to construct a physics-based model. The use of ROSETTA allows for communication between the SMEs and the modelers, which was used to improve the accuracy of both models. This improvement comes from the iterations of the qualitative assessments and quantitative analyses, successively building off of insights gained from previous iterations. This paper shows the first steps in leveraging the benefits of both qualitative and quantitative models.

Michael Z. Miller · Satya S. Pogaru · Dimitri N. Mavris
Georgia Institute of Technology
Aerospace Systems Design Laboratory
275 Ferst Dr. NW
Atlanta, GA 30332
e-mail: {mzmiller,spogaru}@asdl.gatech.edu,
dimitri.mavris@aerospace.gatech.edu

1 List of Abbreviations

ABM – Agent-Based Modeling
DES – Discrete Even Simulation
DG – Distributed Generation
DisCo – Distribution Company (Power Distributor/Generator)
DLC – Direct Load Control
DR – Demand Response
DS – Distributed Storage
HoQ – House of Quality
HVAC – Heating, Ventilation, and Air-Conditioning
M&S – Modeling and Simulation
m-space – 'System-Level Metrics' space
MCS – Monte Carlo Simulation
PMV – Predicted Mean Value
QFD – Quality Function Deployment
ROSETTA – Relational Oriented Systems Engineering and Technology Tradeoff Analysis
R-space – 'Requirements' space
SE – Systems Engineering
SME – Subject Matter Expert
SoS – System of Systems
VP – Variable Pricing
x-space – 'Design Variable' space

2 Motivation

A disconnect exists between the SMEs and the modelers when assessing a complex system or system of systems (SoS), especially early in the design process where behavior of a system or SoS is more uncertain. The SMEs perform qualitative analysis, often using Systems Engineering (SE) practices that can be captured using Quality Function Deployment (QFD), which will be further explained later in this paper; therefore, the building of this qualitative model can be thought of as a SME-driven process. Modelers often seek to gain insight into a complex system by performing quantitative analysis using physics-based models. It is believed that neither of these techniques can independently provide as much insight into a complex system or SoS as some way of combining these techniques. There is a need for a way to leverage both qualitative assessment and quantitative analysis in order to construct a desirable model. Since the qualitative assessments and quantitative analysis are not done using the same tools/techniques, there is a need to translate between the SMEs and modelers. ROSETTA allows for this translation to occur, and this approach may be applied to guide model construction; more details about ROSETTA will be discussed later in this paper. To understand how ROSETTA can be used to translate between the qualitative assessments and quantitative analysis, a model of the Smart Grid was constructed. This sample problem built off of work done by Duncan, et al. [2] They showed how the ROSETTA methodology could be applied to the Smart Grid, although they discussed the use ROSETTA has a framework to gain insight into the Smart Grid SoS, not necessarily to aid in construction of a SoS model. [2]

3 Background on Smart Grid

The current energy grid was conceived over 120-years-ago, and the grid is reaching its limitations in terms of capacity, efficiency, and life period.[12] These limitations directly manifest themselves as problems for both the power distributor/generator (DisCo) and the customers relying on the electricity; some examples of these problems include: blackouts; huge losses associated with distribution and transmission; inability to quickly locate and isolate faults; and integration of renewable sources of electricity production, which are intermittent; etc. These issues have lead to a push in recent years for large, system-wide upgrades to, what is commonly referred to as, the Smart Grid. [12] One of the enabling technologies for the Smart Grid is two-way communication between the DisCo and the power consumers. This allows for the power consumption to be monitored from generation to end use, which can then enable implementation of several different programs that work to improve the operability, efficiency, and capacity that is needed to meet future power demands. Some of these programs include: Demand Response (DR), better utilization of Distributed Generation (DG), better utilization of Distributed Storage (DS), Load Balancing, and Fault Isolation. In order to accurately assess the overall impact of these programs on the energy grid, one needs a holistic understanding of the Smart Grid as a SoS.

To gain insight into the Smart Grid SoS, one needs to examine the individual technologies of the various programs that will be implemented. DR seeks to reduce the peak load, which minimizes the use of less efficient and more expensive peaking plants. This can be used to help maximize the profits of the DisCo. [12] DG seeks to increase efficiency in power distribution by moving from a few, large-scale power production facilities to a larger number of distributed small-scale power production units; this small-scale power production could be small enough to supply partial power for one residential home, or large enough to support an entire neighborhood. [12] DS, which is sometimes coupled with DG, seeks to use multiple storage facilities/types to use when power demand is high, or energy production is low. [12] Load Balancing involves transferring power at a certain current in order to maximize the efficiency of electricity transmission. [10] Fault Isolation seeks to locate and to minimize service interruptions that result from hardware failures. The current energy grid accomplishes Fault Isolation by relying on reporting from energy consumers, but the Smart Grid seeks to automate this process. [4] Each of these programs has several different ways of being implemented, which drastically increases the complexity when performing a SoS analysis of the Smart Grid. In order to gain insight into such a complex SoS, one must determine what parameters of the various Smart Grid programs and technologies are necessary to model in order to gain adequate insight.

Due to the complex nature of the Smart Grid SoS, modeling all the parameters associated with the different programs can quickly become infeasible because of the computational requirements. As a proof of concept, one can only examine some of the Smart Grid programs and technologies in order to gain insight into the construction a SoS model. The rest of this paper will only focus on DR and day-ahead load predictions.

3.1 Demand Response

DR is defined by Federal Energy Regulatory Commission as, "changes in electric usage by demand-side resources from their normal consumption patterns in response to changes in the price of electricity over time, or to incentive payments designed to induce lower electricity use at times of high wholesale market prices or when system reliability is jeopardized". [3] This change in consumption behavior reduces the likelihood of forced outages or full-scale blackouts. [13] DR can be achieved either through Direct Load Control (DLC) or Variable Pricing (VP). DLC is when the residential user temporarily cedes control of some of their appliances to the DisCo. The most frequently discussed VP schemes are: Dynamic Pricing and Real-Time Pricing [1]. To minimize computational resources, only one general form of DLC being the only DR scheme modeled for the proof of concept of the method.

3.2 Load Prediction

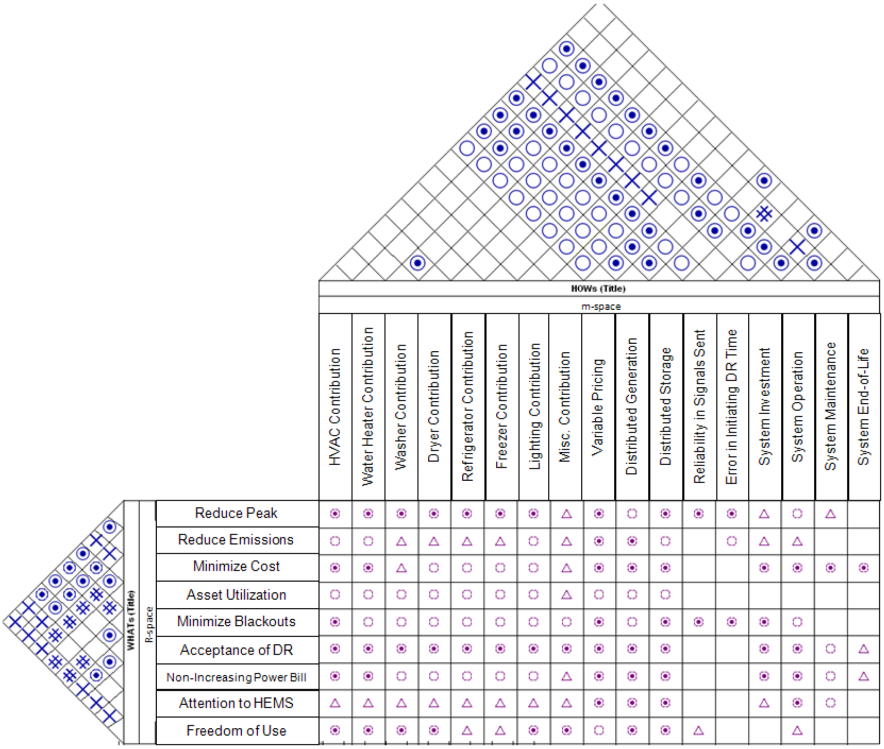
The load prediction was examined in order to assess its impact on the DR scheme modeled. Two types of load predictions are commonly used. The first type was done through day-ahead load prediction model, where the actual load data of similar day over the past five years was aggregated to get the prediction for the desired day. The second type is done through real-time prediction of the load, which can be done by using various prediction algorithms. This proof of concept only considers day-ahead predictions.

4 Modeling and Simulation

As previously discussed, both a qualitative model and a quantitative model were needed. These models were compared to each other in order to gain further insight into the modeling of this complex SoS. A physics-based model was developed, using lower-fidelity approximations to model the behavior of the Smart Grid systems that were modeled. The ROSETTA framework utilizes the QFD methodology for the qualitative model. In the QFD, SMEs were able to identify relationships between systems in order to gain insight into the behavior of the SoS.

4.1 Qualitative Modeling

The qualitative model was constructed using data from SMEs and captured using a QFD. This begins by constructing a top-level House of Quality (HoQ) showing the various technologies of the Smart Grid; this HoQ, which maps the stakeholders' requirements (R-space) to the SoS-level metrics (m-space), can be seen in Figure 1. The next step of QFD was to look at the next lower level HoQ, which maps the m-space to the design variables (x-space). This was done to ultimately create a mapping between the x-space and the R-space. This is shown in Figure 2.

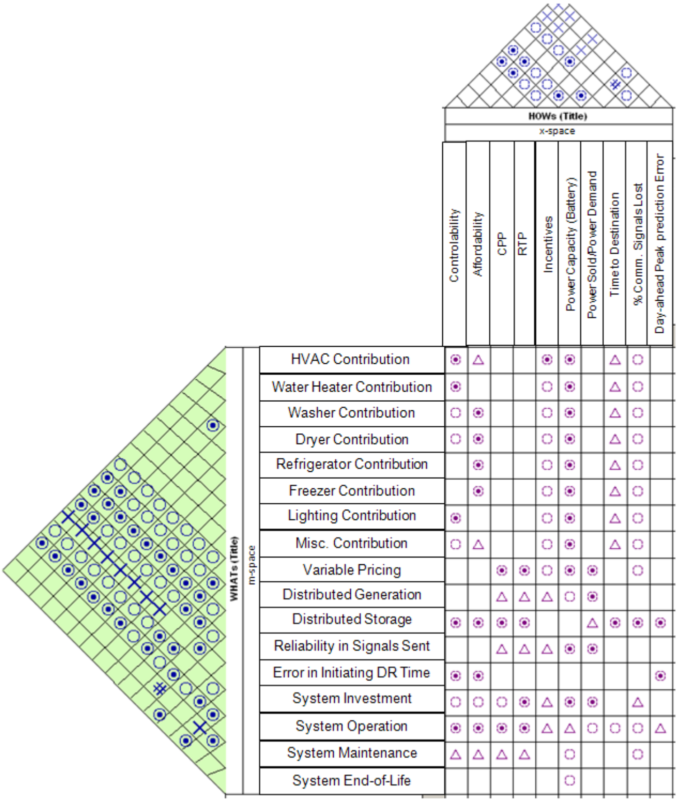


	HVAC Contribution	Water Heater Contribution	Washer Contribution	Dryer Contribution	Refrigerator Contribution	Freezer Contribution	Lighting Contribution	Misc. Contribution	Variable Pricing	Distributed Generation	Distributed Storage	Reliability in Signals Sent	Error in Initiating DR Time	System Investment	System Operation	System Maintenance	System End-of-Life
Reduce Peak	⊙	⊙	⊙	⊙	⊙	⊙	⊙	△	⊙	○	⊙	⊙	⊙	△	○	△	
Reduce Emissions	○	○	△	△	△	△	○	△	⊙	⊙	○		○	△	△		
Minimize Cost	⊙	⊙	△	○	○	○	○	△	⊙	⊙	⊙			⊙	⊙	⊙	⊙
Asset Utilization	○	○	○	○	○	○	○	△	○	○	○						
Minimize Blackouts	⊙	○	○	○	○	○	○	○	⊙	○	⊙	⊙	⊙	⊙	○		
Acceptance of DR	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙	⊙			⊙	⊙	○	△
Non-Increasing Power Bill	⊙	⊙	○	○	○	○	○	△	⊙	⊙	⊙			⊙	⊙	○	△
Attention to HEMS	△	△	△	△	△	△	△	△	⊙	⊙	⊙			△	⊙	○	
Freedom of Use	⊙	⊙	⊙	⊙	△	△	⊙	⊙	○	⊙	⊙	△			△		

Fig. 1 First-level HoQ of the Smart Grid showing the relationships between the Requirements (R-Space) and System-Level Metrics (m-Space)

Since this paper is not trying to build a model for the entire Smart Grid, the qualitative model was reduced to only include aspects of DR and day-ahead load prediction. Within DR, there are several appliances that were not considered because of the impracticality of controlling those for residential consumers (e.g., lights and miscellaneous appliances). [12] The following analysis utilized information presented on the scoped-down QFDs shown in Figure 3 and Figure 4.

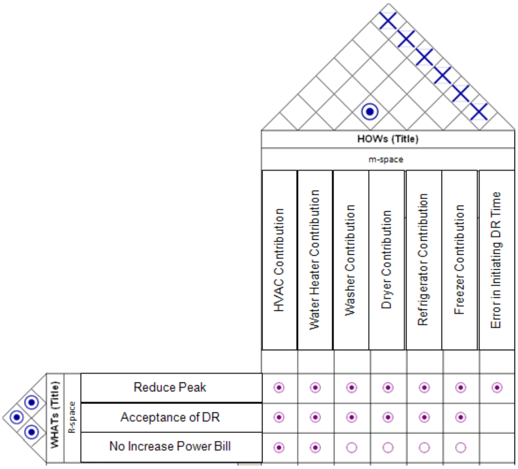
The top-level HoQ, shown in Fig. 3, maps the R-space to the m-space. Here the R-space is consisting of the stakeholders involved in the DR, namely the DisCo and consumers; government regulators were not considered for brevity. The



	Controlability	Affordability	CPP	RTP	Incentives	Power Capacity (Battery)	Power Sold/Power Demand	Time to Destination	% Comm. Signals Lost	Day-ahead Peak prediction Error
HVAC Contribution	⊙	△			⊙	⊙		△	○	
Water Heater Contribution	⊙				○	⊙		△	○	
Washer Contribution	○	⊙			○	⊙		△	○	
Dryer Contribution	○	⊙			○	⊙		△	○	
Refrigerator Contribution		⊙			○	⊙		△	○	
Freezer Contribution		⊙			○	⊙		△	○	
Lighting Contribution	⊙				○	⊙		△	○	
Misc. Contribution	○	△			○	⊙		△	○	
Variable Pricing			⊙	⊙	○	⊙	⊙		○	
Distributed Generation			△	△	△	○	⊙			
Distributed Storage	⊙	⊙	⊙	⊙			△	⊙	⊙	⊙
Reliability in Signals Sent			△	△	△	⊙	⊙			
Error in Initiating DR Time	⊙	⊙								⊙
System Investment	○	○	○	⊙	△	⊙	⊙		△	
System Operation	⊙	⊙	⊙	⊙	△	△	○	○	○	△
System Maintenance	△	△	△	△		○			○	
System End-of-Life						○				

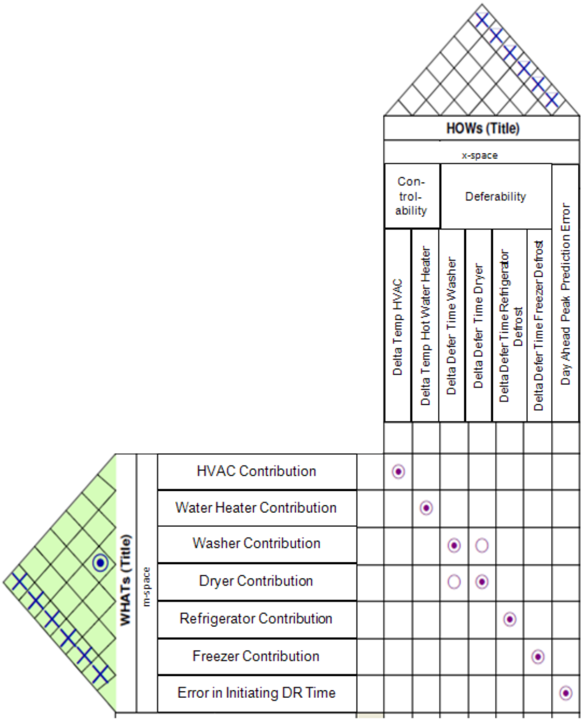
Fig. 2 Second-level HoQ showing the relationships between System-Level Metrics (m-Space) and Design Variables(x-Space)

stakeholders have unique interests/requirements for a DR program. As an example, the DisCo might be interested in DR to reduce the peak loads that cause high power production costs, while consumers might participate in order to decrease their power bills or at least to keep their power bills from increasing. The m-space defines the possible ways to meet the stakeholders' interests/requirements. The second HoQ, shown in Figure 4, maps the m-space to the x-space, which are the variables used to describe these programs in order to see if they achieve the requirements in the R-space. Besides these mappings, the QFD also maps correlations among parameters in the R-space, m-space, and x-space, by utilizing the 'roof' and the 'greenhouse' of the HoQ. As an example, consider the Heating, Ventilation, and Air Conditioning (HVAC) (from the m-space), which is controllable, but the washer or dryer are deferrable, but currently assumed to not be controllable. For brevity, the other mappings are not explicitly stated, but can be seen in Figure 3 and Figure 4.



	HVAC Contribution	Water Heater Contribution	Washer Contribution	Dryer Contribution	Refrigerator Contribution	Freezer Contribution	Error in Initiating DR Time
Reduce Peak	⦿	⦿	⦿	⦿	⦿	⦿	⦿
Acceptance of DR	⦿	⦿	⦿	⦿	⦿	⦿	
No Increase Power Bill	⦿	⦿	○	○	○	○	

Fig. 3 Down-selected first-level HoQ of Demand Response



	Con-trol-ability		Deferability				
	Delta Temp HVAC	Delta Temp Hot Water Heater	Delta Defer Time Washer	Delta Defer Time Dryer	Delta Defer Time Refrigerator Defrost	Delta Defer Time Freezer Defrost	Day Ahead Peak Prediction Error
HVAC Contribution	⦿						
Water Heater Contribution		⦿					
Washer Contribution			⦿	○			
Dryer Contribution			○	⦿			
Refrigerator Contribution					⦿		
Freezer Contribution						⦿	
Error in Initiating DR Time							⦿

Fig. 4 Down-selected second-level HoQ of Demand Response

4.2 Quantitative Modeling

No publicly available software was found that captured all of the desired metrics for DR. This was a large motivator for building an M&S environment. The model constructed was an Agent-Based Model (ABM) [14] that utilized discrete event simulation (DES) in order to minimize simulation run time. An ABM was chosen in order to capture the uniqueness of different residential energy consumption users and then aggregate individual consumptions to generate system level data. It is important to note that this quantitative model was being used early in the design process to help gain insight into the complex SoS. For this reason, the model constructed could be of lower fidelity, which meant that utilizing DES was acceptable (i.e., there was no need to model the exact flow of power and minute-by-minute actions of the consumer at this phase in the design process).

In the model used, the agents are defined as the residential energy consumers; the methodology of constructing this model builds off of work presented by Lewe, et al. [6] The agents in the model had a level of sentience that was limited to responding to changes in their comfort level due to the internal temperature. This was modeled using Fanger's equation on comfort. [8] In this model, one agent represents the behavior of all of the residents in the household. This is advantageous for several reasons: 1) Fanger's equation calculates a predicted mean value (PMV) of comfort for a group of people; 2) the energy consumption of one person is not as predictable as the energy consumption of a family [9]; 3) this reduces the simulation run time because it reduces the number of agents. The simulation was run with 1000 agents (i.e., households), each with unique characteristics (e.g., house size, thermal efficiency, number of residents, daily schedule, etc.). All actions, not related to thermal comfort, of the agent were governed by randomized discrete events. Besides agent actions, several appliances (e.g., washers, dryers, refrigerators, and freezers) were modeled using probabilistic DES, from Department of Energy consumption data. [3] A summary of modeling details can be seen in Table 1. Changes in internal temperature used in the M&S environment were based on fundamental heat transfer equations. [5] The external temperature was modeled using a hot summer day with the temperature varying between 70°F and 95°F.

Table 1 Summary of the appliance control in the quantitative model

	HVAC	**Water Heater**	**Washer**	**Dryer**	**Freezer Defrost**	**Refrigerator Defrost**	**Other Power Loads**
Agent Controls	X						
Probability Schedule [3]		X	X	X	X	X	X

In order to guide construction of an accurate SoS model, the quantitative model needs to track metrics stated in the qualitative model; there is a need to track all metrics because previously unrealized relationships or trends might arise from the quantitative model. Most of the metrics were obvious to track, but the system metric of ‘Acceptance of DR’ was quantified by keeping track of the inverse of the agents’ total annoyance in response to DR events. This total annoyance was the aggregation of two types of annoyance: 1) thermal discomfort, which occurs when their thermostat is set too high or low from their comfortable setting, and 2) deferring annoyance, which occurs when they are unable to perform an activity (e.g., laundry) because a DR event differed that load. Hence, it should be noted that ‘Acceptance of DR’ is a measure of “how much less intrusive the given DR is”. In the second-level HoQ, which is shown in Figure 4, the ‘delta Temperatures’ refer to the change in the set temperature caused by a DR event. The ‘Delta Defer Time X’ metrics refer to the length of time that a DR event would defer the use of that specific appliance after the end of DR event.

5 Leveraging Both Models

ROSETTA utilizes surrogate models of the quantitative model in order to rapidly perform a Monte Carlo Simulation (MCS), which is used to establish relationships and trends within the HoQs. After performing MCS on the surrogate models of the developed quantitative model, the correlations, sensitivities, and impacts of the design variables can be examined. These surrogate models were constructed using Response Surface Methodology. Examining the correlations, sensitivities and impacts of the design variables is important in order to understand what aspects of the M&S environment need further development. The quantitative results of second-level HoQ (refer to Figure 4) can be seen in Fig. 5. The results from the quantitative provide more insight into the relationships established from qualitative model. It is important to note that all of the red circles/ellipses in the roofs and greenhouses of the HoQ of Figures 5, 6, 7, and 8 show the correlations between those requirements, metrics, or design variables. These correlations result from the MCS, and each black point represents a data point from a simulation run. A red circle indicates no correlation, whereas an ellipse indicates a correlation in the direction of the major axis of that ellipse.

The ROSETTA results shown in Figure 5 demonstrates the quantitative model results being translated to a form that is easily understood by the SMEs, whom perform the qualitative analysis. When comparing the qualitative and quantitative model assessments, it can be see that the ‘Day Ahead Peak Prediction Error’ has an impact on more parameters in the m-space than was previously captured by the qualitative assessment alone (refer Figure 3 and Figure 4). In fact, most appliances seem to be more sensitive to the ‘Day Ahead Peak Prediction Error’ than other variables. Also from Figure 5, two significant points should be observed: 1) The variation of the appliance contributions with respect to the ‘Day Ahead Peak Prediction Error’, all of the discrete appliances (Washer, Dryer, Fridge, and Freezer) show almost similar variation trend (though not same in terms of magnitude), and 2) from the prediction profilers, it can be observed that appliance contributions

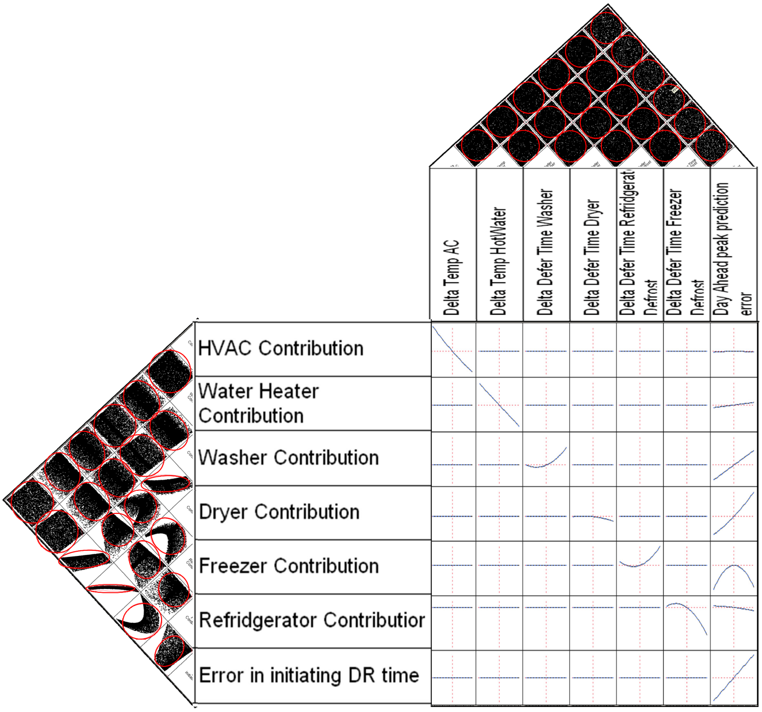


Fig. 5 The second-level HoQ that results of the first iteration of modeling

depend mainly on 'Day Ahead Peak Prediction Error'. For the above explained two significant points, it is believed that there exist correlations among the discrete appliances in m-space (Figure 5). Some of these correlations can be seen as trends in the greenhouse (m-space) of the QFD in Figure 5. When closely examining these results, it is important to note the non-linear behavior exhibited by the Freezer, with respect to 'Day Ahead Peak Prediction Error.' Since this behavior was exhibited only with the Freezer, one can begin to speculate that there might exists a (some) error(s) in this low-fidelity model. This is one of the key benefits to using ROSETTA for aiding in model construction – one can then take the quantitative results to the SMEs for re-evaluation because ROSETTA allows for easy communication between quantitative analysis and qualitative assessments.

This re-examination by the SMEs not only showed that the something must be incorrect with the modeling of the Freezer, but several other relationships did not make intuitive sense. One of the major errors observed is that the 'Freezer Contribution' varied (nonlinearly) with the 'Delta Defer Time Refrigerator Defrost' and that 'Refrigerator Contribution' varied (nonlinearly) with the 'Delta Defer Time Freezer Defrost.' Examining this closely showed that the metrics being tracked within the model were incorrect; however, one should note that even someone with no experience in the field could have noticed this error in the model because it was a coding problem. With this in mind, it is important to note the true benefits of ROSETTA, which creating a commonality between qualitative assessments and quantitative analysis in order to understand complex system (or SoS) behavior. Looking at this level HoQ, one should go back to the non-linear relationship that is shown between 'Freezer Contribution' and 'Day Ahead Peak Prediction Error.' This trend can be interpreted to mean that regardless of the 'Day Ahead Prediction Error' the Freezer does not contribute to the overall power consumption. By consulting SMEs, one can find that this trend is not correct because the Freezer's contribution to the peak should depend on the direction of the prediction error. This highlights an error in modeling the Freezer that was corrected in the next model iteration.

To further highlight the benefits of ROSETTA, one should examine the higher level HoQ, which relates the m-space to the R-space and can be seen in Figure 6. Looking at this figure, one notices the sensitivity to 'Reduce Peak' and 'Non-increasing Power Bill' to 'Dryer Contribution' that do not exist with the other appliances. These sensitivities can be interpreted to mean that the Dryer plays a larger role in these requirements than other appliances. By consulting SMEs, one realizes that the quantitative trend shown in this figure do not make intuitive sense because less 'Dryer Contribution' should result in more reduction of the peak load, not the opposite. This error was addressed in the next model iteration.

Besides these errors, the SMEs were able to help with presenting the results from the quantitative model in a more intuitive manner. By looking at Figure 5, one may initially think that 'HVAC Contribution' and 'Delta Temp AC' have an inverse relationship. This was done because the modelers believed that tracking an increase in 'Delta Temp AC' should result in a decrease in 'HVAC Contribution.' However, the SMEs suggested the modelers show that increasing 'Delta Temp AC' has an increase in the reductions from 'HVAC Contribution.' This feedback from the SMEs is basically saying that the modelers were not tracking metrics in an intuitive way for the SMEs (in this study) to understand the quantitative data. Again, this highlights the importance of ROSETTA in being able to translate between the SMEs and the modelers. This and other ways to more intuitively present the data were implemented in the next model iteration. The results of this second iteration can be seen in Figure 7 and Figure 8.

Many of the trends previously identified in the first round of modeling can be seen in the second iteration of modeling, but more importantly, all of the previously identified errors are corrected. When comparing Figure 5 and Figure 6 with Figure 7 and Figure 8, one can also notice that many of the trends and correlations are more tightly defined. However, when comparing Figure 7 and Figure 8 to Figure 3 and Figure 4, respectively, one notices several differences. The sensitivities shown in Figure 7 do not correspond to the relationships identified in Figure 4. Majority of the relationships identified in Figure 4 result in no or only small variations, in terms of sensitivities, when examined in Figure 7. For example, the HVAC shows a variation range of 1600KW in magnitude whereas the Freezer shows a variation range of 0.3KW, which is why it show little or variations in the sensitivity plots. Similarly, most of the other appliances also show considerably very less variation than HVAC. This can be interpreted as changes in the set

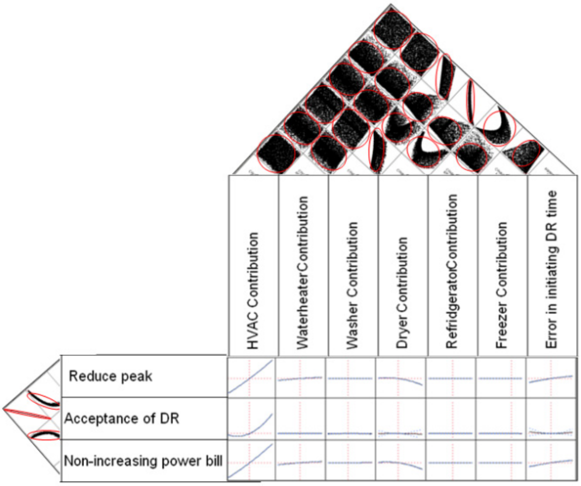


Fig. 6 The first-level HoQ that results from the first iteration of ROSETTA. Note: the 'Reduce Peak' metric counts reductions in peak as negative so that the correlation seen between 'Reduce Peak' and 'Acceptance of DR' was a strong, positive relationship.

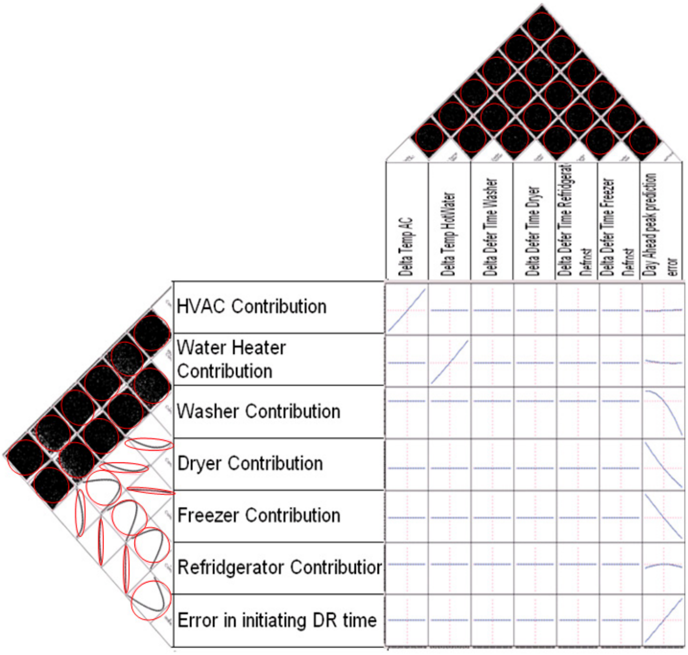

Delta Temp AC
Delta Temp HotWater
Delta Defer Time Washer
Delta Defer Time Dryer
Delta Defer Time Refridgerat Defrost
Delta Defer Time Freezer Defrost
Day Ahead peak prediction error
HVAC Contribution
Water Heater Contribution
Washer Contribution
Dryer Contribution
Freezer Contribution
Refridgerator Contributior
Error in initiating DR time

Fig. 7 Results of the second-level HoQ second iteration from ROSETTA

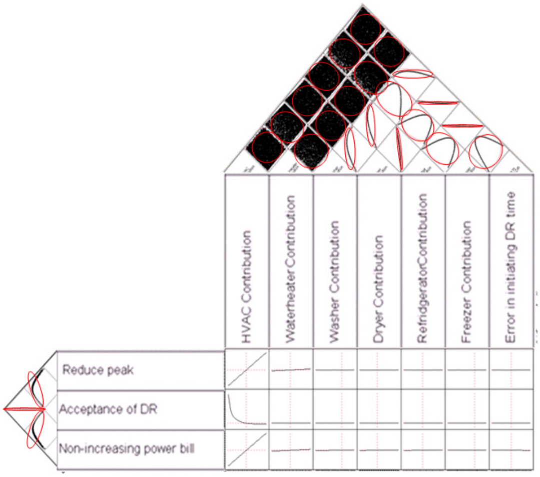


Fig. 8 Results of ROSETTA on first-level HoQ from the second iteration of modeling

temperature of the HVAC have more impact than changes in the deferral times of other appliances. Also, when comparing the results from Figure 7 and Figure 4, one notices that the results in Figure 7 can actually quantify the type (e.g., linear, quadratic, inverse, etc.) of relationship instead of just stating that a relationship exists. To further explain this, Figure 9 shows the three down-selected requirements. On the left of these requirements (i.e., the greenhouse in Figure 3) are the symbols used in QFD (i.e., qualitative assessment), whereas the symbols on the right side (i.e., the greenhouse in Figure 8) are the correlations seen from the quantitative analysis. The symbols on the left all indicate strong, positive correlations; however, the red ellipses on the right show the actual degree of these correlations. From examining this figure, one can see a stronger correlation between 'Reduce Peak' and 'Non-increasing Power Bill' as opposed to the other requirements. This correlations is stronger because the ellipse in this box not only has a larger slope (in terms of the major axis), but also a smaller minor axis than the other two ellipses. Due to space limitations, other correlations are not explicitly discussed.

Even in this second iteration, the importance of translating between the SMEs and the quantitative results of the model can be seen. For example, when analyzing the results, the SMEs begin to wonder what metrics or design variables accounted for the possibility of a rebound effect. A rebound effect is the synchronization of various loads after a DR event that causes the power consumption to spike to levels that could have been higher than the initial peak [11]. Currently, the model does not consider assessing potential rebound effects; this will be accounted for in future iterations of constructing this model. It is important to note that the SMEs (used for this paper) did not initially capture the idea of the rebound effect in the original QFD (Figure 1 and Figure 2), but this lack of inclusion was captured by examining the results from the quantitiative analysis, presented using ROSETTA (Figure 7 and Figure 8). Although this error might not have occurred if

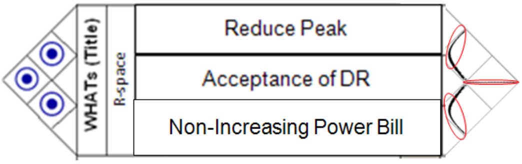


Fig. 9 Analyzing the correlations between requirements (i.e., the greenhouses in Figures 3 and 8)

more vetting was used in selecting SMEs, this example highlights the power of ROSETTA, especially when the definition of a SME can be rather nebulous.

From this HoQ, it can be seen that the qualitative assessment did not adequately capture the correlations between all the design variables. For example, it was believed that there would be a strong correlation between the 'HVAC Contribution' and the 'Error in initiating DR time', but because of the way HVAC is used throughout the day (i.e., the shape of the load curve from HVAC use throughout the day) there exist little, correlation between these design variables. However, the quantitative model did verify the correlation between 'Washer & Dryer Contribution' and 'Error in initiating DR time'. The differences in the impacts of these appliances on the error in initiating DR time were realized by using ROSETTA, and it can be accounted for in future iterations of constructing this model. It is also important to note the strong positive correlation between the requirements 'Reduce Peak' and 'Non-Increasing Power Bill', which was expected. These requirements show an interesting correlation when compared to the requirement 'Acceptance of DR'; this relationship seems to show a tradeoff between these requirements, which was expected, but not adequately captured in the qualitative analysis of QFD.

6 Conclusions

The gap in translation between SMEs, giving qualitative assessments, and modelers, using physics-based models to perform quantitative analysis, was mitigated by using ROSETTA to aid in constructing a model of the Smart Grid. The quantitative model was able to verify some aspects of the qualitative model, and also highlight some misrepresented/non-intuitive relationships (e.g., the qualitative model might say there is an impact, but the quantitative model shows that whether that impact is a linear or non-linear relation). From the opposite perspective, the qualitative assessments were able to show inaccuracies in the quantitative model. This was accomplished by using ROSETTA as a means of translating between these two approaches. Any discrepancies that arise between the qualitative and quantitative model were further examined in both models in order to increase the accuracy of the quantitative model. Even with performing two iterations, this model was still not able to capture all aspects of the Smart Grid DR and day-ahead prediction. This means that using ROSETTA to aid in model construction should be viewed as an iterative process until the desired level of model fidelity is reached. By using ROSETTA to compare the qualitative assessments and quantitative analysis in an iterative process, it is believed that a model can constructed to a desired level of fidelity earlier in the design process.

7 Future Work

This paper covered the use of ROSETTA as a guide for developing a model of only a subset of the Smart Grid: the DLC aspect of DR with day-ahead load predictions. It will be necessary to further build on this model to incorporate other aspects of the Smart Grid. The first step should be to perform another iteration of the qualitative and quantitative models using insights gained from the second iteration of modeling (e.g., rebound effect). The next logical step is to include the impacts of DS and DG because it is believed they might impact the feasibility of DR. Another important aspect of DR, which was not captured in this model, is the use of VP for a DR program. In order to be able to accurately assess these pricing schemes it will be necessary to increase the agents' sentience. This could be accomplished by incorporating utility functions for individual agents. It is not required that all aspects of the Smart Grid be captured by one model; for instance, load balancing may be done using a separate model (ABM might not be a suitable modeling technique for analyzing load balancing), if there is no relationship between load balancing and other DR programs.

References

1. Borenstein, S., et al.: Dynamic Pricing, Advanced Metering, and De-mand Response in Electricity Markets. Center for the Study of Energy Markets, UC Berkeley (2002), http://escholarship.org/uc/item/11w8d6m4 (retrieved)
2. Duncan, S., et al.: An Assessment of ROSETTA for Smart Electricity Grid System-of-Systems Design. In: 6th Annual System of Systems Engineering Conference, Albuquerque, NM (2011)
3. Federal Energy Regulatory Commission. FERC: Industries - Reports on Demand Response & Advanced Metering (2012), http://www.ferc.gov/industries/electric/indus-act/demand-response/dem-res-adv-metering.asp (web accessed April 18, 2012)
4. GE Digital Energy: Smart Grid Distribution (2010), GE Digital Energy. General Electric Company, http://www.gedigitalenergy.com/smartgrid_distribution.htm (web accessed April 18, 2012)
5. Incropera, F.P., DeWitt, D.P.: Fundamentals of Heat and Mass Transfer. J. Wiley (2002) (print)
6. Lewe, J.H., et al.: Agent-Based Modeling for Smart Grid: Application to Consumer Reaction to Demand Response. In: 22nd Annual INCOSE International Symposium (2012)
7. Mavris, D., Griendling, K.: The Relational Oriented Systems Engineer-ing Technology Tradeoff (ROSETTA) Environment. In: 6th Annual System of Systems Engineering Conference, Albuquerque, New Mexico (2011)

8. Olesen, B.W.: Thermal Comfort. Technical Review 2, 3–41 (1982) (print)
9. Palmborg, C.: Social habits and energy consumption in single-family homes. Energy 11(7), 643–650 (1986)
10. Ricci, A., et al.: Power-Grid Load Balancing by Using Smart Home Appliances. In: ICCE 2008. Digest of Technical Papers, pp. 1–2, 9–13 (2008)
11. Siddiqui, O.: The Green Grid: Energy Savings and Carbon Emissions Reductions Enabled by a Smart Grid. Tech. no. 1016905. Electric Power Research Institute, Palo Alto (2008) (print)
12. Smartgrid.gov "Smartgrid.gov: What is the Smart Grid?", http://www.smartgrid.gov/the_smart_grid (accessed November 2, 2011)
13. US Department of Energy. Benefits of Demand Response in Electricity Markets and Recommendations for Achieving Them: A report to the United States Congress Pursuant to Section 1252 of the Energy Policy Act of 2005 (2006)
14. Wooldridge, M., Jennings, N.R.: Intelligent Agents: Theory and Practice. The Knowledge Engineering Review, 10(2), 115–152 (1995)

Chapter 13
Prototyping Systems Thinking Curriculum Development for Pre-college Students

Ben R. Jurewicz

Abstract. This paper describes the results of the first 5 years effort to develop a new focus and process for creating an awareness of systems thinking methodology in pre-college students. The effort was undertaken as part of the University of Texas Pre-Freshman Engineering (PREP) program based in San Antonio, TX. The PREP program conducts a 6 week summer classroom environment on local university campuses for Texas Middle School (7-8-9 grades) students focusing on the basics of Science, Technology, Engineering & Math (STEM.) Our particular effort was to add a fourth year known as PREP IV. PREP IV served as a prototyping laboratory to experiment with new curricula for introducing the systems thinking mindset to pre-college students.

Several approaches, such as Community Based Projects, Student Centered Learning, Activity Based Learning, and Challenge Based Learning were incorporated into the curriculum developed. As a result of the exploratory prototyping fundamental understandings of student basic needs were uncovered including: (1) creating a systems thinking awareness or mindset in the students; (2) developing a teachable approach to grasping the concept of what is a system; (3) learning how to do the abstraction required for modeling a system; and (4) analyzing the behavior of a system from a systems model. Most of these basics of system thinking are not intuitive but require innovative development approaches. This paper discusses teaching approaches that address those new understandings. Specifically, it includes the approach of tightly integrating the study of system thinking concepts with the learning exploration of a local community watershed as a system. System definition, modeling, and behavior analysis in a parallel top down theory and bottom up approach results in system thinking mindset retention that was measured by pre and post testing of students. In addition, the basics of the systems engineering process were built into the curriculum providing a process framework to support the systems concept mindset and awareness.

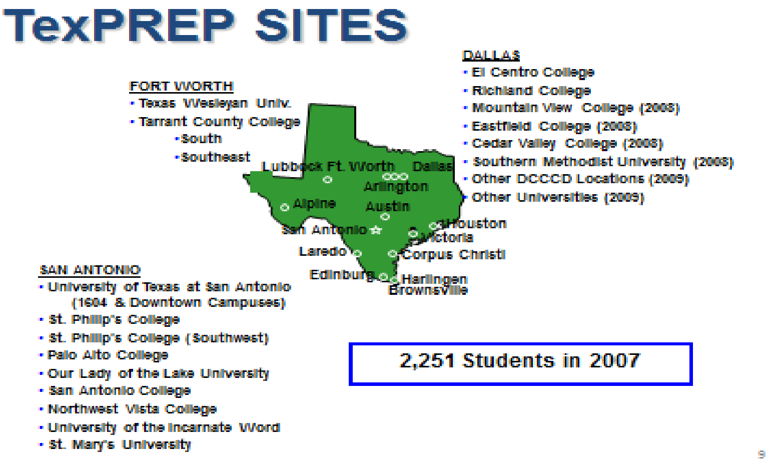
Ben R. Jurewicz
St Mary's University
One Camino Santa Maria
San Antonio, Texas 78228
United States
e-mail: meb@world-net.net

A high level watershed system model was built using the STELLA computer model. This model was constructed from the ground up in parallel with basic water science teaching of: the water cycle, rainfall rates, rain accumulation, watershed area determination (using U.S. Geological Service or USGS topographic maps and Google Earth), soil analysis, evaporation, and river water flow fundamentals. The model was calibrated using an historical rainfall event along with measured stream water flow rates at local stream USGS monitoring stations. Once calibrated the model was exercised to discover the benefits of reduced flows and flooding associated with creating water retention dams. Soil and surface runoff conditions were varied to develop a student understanding of increases in community land development on the potential for flooding. The model and graphical outputs are discussed in the paper.

As a result of the PREP IV experience the students could clearly articulate an understanding of a very important community system. They showed this in a formal presentation to local senior water system executives, water science university professors, and parents. Continuing PREP IV classes are extending the systematic discovery of how to develop curricula that creates sustainable systems mindset awareness. Areas other than watershed systems, such as aerospace, energy, environment, and medicine are also being introduced into the PREP IV systems curriculum.

1 Introduction

Our modern society is generating an increasing complexity in products, processes, and organizations. The possible adverse impacts of uncertain events, misjudgment in reaction and control, and interrelated unforeseen reactions necessitate an increased understanding of systems. While significant effort has gone into developing an understanding of systems theory there is little widespread appreciation of the concept with the vast majority of people. Systems Thinking, Systems Engineering, and Systems Management courses are offered at the college level and for professionals. There are a number of pre-college efforts to move the concept of systems thinking to younger students. The TexPREP program originated at the University of Texas in San Antonio about 30 years ago. It is a 7 week summer program that has focused on creating an interest in Science, Technology, Engineering, and Math (STEM) careers middle and high school students, primarily but not exclusively minority students. Over the last 5 years the TexPREP program has served as a prototyping facility for innovative approaches to developing systems thinking in pre-college students. In particular, the program created a separate 4th year called PREP IV which was expected to be a learning laboratory for developing approaches to teaching systems thinking. This paper discusses insights that have been discovered in the process of teaching systems and provides a community-based example of how the TexPREP students were able to use their newly developed understanding of the systems thinking approach.



TexPREP SITES

FORT WORTH
- Texas Wesleyan Univ.
- Tarrant County College
  - South
  - Southeast

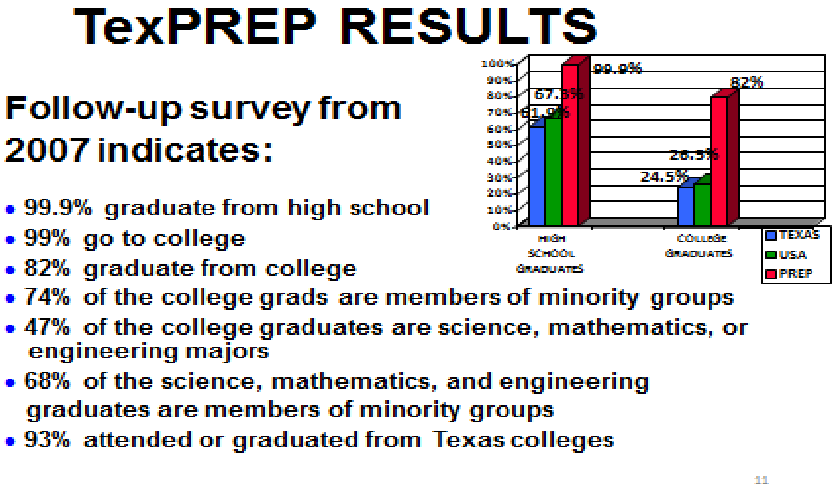
DALLAS
- El Centro College
- Richland College
- Mountain View College (2008)
- Eastfield College (2008)
- Cedar Valley College (2008)
- Southern Methodist University (2008)
- Other DCCCD Locations (2009)
- Other Universities (2009)

SAN ANTONIO
- University of Texas at San Antonio (1604 & Downtown Campuses)
- St. Phillip's College
- St. Phillip's College (Southwest)
- Palo Alto College
- Our Lady of the Lake University
- San Antonio College
- Northwest Vista College
- University of the Incarnate Word
- St. Mary's University

2,251 Students in 2007

2 PREP Program

The TexPREP program location sites and summary data are shown in the following two figures. Students in the program attend daily classes at local universities.



TexPREP RESULTS

Follow-up survey from 2007 indicates:

- 99.9% graduate from high school
- 99% go to college
- 82% graduate from college
- 74% of the college grads are members of minority groups
- 47% of the college graduates are science, mathematics, or engineering majors
- 68% of the science, mathematics, and engineering graduates are members of minority groups
- 93% attended or graduated from Texas colleges

3 The Systems Approach

Much has been written on the systems approach as a dynamic thinking process, an engineering methodology, or as a project management concept. The approach we used with our PREP IV students was systems as a problem solving method for certain types of problems. These types of problems might not have an obvious or easy answer. In fact, many times the existence of this type of problem is not even known until one explores a situation using the systems approach. Our approach emphasized 3 very basic system thinking ideas: awareness of the existence of a specific system; abstraction of selected system characteristics as the basis of modeling a system; and imagination of future states as the foundation for systems analysis. Exercising these core concepts, well, is the necessary foundation for effective and enduring system thinking, systems engineering and systems management approaches in all types of problem solving. Moreover, we were convinced at the outset that no systems thinking learning would be effective or enduring without the very close integration to a relevant practical community based problem.

4 Teaching the Systems Approach

The PREP IV program consisted of 3 subject class periods a day and a study period. One class was systems problem solving theory, another was water science and the third was computer programming basics. The water science class was chosen since our community based problem was related to current activity by local universities and water authorities who are working on understanding the flood potential of the local Salado Creek Watershed. The general concept was to integrate the classroom activities. Systems learning involved understanding system theory. To support this we needed to relate to a specific area, hence the Salado Creek Watershed. Also, the activity in systems problem solving requires using a software program and for these early efforts we chose to use the STELLA systems modeling computer program Classes were comprised of from 20 to 25 students.

The first year's class in 2007 had both the systems theory and water science analysis as part of one class period. It proved to be too much to cover in too short of a time. Our second attempt was to have parallel systems and waters science classes with a convergent focus at the last three weeks. This worked somewhat better, but was not completely satisfactory. Our third attempt had parallel classes that were tightly integrated from the start and all the way through the 6 weeks. Up to this time the computer classes were somewhat separate from the systems and waters science classes. We are currently exploring integrating the computer science class with a systems and an application science class.

Teaching systems theory to students even with concrete examples has not proven effective in instilling a strong awareness of what is a system. Thus, our approach has been to develop system theory through using the application system. Systems classes alone have not been very effective at developing the ability to create an abstract behavior model from just the concept of a system.

System behavior modeling requires a high degree of abstraction. To a degree this can be learned by most anyone with practice and repetition, but some students have a higher degree capability naturally to abstract. Our approach has been to use a very close analogy of the actual system as an introduction, first in the systems theory class. Specifically, we found the bathtub stock and flow analogy was very effective as an introductory system model for a watershed. Using a model like STELLA clearly enhances this process for those students with strong abstraction capabilities as well as those without the capability.

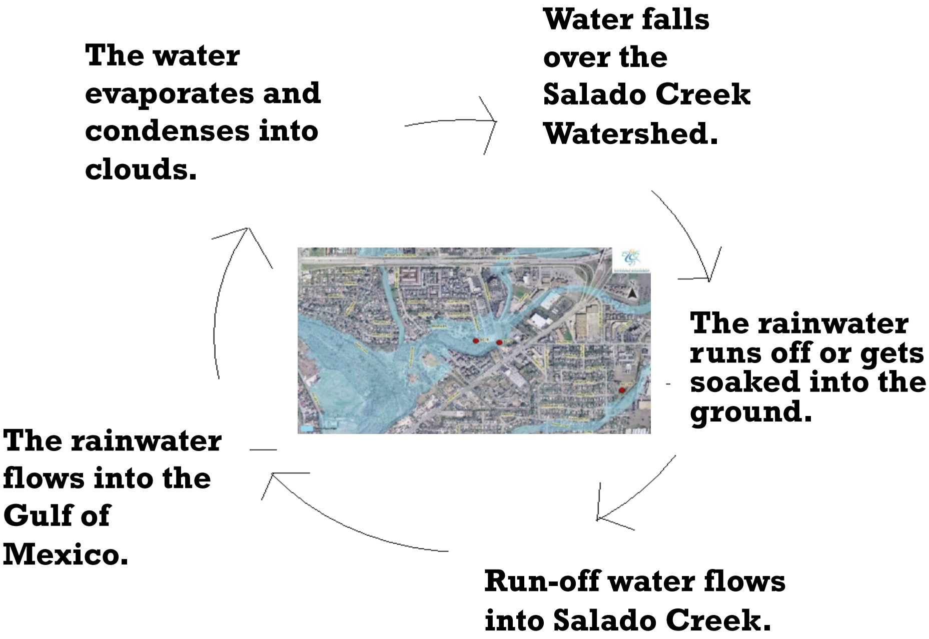
We found that the systems analysis, using a system behavior model, requires time spent understanding the problem context and a good deal of class time devoted to exploring "what-if" scenarios. This analysis process is enhanced by practicing imagination or visioning with various artifacts. We created artifacts, such as the picture in the figure below to see things in different ways to help students learn this process.

Each of the three areas (defining a system, modeling a system's behavior, and analyzing a system's behavior) requires continued development. Nonetheless, we feel that we have made headway as described in the following example of student's results.

Source: Malcolm Lancaster,M.D. (Used to train medical students in interpreting electrocardiograph outputs.)

5 Defining a System

The water science class initially explored the water cycle as a large system. A watershed was then defined as an element of this much larger system as indicated in the figure. Initially, systems can be conceptually defined by a circle with a set of elements inside and outside. As a conceptual diagram this aids students to understand that a system has a boundary, defined by the circle. Inside are the major elements of the system and outside are the other systems that interact with our system of interest. In pure systems theory the major inside circle elements are defined and relations established. The external systems interacting with our system of interest are also defined. Focusing on the watershed system as part of the larger water cycle gets this across more clearly than circle diagrams although they have an initial role.



Water falls over the Salado Creek Watershed.

The rainwater runs off or gets soaked into the ground.

Run-off water flows into Salado Creek.

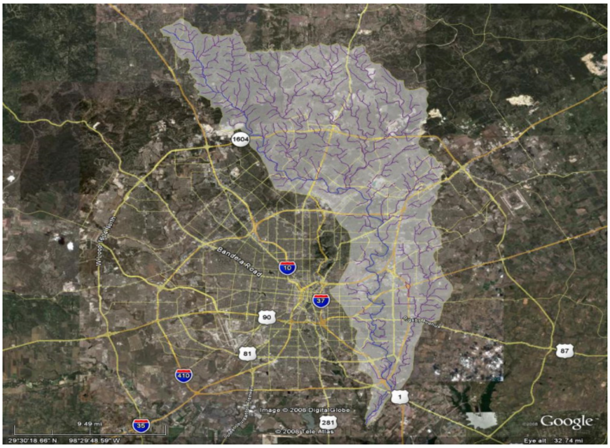
The rainwater flows into the Gulf of Mexico.

The water evaporates and condenses into clouds.

Salado Creek Watershed - Part of a Much Larger Water Cycle System

6 The Salado Creek Watershed System in San Antonio, TX

The Salado Creek Watershed (SCW) was studied very intensively. U.S. Geological Survey (USGS) data delineating the actual SCW boundary were used to overlay the SCW boundary on a Google Earth map of the San Antonio area. The Salado Creek and all tributary streams were identified as shown below.



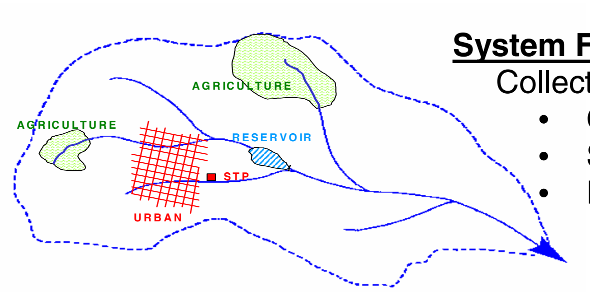
Defining the Salado Creek Watershed System

7 Modeling the Salado Creek Watershed

For our purposes the students were focused on understanding the flow of water through the SCW associated with an Input rainfall event. Behavior of the SCW system was focused on the water flow activity leading to the SCW system Output of stream flow. How rainfall of a specified intensity and duration was converted to a level of stream flow was the system behavior desired. This required the development of a Stock and Flow model. Understanding of the sequential and parallel set of causes and effects during a rainfall event in a watershed was clearly discussed as part of the science class activity. An understanding of the physical processes that controlled the transport and accumulation of water was explored with hands on measurements and activities. This provided the basis of selecting a network of Stocks and Flows that connected the input rainfall to the output stream flow. Early efforts provided a fully developed model to the students who then used it to explore alternative scenarios. The more recent efforts build the SCW systems model from single Stocks and Flows and allow them to participate in full development of the SCW model. In the process surface topography, soil topology, evaporation, and other phenomena are explored. This was accompanied with a field trip and computational exercises.

Input:
Accumulated Daily (in or cm)
Rainfall Rate (in/hr or cm/hr)

Output:
Streamflow (cfm or m^3/sec)



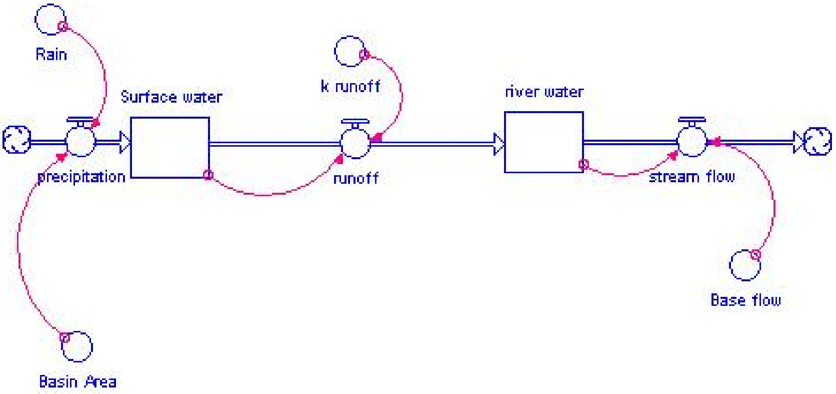
System Function of Interest:
Collect & Drain Water
- Ground Seepage
- Surface Runoff
- Evaporation

Watershed Characteristics Modeled:
- Surface Topology
- Land Usage Patterns
- Soil Classification
- Basin Retention

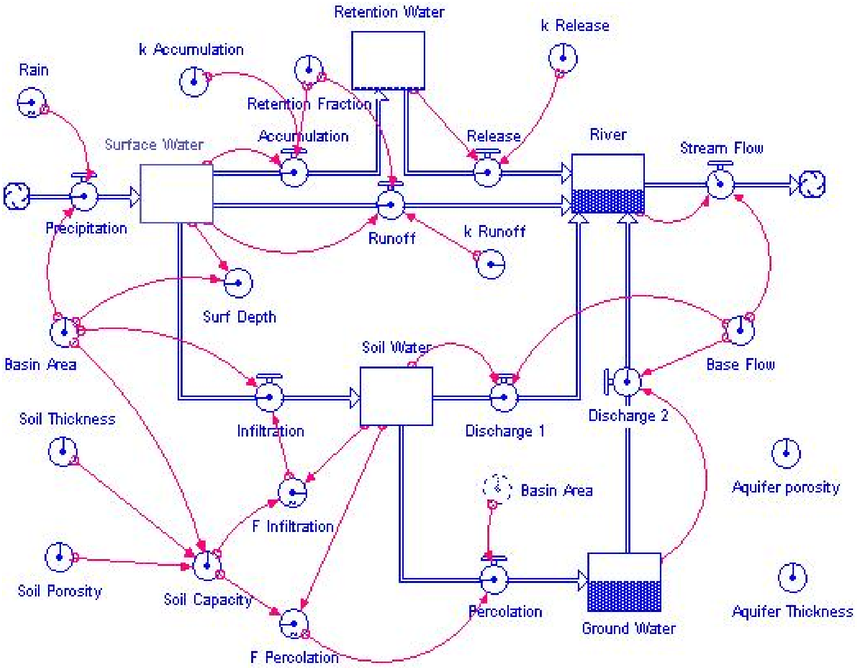
Behavioral Watershed Characteristics of Modeling Interest

8 The Salado Creek Watershed Model

The student developed SCW models from the starting version to the final version are show. After exposure to the bathtub systems flow model, the initial construct for a watershed model was developed with the students as shown below. From this simple stock and flow model the more detailed model was constructed.



Initial Watershed STELLA Model

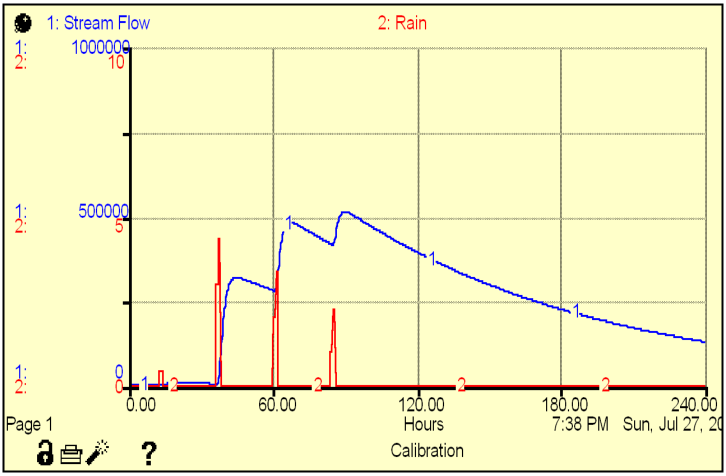


Final STELLA Watershed Model

9 Systems Engineering Approach

We used real world data from Salado Creek station 13 to test our watershed system in the Stella program. In June and July, Salado Creek experienced a lot of stream flow. It had rained on June 16th, 20th, and 28th. Each one was above 1 inch with the 28th being 3.07 inches. After that rain, the peak flow in cubic feet per second was almost 4000. The changes in the graph were similar to the actual data, but less dramatic. The graph in the Stella program is almost as accurate as the actual data from the Salado Creek station 13, but it doesn't show the little changes that happened between the days. The real data showed more changes among the days with more increases and decreases around the June 16 and June 23 portion of the data. The computer model cannot account for all of the complexities of the Salado Creek watershed.

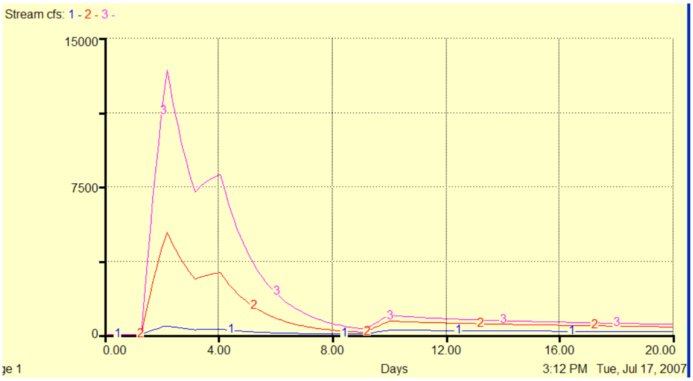
Calibrating the Model (With Real Time data)



Varying Rainfall Accumulation Levels

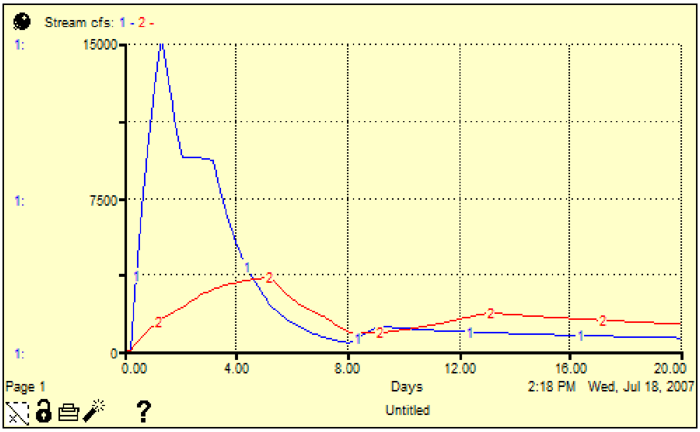
This graph represents the number of cubic feet per second (CFS) given different scenarios of rainfall. The y-axis (vertical line) of the graph represents the CFS, and the x-axis (horizontal) of the graph represents the days. The rainfall totals used in this scenario were one, five, and ten inches. The blue represents what would occur if it rained 1 inch in one day. The CFS for one inch for one day was about 2,000. It doesn't show any significant change in the CFS, because it was a small rain event. The red line represents what would occur to the CFS at five inches in one day. The CFS at five inches in one day was about a little over 5,000. The reason is because this was a somewhat considerable rainfall event. There is somewhat a significant change. In comparison with 1 inch in one day, the CFS was greatly altered. The pink line represents ten inches of rain falling within only one day, which means this was a huge rainfall event. The CFS for ten inches in one day can be estimated by looking at the graph to be 13,000. In comparison to five inches one day, the CFS for 10 inches greatly surpasses it. After the 8th day, the CFS for each scenario evens out. In conclusion, the greater the rainfall in inches the greater was the CFS.

Rainfall Accumulation Different Rain Events

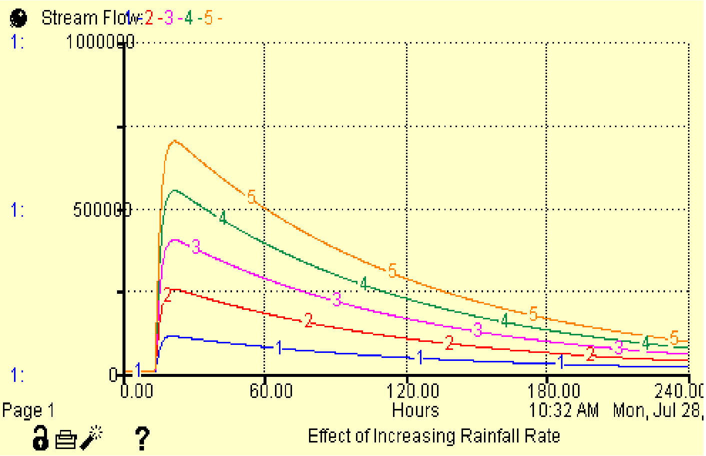


This graph compares the events of 2 inches per day for five days and 10 inches of rainfall in one day. The stream discharge in cubic feet per second (cfs) for the event where it rained 10 inches in one day is higher than the stream discharge cfs for the event where it rained 2 inches per day for five days. The graph for the event of 10 inches in one day decreases drastically after the first couple of days. This is because it stops raining after the first day, so the discharge is lower. The graph for the event of 2 inches per day for five days increases after the first day because it continues to rain for the next four days. These two graphs are dramatically different even though the same amount of rainfall fell during the five days. It is because the rate at which the rain fell was different in both cases, so the graphs would also be different. Since more rain fell in a shorter amount of time, the discharge is higher than when it rained at a constant rate for 2 inches per day for five days. The conclusion is, the discharge in cubic feet per second is higher during situations where more rain falls in a condensed amount of time than at a constant rate over a long period of time.

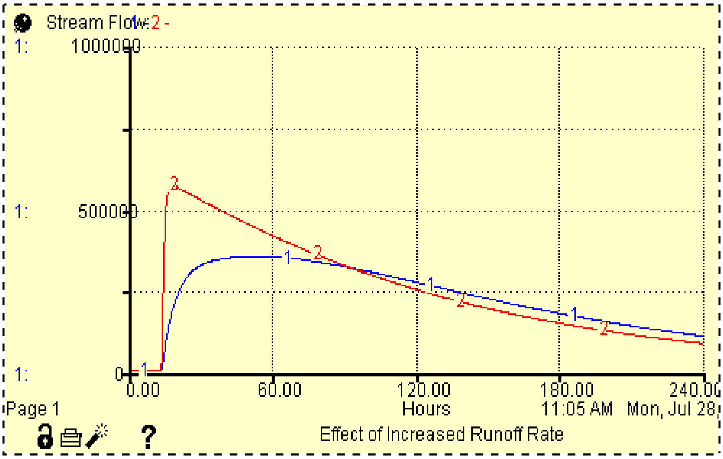
Same Total Rainfall Accumulation



Effect of Increasing Rainfall Rate (10% Retention)
Increasing Rainfall Rate (10% Retention)



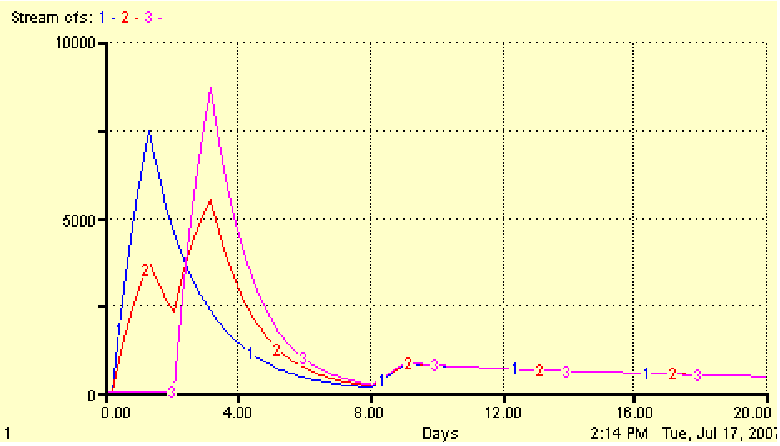
Effect of Increased Runoff Rate
(5 inches of Rain in 1 hour and 10% Retention)



Varying the Retention Quantity in Dams

This graph deals with showing the (impact of) fraction retained (in dams). The graph shows that the highest amount of retention led to the highest amount of stream flow. At a 0% retention rate and 5 inches of rainfall, the cfs level of the stream is shown in 1. At 50% retention rate and with 5 inches of rainfall, the cfs level of the stream is shown in 2. Then finally at 100% retention rate and 5 inches of rainfall, the cfs level of the stream is shown in 3. This graph shows that it just delayed the problem and didn't fix it. The best one was 50% because it fixed the problem.

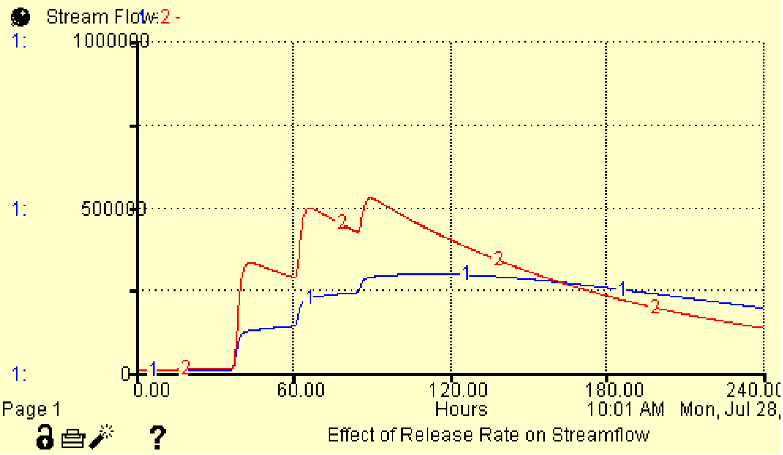
Rainfall Retention Variation



Effect of Release Rate on Stream Flow

This graph represents what happens to the water in the stream as water from a retention dam is slowly released into it. These curves in the graph are the amount of water in the stream after a 3-day period. You have 3 different cases. In the first case water is retained for one day and then released back into the stream on the first day. In the second case water is retained for two days and then released back into the stream on the second day. On the final case study water is retained for three days and then released back into the stream on the third day. As you can see, retention dams help for quick water control but when you put the water back into the stream it rises again so engineers have to be careful they do not flood the stream when they release the water.

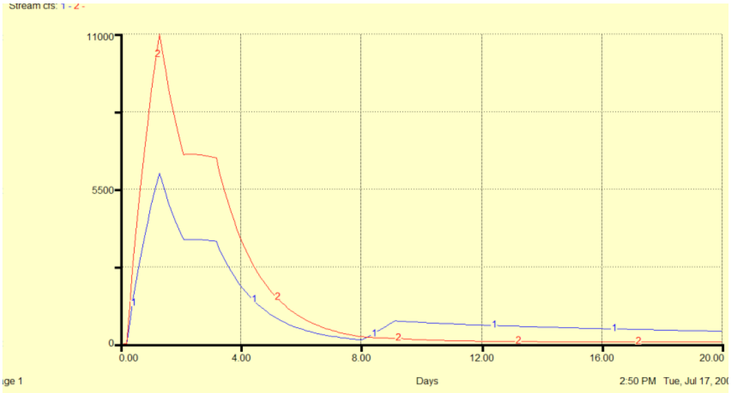
Release Rates



Effect of Complete Urbanization

The graph shows that as the percent of urban land use increases there is more water in the stream flow. This shows that if the whole area of Salado Creek were to become urban, then there would be more water and also more flooding. The red line represents the total amount of water there would be in Salado Creek if all the land were urban. The blue line represents the result with the actual present day percentages of land that is urban, suburban, forest, and farm. This scenario was set so that Salado Creek would receive five inches of rainfall. Also, to show the difference that would occur if all the Salado Creek would become an urban area instead of the type of area it is formed of, the percent of urban land area was changed to 100 percent. The other land areas such as: forest, farm, and suburban areas were all set to 0 percent. The final result was that if Salado Creek were only urban (all pavement, no plants, no absorbent surfaces), then the amount of water that would flow would increase significantly. The amount of water flowing in the current condition is half of the amount flowing in the other graph that shows all the land being urban. The amount of water measured in the graph is measured in cubic feet per second.

Urbanization



10 Summary

The PREP IV program in Texas is a learning environment for developing new methods to teach the systems way of thinking to young students, i.e., pre-college. We think of it as a Systems Education Environmental Development (SEED) laboratory. Prototyping new approaches to teaching systems thinking was our objective and the efforts were assessed to see how well they work. Our measurements, thus far, by pre and post testing indicated significant statistically-based improvement in student understanding of systems.

Our approach was based on teaching the systems concept through integrating systems theory with a community problem application. We used general systems concepts together with a water science focus. The water science focus was specifically on rainfall in a watershed and the nature of its contribution to the level of stream water flow. The community service oriented focus was used to make the abstract systems concepts relevant, understandable, and interesting.

The teaching approach started with an initial concept, followed by a discussion of basic examples, and then a hands-on activity was used to firm up the concepts. This “think-talk-do” approach evolved over several years. Our feeling is that it should be a spiraling process repeated over and over as new systems concepts are introduced. It has become clear that there are three major concepts that need to be embedded in young students to truly develop a systems thinking mindset. These are: (1) a deep awareness of the system concept so that students can isolate and define a relevant system from a situation; (2) an abstraction capability to select the relevant characteristics from the system to make a representative model of system behavior; and (3) an exploratory imagination to formulate scenarios to analyze with the system’s representative model. We found that if we could develop these three concepts in students, then the student is truly capable of serious systems thinking.

The Salado Creek Watershed (SCW) focus and the resulting student analyses shown in the graphs of this report indicate a clearly developed process of systems thinking, even a systems engineering mindset. A positively received presentation of these SCW results to a large group of community professionals affirmed that the students had grasped the ideas of systems thinking and would be well prepared to enter college in Science, Technology, Engineering, and Mathematics (STEM) fields.

Acknowledgement. This material is based upon work supported by, or in part by, the U.S. Army Research Laboratory and the U.S. Army Research Office under grant number W911NF -11-1-0150.

Chapter 14
An Integrated Approach to Developing Automotive Climate Control Systems

Guillaume Belloncle, Patrick Chombart, and Bernard Clark

Abstract. The development of embedded systems for complex cyber-physical products involves multiple processes and disciplines – from project management and requirements engineering, to configuration, integration, simulation, test and verification management. Traditional model based systems engineering approaches, where generated models and simulations are largely isolated from one another, make it almost impossible to get a 'holistic' view of a complete product or systems behavior.

While most frameworks for model-based design support a single discipline, what is actually needed is a framework that can handle the multi-disciplinary architecture and systems integration of the complete product or system. There is a need for standards to enable the combination of cross-discipline design efforts in a common environment that fully supports modeling, simulation and embedded software generation and validation.

This paper outlines an integrated approach where the Requirement, Functional, Logical, and Physical (RFLP) decomposition of complex cyber-physical products is achieved in a fully integrated 2D / 3D collaborative systems development environment. We will outline how the Modelica[1] language, in conjunction with the open 'Functional Mockup[2]' and AUTOSAR[3] concepts, can be leveraged by a

Guillaume Belloncle · Patrick Chombart
Dassault Systemes
10, Rue Marcel Dassault
78140 Vélizy-Villacoublay
France
e-mail: {Guillaume.Belloncle,Patrick.Chombart}@3ds.com

Bernard Clark
Dassault Systèmes UK Limited
Suite 9
Riley Court
Milburn Hill Road
CV4 7HP
Coventry
United Kingdom
e-mail: Bernard.Clark@3ds.com

Product Lifecycle Management (PLM) based systems integration platform, to define and evaluate the functional definition of a complete product or system. We will illustrate this approach through an example of an automotive climate control system development process.

1 The Systems Engineering Challenge

Systems engineering is not new in the automotive industry. Most leading automotive companies have formalized the discipline in one form or another into their new product development process. However, many organizations have not derived all the benefits that they expect because of the poor collaboration and orchestration of their business processes. There is a lack of control in managing data and model consistency across a large number of systems engineering tools – particularly in the context of highly configured products that are developed across an extended enterprise.

Another source of frustration is the fact that existing systems engineering processes are largely document driven, and that systems are still verified and validated late in the overall development process through physical prototypes. The premise of the classical systems engineering 'V'-based systems development model is that every stage of the development is tested, verified, and validated. In spite of this premise, the effectiveness of the V model has been limited by the lack of integration between the tools that are employed at each stage of the process.

It is not uncommon for organizations to literally use hundreds of different tools at various stages of the systems development process. These tools are all aligned by the individual needs of the different engineering disciplines, and exemplified by the different digital engineering-based models, which have limited or no connectivity to one another.

The crux of this problem stems from the classical approach to defining and developing systems. Systems are usually initially defined with a standalone requirements-management tool that has loose integrations to a systems architecture tool, which is typically done with a combination of UML & SysML (or even Visio) based tools. In the context of cyber-physical products however, these architecture modeling tools have limited value. While they are excellent for the high-level systems architecture definition and detailed design of a given software module, they are limited by their inability to be well integrated into the overall product development process. Typically, using standalone tools results in:

- Limited **integration and traceability** capabilities exist between the high-level product requirements definition through to the decomposed functional, logical and discipline specific architecture models, and then through to the instantiation and simulation of these models in a 3D-based virtual product definition.
- No ability to **share the systems architecture** with the different engineering domains in a unified way. This is due to the traditional "models", which are expected by the engineering teams to be normal 2D schematics of the electrical, hydraulic, pneumatic, power management systems, etc.; these are different and 'disconnected' from the traditional SysML-based diagrams that the systems architect uses.

- No ability to have **configuration management** of the systems architecture at a very granular level and no integration between the embedded software development process and the physical (virtual 3D-based) product modeling environments. As a result, it is not possible to receive inputs from the physical dependencies of the 3D model, or to simulate the embedded software within the context of the virtual product.
- **Limited coverage of the entire systems & product development lifecycle**, as these systems development tools do not cover all the engineering activities needed to develop the complete product. I.e. Poor code generation, poor support for the integration and validation of the software with the physical product under development.

Models and data are often tool specific and evolve over time, making it almost impossible to manage their consistency without a major change in the way organizations manage their overall systems engineering development process.

Producers of complex cyber-physical products are demanding a more unified and integrated approach to systems engineering. They need a system's engineering development platform as well as tools that enable them to quickly and easily define and navigate the complex relationships that exist between the many different entities that make up the complete product, with all of its embedded systems.

1.1 The Need for an Integrated Systems Engineering Development Platform

Stand-alone modeling approaches to systems modeling and simulation are no longer adequate for addressing the combined pressures of reducing product lead times, improving responsiveness, and accommodating the greater levels of complexity introduced with software. Models must contribute to the validation of the entire system, not just the individual parts or sub-systems.

Innovations in multi-physics analysis are enabling designers to combine the simulation of multiple physical properties in a unique simulation. However, increasingly complex automotive designs, such as hybrid vehicles, are triggering the need to chain together models of multiple parts, sub-systems, or systems in order to reliably represent complete vehicle operating scenarios. For example, there are different patterns of interplay between internal combustion engines, electric motors, braking, energy recovery, and battery charging and discharging in hybrid vehicles that must be modeled in conjunction with one another under different acceleration, deceleration, cruising, and braking conditions.

What is needed is a systemic view, as depicted in figure 1, that sees models not only as individual sub-optimizations, but also as product-level exercises that optimize the design of the whole vehicle. This approach is critical to the vision of delivering the "3D digital product experience" with the goal of embedding intelligence in 3D based simulations. Additionally, to improve the relevance of simulations, environment models are increasingly required that include the 'driver-in-the-loop' behavior, as well as the related operating scenarios.

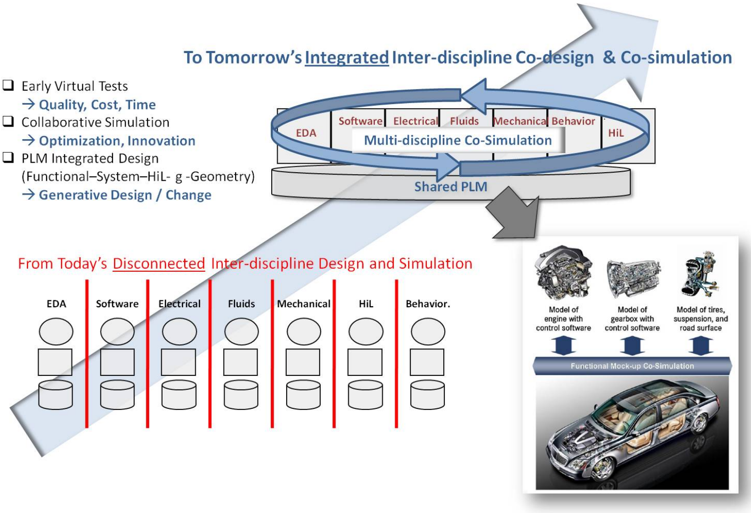

To Tomorrow's Integrated Inter-discipline Co-design & Co-simulation
❑ Early Virtual Tests
→ Quality, Cost, Time
❑ Collaborative Simulation
→ Optimization, Innovation
❑ PLM Integrated Design
(Functional–System–HiL- g -Geometry)
→ Generative Design / Change
EDA
Software
Electrical
Fluids
Mechanica
Behavior
HiL
Multi-discipline Co-Simulation
Shared PLM
From Today's Disconnected Inter-discipline Design and Simulation
EDA
Software
Electrical
Fluids
Mechanical
HiL
Behavior.
Model of engine with control software
Model of gearbox with control software
Model of tires, suspension, and road surface
Functional Mock-up Co-Simulation

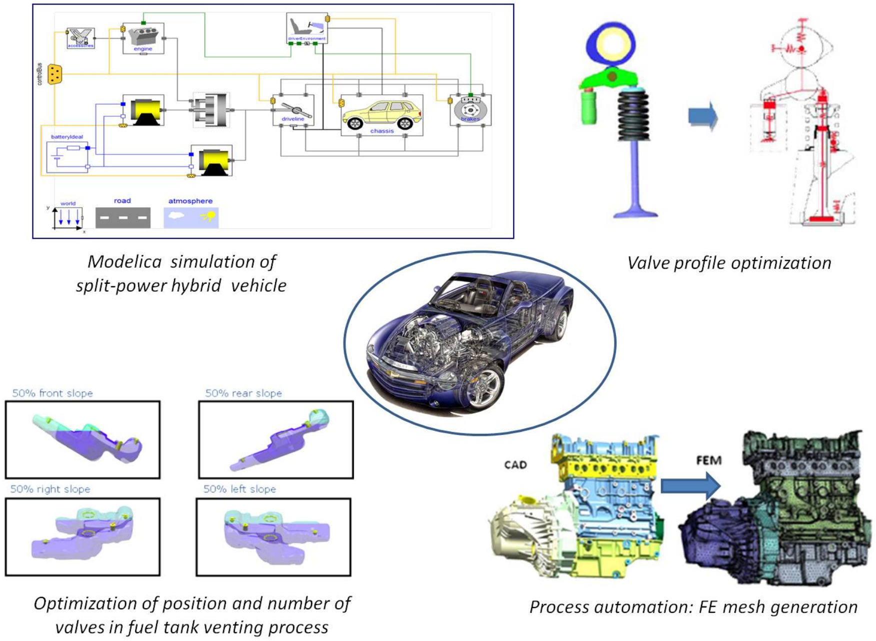
Fig. 1 The need to move to an integrated cross-disciple co-design & co-simulation environment

1.2 The Need for Multi-disciplinary Design Optimization

An ever-increasing drive to improve performance, reduce costs, and increase efficiencies associated with complex system development has led to the need to explore computational methodologies that enable the development of better systems in less time with higher quality and reliability. This impetus has been particularly visible in industries where the complexity and multidisciplinary aspect of systems can lead the design team to challenging problems involving conflicting requirements that do not appear to have an optimum solution space.

If we consider an automotive example, where the performance of antilock braking systems provides a good case in point, braking distance can be shortened by increasing the size of the tires; however, bigger tires may in turn penalize fuel economy, increase vehicle weight, increase road rolling resistance, impact vehicle aerodynamics and ultimately penalize fuel consumption. In turn, such adjustments may also dictate changes in embedded software logic. Two of the most important computational methodologies required are multidisciplinary design optimization (MDO) and multi-physics simulation. Figure 2 graphically shows some automotive examples of multi-disciplinary design optimization scenarios.

Multidisciplinary design optimization is a field of engineering that uses optimization methods to solve design problems incorporating a number of engineering disciplines simultaneously. Although including all disciplines simultaneously significantly increases the complexity of the engineering design problem, the optimum of the simultaneous problem is far superior to the design found by optimizing each discipline sequentially, since it can account for interactions between the disciplines.



Modelica simulation of split-power hybrid vehicle

Valve profile optimization

Optimization of position and number of valves in fuel tank venting process

Process automation: FE mesh generation

Fig. 2 Automotive examples of multi-disciplinary design optimization

The Dassault Systèmes's (DS) MDO products are built on Isight technology, which is a software framework that replaces the manual trial and error portion of the traditional design optimization process with an automated, iterative procedure. This technology loosely couples all of the relevant modeling codes then automatically runs these codes, evaluates the output, adjusts the input based on defined objectives, and re-runs the codes, continuing with this process until the design objectives are satisfied. See figure 3.

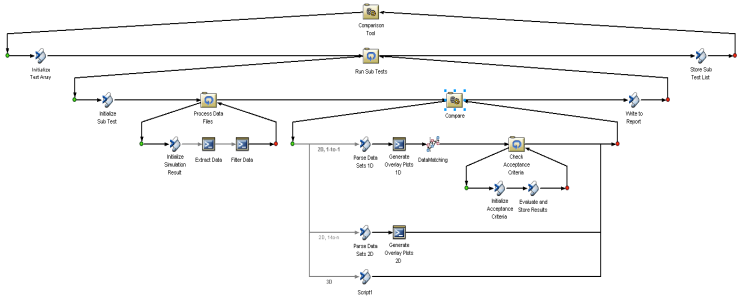


Fig. 3 Typical "System of Systems" Multi-discipline Design Optimization (MDO) flow represented in Isight

The deployment of MDO can be very effective in systems-level design as a bridge between disciplines and subsystems. It can also be used as an optimization tool for exploration of design solutions when coupled with higher fidelity computer-aided engineering tools (i.e., finite element analysis in structural design or computational fluid dynamics in aerodynamics) and multi-physics tools involving coupling of multiple, high-level design disciplines (i.e., fluid-structure interaction problems or software coupled with electromechanical components).

1.3 The Role of Standards – Modelica, Functional Mockup Interface and AUTOSAR

Standards are pivotal for integrating modeling and providing end-to-end traceability. The new Modelica, FMI / MODELISAR and AUTOSAR standards are opening new possibilities, not only to integrate the automotive modeling processes, but also to provide new, flexible capabilities that can help automotive companies develop the mixed-domain models that are increasingly required by today's vehicles.

These standards will enable OEMs and suppliers to concentrate on developing new functionality rather than designing interfaces. They will boost the cooperation between organizations; not only between OEM’s and their suppliers and partners, but also across disciplines within any given organization.

2 Modelica for Modeling and Simulating Multi-domain Cyber-Physical Systems

Modelica is a relatively new language which emerged in the late 90s. It offers a robust solution to address the needs of industry brought about by the increasing complexity of products and systems, and the need to improve quality and reducing overall time to market of these complex products.

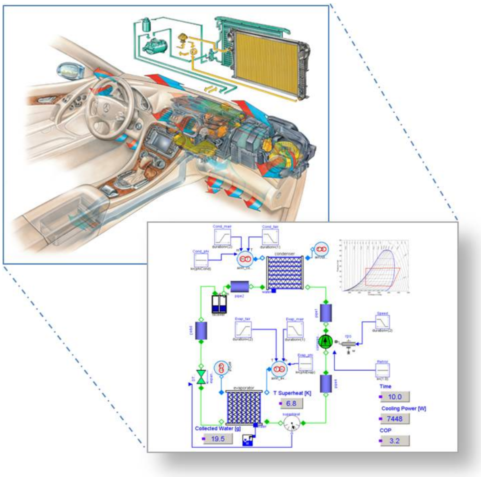


Fig. 4 Modelica Representation of Typical Automotive

Modelica is a declarative, object oriented, open-source modeling language that can support extensions for multiple domains including mechanical, electrical, electronic, hydraulic, thermal, control, electric power, and process-oriented components. Significantly, Modelica can support mixed domains within the same model, a benefit that has become important for automakers that are developing hybrid vehicles. It provides a higher-level alternative to traditional modeling approaches that represents models as data that flow in and out of computational blocks. It also provides a more dynamic alternative to SysML, which provides snapshots of a system state, whereas Modelica is designed to solve difficult system problems, for dynamic interaction giving performance estimates and measurements in particular:

- Multi-discipline problems involving simultaneous technologies from *multiple domains* such as: mechanical, hydraulics, pneumatics, thermodynamics, flow dynamics, electrical, software, real-time, etc.
- Problems where the components are *highly coupled* together, where traditional hierarchical design does not work, or does not readily provide the ability to reach optimal designs
- Problems involving hybrid mathematic solving such as continuous-discrete modeling and simulation

Figure 4 shows diagrammatically a Modelica representation of a typical automotive climate control system. Modelica is defined and managed openly, by the non-profit Modelica association with the objective of delivering a scalable, equation based, dynamic modeling environment that unifies multiple engineering and physics domains. By leveraging investments in component libraries created using the Modelica language, it provides the ability to design, optimize, and check, as early as possible in the design process, the behavior of a planned future product in a virtual environment.

Additionally, the Modelica association is planning to extend the language scopes to encompass additional domains such as HiL, safety and embedded software to provide a comprehensive coverage of the system engineering disciplines as shown in figure 5.

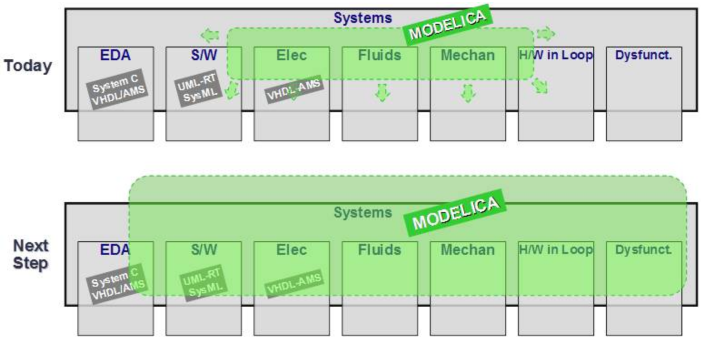


Fig. 5 Evolution of Modelica scope

Thus, Modelica has the potential to become ‘the’ standard for dynamic system modeling in all disciplines.

By using the open source *Modelica* language and its multi-physical solvers, systems engineers can execute and analyze system or sub-systems models while mixing dynamic and state logic behaviors. They can also add control algorithms by linking to third-party models from tools such as MATLAB/Simulink. Modelica makes it possible for users to generate deterministic behavior and reuse modular mechatronics components.

3 Functional Mockup Interface for the Multi Disciplinary Exchange of Models and Their Co-simulation

The concept of a ‘Functional Mockup Interface[2]’ was defined by the European ITEA2 Modelisar project. The main output of this project was the definition of a Functional Mockup Interface[5] (FMI). This FMI definition provides advanced run-time interoperability interfaces that enable accurate model compositions to be created by allowing several pre-compiled simulation units to be combined into one simulation framework.

FMI is an open, general and vendor independent tool interface standard for enabling systems simulation by creating a Functional Mockup[2] as shown in figure 6.

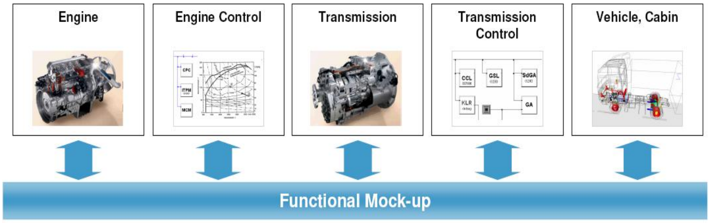


Fig. 6 Functional Mockup for co-simulation & model exchange

The FMI specifications are published under a copyright free license. It includes the definition of four key capabilities for model composition including model interface, co-simulation interface, lifecycle management interface and application interface (including HIL). See figure 7.

The use of FMI offers a number of values and benefits:

1. The FMI definition is tool & modeling language neutral compared to e.g. Simulink from The MathWorks
2. The delivery of Functional Mockup Units (FMU’s) can preserve the providers Intellectual Property (IP) since code is delivered in binary format (documentation and source code can optionally be added to the FMU package)

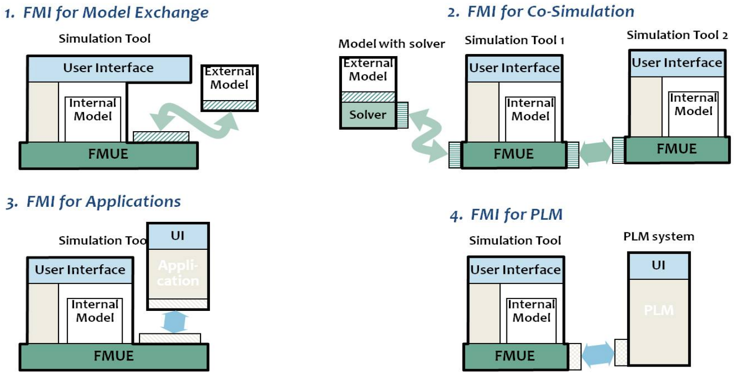


Fig. 7 FMI for model composition, co-simulation, lifecycle management and application interface (including HIL)

3. The export formats generated can be "composed" – manually with very lightweight tools – with models coming from other (non-Modelica) modeling environments. These composition capabilities are expressed diagrammatically in figure 6, in an automotive use case, where often specialized or legacy tools are used to create, model and simulate individual subsystems
4. FMI specifications have been designed with a minimum overhead, to preserve – as much as possible - the capability of real-time execution.

In practice the FMI standard has four layers, and is implemented through a standardized XML description that acts as meta-data to enable the digital composition.

3.1 The Standard for Model Exchange

The intention is that a modeling environment can generate C-code of a dynamic system model that can be utilized by other modeling and simulation environments. Models are described by differential, algebraic and discrete equations with time-, state- and step-events. The models to be treated by this interface can be large for usage in offline or online simulation, or can be used in embedded control systems on micro-processors. It is possible to utilize several instances of a model and to connect models hierarchically together. A model is independent of the target simulator because it does not use a simulator specific header file as in other approaches. A model is distributed in one file called FMU (Functional Mockup Unit).

3.2 The Standard for Co-simulation

The FMI definition provides an interface standard for coupling two or more simulation tools in a co-simulation environment. The data exchange between subsystems is restricted to discrete communication points. In the time between two communication points, the subsystems are solved independently from each other by their individual solver. Master algorithms control the data exchange between subsystems and the synchronization of all slave simulation solvers (slaves). All information about the slaves, which is relevant for the communication in the co-simulation environment is provided in a slave specific XML-file. In particular, this includes a set of capability flags to characterize the ability of the slave to support advanced master algorithms, e.g. the usage of variable communication step sizes, higher order signal extrapolation, or others.

3.3 The Standard for Component Management

The intention is to provide a generic way to handle all FMI related data needed in a simulation of systems within a "Product Lifecycle Management" system. This includes:

- Functional Mockup Unit data, needed for: editing, documenting, simulation and validation;
- Co-simulation data, needed for: editing, simulation, and results management;
- Result data, needed for: post-processing, analysis and reporting.

Generic processes are defined as well as a format description to communicate between the PLM system and the authoring tools.

With the completion of the MODELISAR project in December 2011, FMI has begun picking up critical mass support. Currently, FMI has drawn support in more than 30 tools from 20 third-party vendors. Going forward, FMI will continue to evolve under the auspices of the Modelica association.

4 AUTOSAR for Embedded Software Architecture and Code Generation

AUTOSAR (Automotive Open System Architecture) is an initiative for standardizing the architecture and interfaces of software embedded into automotive ECUs that is being developed by automobile manufacturers, suppliers, and tool developers. The standard, which is now stabilizing with the recent release of version 4.0, is designed around a tiered architecture that includes an application layer that contains all functionality, the interface to the hardware, and the runtime environment.

AUTOSAR frees OEMs of the need to devise their own proprietary systems architectures or spend time writing unique interfaces to different ECUs. By decoupling the application from how it is physically implemented on the device, automakers can focus software development on the embedded applications that define the driving experience and differentiate the car. Furthermore, by providing a standardized architecture, AUTOSAR allows designers the flexibility of taking advantage of the distributed, networked architecture that provides the information backbone within the modern automobile. As the level of software content in the vehicle grows, systems designers must determine how to distribute processing loads across different ECUs, and with AUTOSAR they have the freedom of deployment without the burden of having to port software to different ECU devices. In turn, AUTOSAR frees suppliers from having to worry about writing custom APIs to devices designed by different OEMs.

5 Illustrative Example: Developing an Automotive Climate Control Systems

To illustrate the above points, this example which was developed jointly by Volvo Technology Corporation and Dassault Systèmes as part of the European ITEA2 Modelisar project, illustrates the benefits of using a collaborative PLM systems platform for managing the end-to-end system engineering & embedded software development, based on an automotive climate control system example.

Volvo Technology Corporation delivers climate control systems for cars, trucks and construction equipment. Although climate control systems are technologically mature products, there are still industry challenges to decrease the systems development time while increasing the overall product quality. Typically, these challenges *include*:

- Difficult to work on a “single source of truth”, while maintaining the right level of Intellectual Property (IP) protection for both the OEM and their suppliers - when exchanging requirements, models, data and documents.
- Demonstrating end-to-end traceability, from requirements to detailed hardware and software components, is needed more and more in order to comply with functional safety regulations
- Limited ability to simulate early in the development cycle, the climate control systems behavior while optimizing the controlling parameters in the context of a new vehicle program. This limitation is brought about by the disconnection between the systems architecture definition and the associated software and hardware design tools and processes.

Using industrial data and representative systems engineering workflow processes, a demonstrator has been developed to show how a combined PLM platform and integrated systems engineering approach can address these challenges.

5.1 The Climate Control System

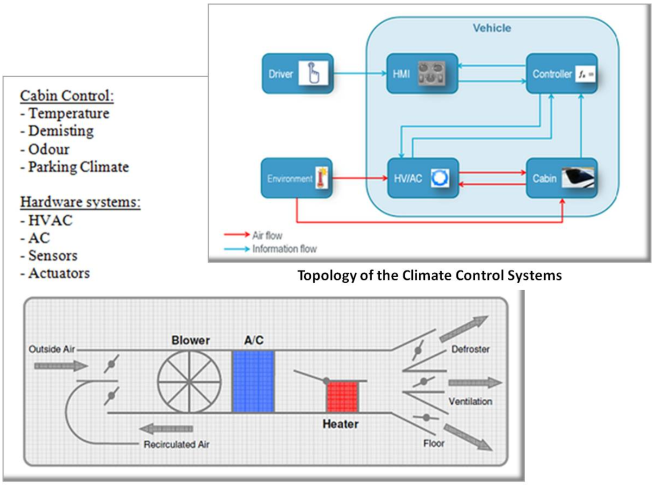
The system being designed in this scenario is shown in figure 8, and is composed of:

A Controller Model: Created by a supplier using MATLAB/Simulink from The MathWorks, and then refined into a FMU component for model exchange and co-simulation;

A Plant Model: The climate control systems components are modeled first with Modelica libraries within the Dassault Systèmes CATIA V6 Dynamic Behavior

Modeling (DBM) environment, and then exported into Functional Mockup Units (FMU's) for model exchange and co-simulation

Systems engineering data and artifact are stored in PLM database leveraging the Dassault Systèmes Requirement / Functional / Logical / Physical (RFLP) development paradigm. This RFLP based integrated data model is stored within a unique central database that facilitates worldwide collaboration across multi-discipline teams.



Topology of the Climate Control Systems

Simplified View of Typical HVAC System

Fig. 8 Topology of Automotive Climate Control System

- The **Requirement** layer is used to capture requirements such as marketing and engineering needs, regulatory requirements, etc. E.g. Minimize blower noise, display inside cabin temperature, automatic fan speed selection, etc., and the associated test cases to verify the climate control system is fulfilling these requirements
- The **Functional** layer is used to perform the functional decomposition of the services the climate control system should perform and structure them with function interfaces, data and control flows (see figure 9).
- Definition of system functions and sub functions; e.g. recycle air, heat cabin, compute required air temperature and flow rate, etc.
- Definition of data exchanged and control flows between sub-functions; e.g. HMI signals, air flow within the cabin, etc.
- Provide traceability via links between functions and requirements; e.g. "compute the temperature increase / decrease required" requirement is implemented by a "temperature sensor" function and a "comparator" function.

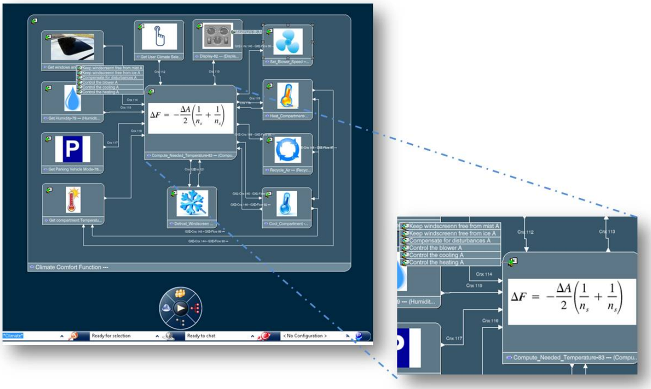

Get windows an...
Keep windscreenn free from mist A
Keep windscreenn free from ice A
Compensate for disturbances A
Control the blower A
Control the cooling A
Control the heating A
Get User Climate Sele...
Display-82 --- (Displa...
Set_Blower_Speed --...
Get Humidity-79 --- (Humidit...
Get Parking Vehicle Mode-78...
Get compartment Temperatu...
Compute_Needed_Temperature-83 --- (Compu...
$\Delta F = -\frac{\Delta A}{2}\left(\frac{1}{n_s} + \frac{1}{n_s}\right)$
Heat_Compartment-...
Recycle_Air --- (Recyc...
Defrost_Windscreen ...
Cool_Compartment -...
Cnx 112
Cnx 113
Cnx 114
Cnx 115
Cnx 116
Cnx 117
Climate Comfort Function ---
Climate
Ready for selection
Ready to chat
< No Configuration >

Fig. 9 Functional Decomposition of Typical Climate Control Systems

- The **Logical** layer represents the implementation of a technical solution for the climate control system matching the functional analysis.
- Definition of the components behavior using Modelica models and libraries (e.g. HVAC model) and FMU Import (e.g. A controller FMU generated from Simulink)
- Definition of exchanged data and flows between components (e.g. Air flows, signal flow between buttons in the Human Machine Interface (HMI) and the climate controller)
- Provide Traceability between logical components and functions, e.g.: “compute needed temperature” function is implemented by an ECU controller defined as a logical component
- The **Physical** layer enable the definition of the detailed 3D design in models (e.g. climate control mechanical parts, HVAC tubing part, electrical wires, etc.), as well as the instantiated embedded software code and its related hardware

This RFLP approach enables the modeling of the interaction chain of multi-disciplines component such as the climate control software controller algorithm, information flows from sensors and driver interfaces to actuators, air flows in the air conditioning system, mechanical parts and 3D vehicle mockup. This multi-discipline modeling enables verification that the system will deliver the necessary climate comfort requirements, as well as meeting the demands of demisting, odor control, noise minimization, energy efficiency and performance such as heating / cooling time in the context of the new vehicle system being studied.

Integrating, virtualizing and simulating all the climate controller system functions allow the early verification and validation of the behavior of the climate control systems in the context of the complete vehicle.

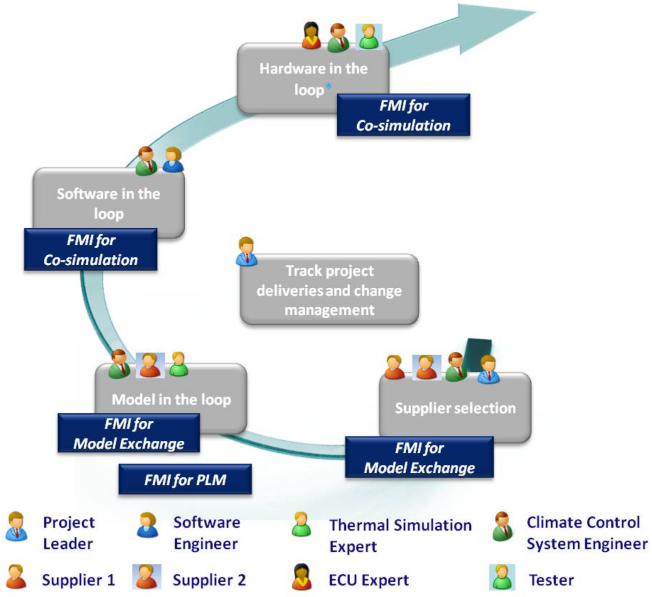
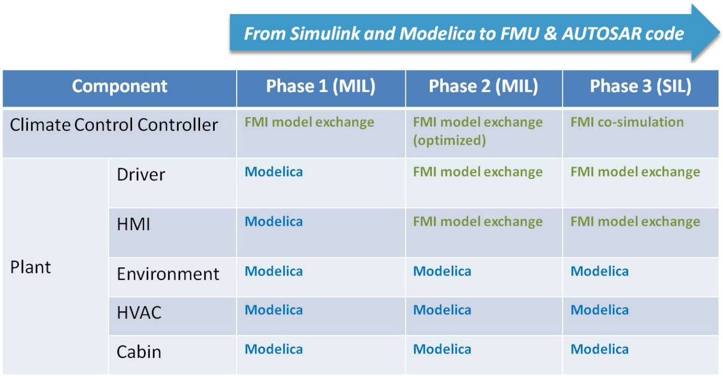


Fig. 10 Scenario Overview of Development Process

5.2 *MIL, SIL and HIL Collaborative Development – Scenario Overview*

The scenario (shown in figure 10) demonstrates a typical end-to-end engineering process with multiple stakeholders involved in developing the climate control system and the various embedded systems activities from multi-physics modeling and simulation, through to code generation, test and co-simulation. The scenario is composed of 3 phases: Supplier Selection, Model in the Loop (MIL) simulation, Software design and Software in the Loop (SIL) simulation. At each phase, the climate control system project leader is able to track the activities and deliveries.


From Simulink and Modelica to FMU & AUTOSAR code

Component		Phase 1 (MIL)	Phase 2 (MIL)	Phase 3 (SIL)
Climate Control Controller		FMI model exchange	FMI model exchange (optimized)	FMI co-simulation
Plant	Driver	Modelica	FMI model exchange	FMI model exchange
	HMI	Modelica	FMI model exchange	FMI model exchange
	Environment	Modelica	Modelica	Modelica
	HVAC	Modelica	Modelica	Modelica
	Cabin	Modelica	Modelica	Modelica

Fig. 11 Model and co-simulation formats used during MIL and SIL phases

Figure 11 shows that as the scenario evolves with the creation of more accurate models and the resulting generated software. FMI is then used for a complete co-simulation of all the climate control system components.

Phase 1: Climate Controller Suppliers – Benchmarking and Selection

The first phase of the scenario deals with supplier selection. Two different suppliers provide their controller models (in FMU for model exchange format) to the OEM using an integrated PLM platform. Using the dedicated security rights in the PLM platform, each supplier can access a dedicated virtual workspace to consult the OEMs requirements (request for information, request for quotation) and deliver their work package back to the OEM. By leveraging FMI, the suppliers can maintain a high level of intellectual propriety protection during Request for Information / Request for Quotations processes.

The OEM can then compares the two provided controllers by simulating them within the 3D virtual mockup of the new vehicle at conceptual design stage (shown in figure 12), and selects the supplier controller model that gives the best results.

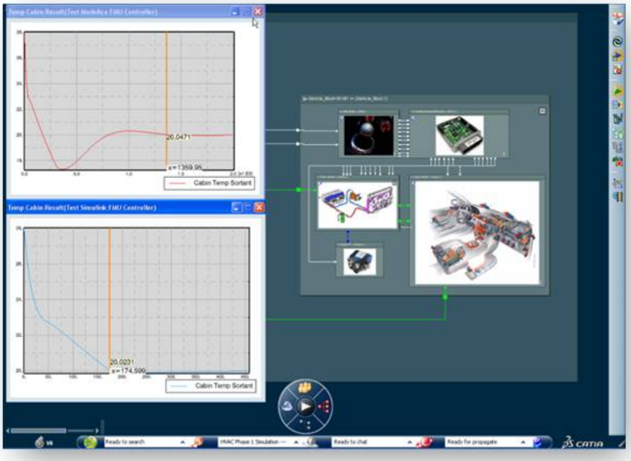


Fig. 12 Comparing FMU's from different suppliers in order to determine supplier of choice

Dedicated project management functionalities in the PLM platform enable the tracking of the simulation results and decisions taken for the "supplier selection" milestone.

Phase 2: Model-In-The-Loop Optimization for Early Validation of the Climate System

During the Model-In-the-Loop phase of the scenario, a controller parameter is optimized using the SIMULIA Scenario Definition tool for cross domain system exploration and optimization using FMI (see figure 13).

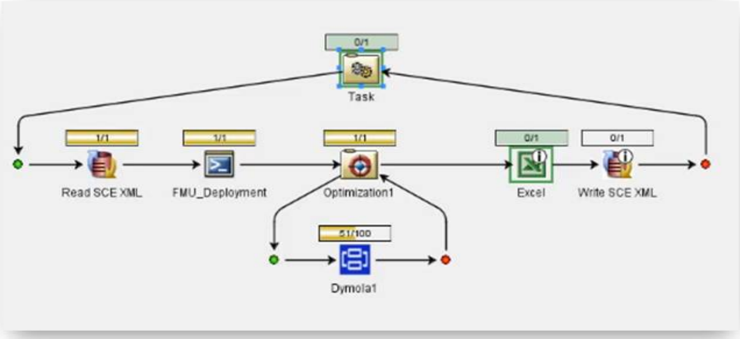


Fig. 13 Using MDO for Model-in-the-Loop and controller optimization

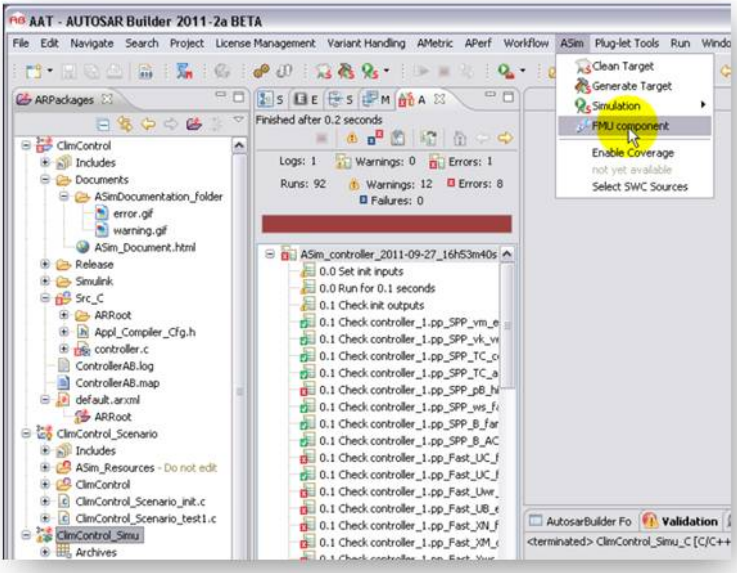
The optimized controller is then simulated with the virtual plant to validate the behavior in the 3D virtual mockup of the new vehicle. Using the RFLP traceability capabilities, test case results corresponding to the performance requirements can be checked, with the actual simulation results achieved being linked to the underlying requirements.

Phase 3: Controller Software Development and 3D Virtual Mockup Co-simulation

The next step is to consider the implementation of the C source code of the controller into the loop. Code is semi-automatically generated from the Simulink model. After software unit testing, a FMU of the controller, including the embeddable AUTOSAR code[6], is then generated (as shown on figure 14) and introduced in the global co-simulation of the virtual system.

The final co-simulation is played with a combination of the FMU including controller compiled code and plant models with both FMUs and Modelica models. Validation test cases and associated report deliveries are then updated and stored within the PLM platform.

This SIL co-simulation with the virtual 3D plant enables to validate the software embedded in the virtual vehicle context, to tune parameters calibration and detect integration issues before going to physical tests. This co-simulation step is crucial in decreasing the actual cost of physical prototypes and physical tests.



```
Finished after 0.2 seconds
Logs: 1    Warnings: 0    Errors: 1
Runs: 92   Warnings: 12   Errors: 8
           Failures: 0
ASim_controller_2011-09-27_16h53m40s
  0.0 Set init inputs
  0.0 Run for 0.1 seconds
  0.1 Check init outputs
  0.1 Check controller_1.pp_SPP_vm_e
  0.1 Check controller_1.pp_SPP_vk_v
  0.1 Check controller_1.pp_SPP_TC_c
  0.1 Check controller_1.pp_SPP_TC_a
  0.1 Check controller_1.pp_SPP_pB_h
  0.1 Check controller_1.pp_SPP_ws_f
  0.1 Check controller_1.pp_SPP_B_far
  0.1 Check controller_1.pp_SPP_B_AC
  0.1 Check controller_1.pp_Fast_UC_f
  0.1 Check controller_1.pp_Fast_UC_f
  0.1 Check controller_1.pp_Fast_Uwr_
  0.1 Check controller_1.pp_Fast_UB_e
  0.1 Check controller_1.pp_Fast_XN_f
  0.1 Check controller_1.pp_Fast_XM_c
<terminated> ClimControl_Simu_C [C/C++
```

Fig. 14 FMU with embedded code generation shown in AUTOSAR Builder

6 Summary

The outlined solution provides a next-generation approach to systems engineering of cyber-physical products. It provides:

- A collaborative systems engineering development environment
- Persistence & navigation on systems engineering data, models, simulations and virtual experiences
- Uniform management of diversity with full versioning and configuration management of systems artifacts
- Traceability and impact analysis of all proposed and implemented changes
- Integration of legacy models & tools
- Support of the Modelica, FMI and AUTOSAR open standards

The solution presented, with its rich and open data structure, the inbuilt collaborative business process support, and the fully integrated domain specific modeling and simulation environments, is unique in industry today. It enables the ability to quickly and easily evaluate requests for changes, or the creation of new cyber-physical product or system variants. This approach offers better flexibility, both in business and expected systems performance terms, leading to a unified performance based systems engineering approach and optimization of the system and its development process.

The FMI standard has been successfully implemented in a number of industrial uses cases. While this paper has used an automotive climate control system as an example, it important to note that a similar approach, as well as the supporting tools, can also be readily applied to other industries. Today 34 tools are compliant to the FMI standard including CAD, existing co-simulation platforms, and non-Modelica simulation tools. The Modelisar generated FMI standard will continue to be supported, maintained and further developed within the Modelica Association.

References

[1] Modelica association website, `http://www.modelica.org`

[2] AUTOSAR website, `http://www.autosar.org`

[3] Functional Mockup definition, `http://en.wikipedia.org/wiki/Functional_Mock-up_Interface`

[4] MODELISAR project, `http://www.modelisar.com`

[5] FMI specifications, `http://functional-mockup-interface.org/fmi.html`

[6] Thiele, B., Henriksson, D.: Using the Functional Mockup Interface as an Intermediate Format in AUTOSAR Software Component Development. In: Modelica Conference (2011)

Chapter 15
Flexible Product Line Derivation Applied to a Model Based Systems Engineering Process[*]

Cosmin Dumitrescu, Patrick Tessier, Camille Salinesi,
Sebastien Gérard, and Alain Dauron

Abstract. Systems engineering enables the successful realization of systems, focusing on defining customer needs early in the development cycle. However, when the development of systems needs to rely on legacy designs there is little methodological support. Furthermore, in the automotive domain, product diversity increases system complexity so much, that reuse becomes much more difficult and time consuming. We believe a specific strategy must be adopted to prepare for reuse and to achieve systems engineering by reuse. While product line derivation provides the means to obtain single products form a collection of assets, there is a lack of flexibility and support from a systems perspective. In this paper we present an approach which takes into account systems engineering methodological aspects in product line engineering and provides a means to develop new systems by reusing existing designs. We present the implementation of the tool support for our approach based

Cosmin Dumitrescu · Alain Dauron
RENAULT
1 Avenue du Golf,
78288 Guyancourt, France
e-mail: {cosmin.dumitrescu,alain.dauron}@renault.com

Patrick Tessier · Sebastien Gérard
CEA, LIST
Laboratory of Model Driven Engineering for Embedded Systems
Point courrier 174
F91191 Gif sur Yvette, France
e-mail: {patrick.tessier,sebastien.gerard}@cea.fr

Camille Salinesi · Cosmin Dumitrescu
Centre de Recherche en Informatique
Université Paris I Panthéon-Sorbonne
90 rue de Tolbiac, 75013 Paris France
e-mail: {camille.salinesi,cosmin.dumitrescu}@univ-paris1.fr

[*] Published in proceedings of Complex Systems Design & Management, Paris, December 2012.

on the Papyrus[1] SysML modeller and exemplify the concepts through a derivation example of the electric parking brake system.

Keywords: product lines, variability, derivation, model based systems engineering.

1 Introduction

Whether developing a new innovative system or reusing existing assets, engineers often need to adapt existing systems to new customers needs or deployment environments. In the case of automotive systems, the family of systems assets are useful (though not sufficient) to derive the specification for a new vehicle system project. While existing designs are used, improvements or new functionalities are introduced with each new iteration, which means that systems frequently require redesigning or altering the products.
Product line engineering derivation brings us closer to reuse, but from the systems engineering (SE) perspective we believe it needs to improve:

a) *flexibility* - to satisfy the need of altering existing models, derived from a family of systems
b) *methodological support* - to follow the traditional SE process from requirements to detailed architectural design.

We propose an approach for the configuration of a system from a family of systems, performed in several successive steps, that gradually reduce the model scope, but also enable the specification of new model items. As support for our approach we use the SysML modeller Papyrus with the Sequoia add-on for managing product lines.

The paper is structured as follows: in section 2 we present some industry and background information as well as motivation behind the work. Section 2.1 presents existing work related to the derivation activity. In section 3 we present the scenario that the derivation needs to follow. We also include a brief presentation of the variability modeller we use. In section 4 we describe the implementation principles and an example derivation scenario of the electric parking brake system. Finally in section 5 we conclude the paper with a short summary and outlook on future work.

2 Industry Background and Motivation

Systems engineering, which represents a development paradigm that aims at developing complex systems faster and with full conformance to stakeholder requirements, does not fully solve the problems coming from a product diversity rich environment. Product lines however, present the advantage of lowering development costs by reuse of existing development assets, but its practices would need to

[1] `http://www.eclipse.org/papyrus`

be adapted to meet organization specific requirements and integrate existing activities. The core issue for this scenario represents the derivation and how to integrate it with development activities.

Product Lines represent a set of systems that share a common, managed set of assets throughout their entire life cycle. Two main phases are distinguished: domain engineering, concerning the development of core system assets and application engineering, related to product derivation.

While our research approach is based on the creation of a set of development scenarios for families of systems, the present article focuses on a single common case for the automotive industry: single system development, based on reusable assets from previous experiences. In general, automotive system families are not planned for a long time horizon and they are rarely the product of a dedicated, separate development cycle, with a clear "domain (systems) engineering" phase. This is because there is a relatively short cycle of vehicle feature evolution in order to keep up with a competitive market, but reuse attempts do exist concerning issues such as standardisation, definition of common software architectures, or vehicle platforms.

The most important challenge, however, comes from the product range [1], or "product diversity", with a staggering number of different configurations for each vehicle model: 10^{21} for the Renault "Traffic" van for instance. The variability impacts subsystems relative to the vehicle and eventually propagates to the component level, creating complex design and configuration problems. Furthermore, engineers need to add new model elements while each new project has its own specificities, but still relies on existing designs. In consequence we are confronted with a problem of constant evolution of family of system models. The article presents an approach to integrating product derivation in a model based systems engineering framework using SysML models. The work also represents the first stage of an effort to bring product lines closer to the MBSE world.

2.1 Research Background

As other publications have pointed out, derivation requires methodological and tool support. It is indeed an error prone and complex task consisting in performing complex configuration on a set of assets of different nature and with intricate inter-relations. Furthermore, derivation methodology needs to integrate with existing organization practices, in our case with the systems engineering development process as defined in [10]. In consequence the derivation of the product line needs to be regarded, as proposed by Djebbi [6], as a decision making activity, where the engineer might take into account existing alternatives or develop new assets to reach a completely defined product. However, another constraint is to allow the user to perform usual SE activities during derivation according to the framework proposed by Chalé [3], with a focus on flexibility, which is not an issue considered by the previously mentioned approach.

Deelstra points out in [4] that the derivation "is a time-consuming and expensive activity" and provides a through investigation based on real industry case studies, that identify the sources of these problems. We are confronted with a series of similar issues, such as managing the complexity of the derivation activity or evolving the family of systems by adding new assets during derivation, but in the context of a predefined framework for SysML models (Renault SysML profile).

In [15] Rabiser presents an approach to support product derivation through adaptation or augmentation of variability models. This approach is supported by the DOPLER tool suite and it explicitly captures derivation information in a decision model. The solution that we propose relies heavily on complementary information contained in the system model, such as predefined stakeholders or specific points of view. We do not capture role information for example in the variability model, as in our context this kind of information can be provided by family of system models. At the same time the SE framework provides an asset collection structure which proves useful during derivation. The problem of finding the right assets during derivation is also pointed out by Hunt [11] who provides an evaluation of the impact of organizing the asset base in different ways.

Product diversity is the cause of an outstanding complexity increase in both automotive software and system domain, as suggested by Astesana [1] and Tischer [20]. Variability and complexity impact all aspects of the organization activity, from marketing, customer interaction (through online configurators), engineering (through the design of families of systems and capitalization) and eventually manufacturing in the plant and logistics.

3 Context and Needs

In order to understand the industry needs, we have developed a set of systems engineering scenarios involving the reuse of assets (e.g., requirements, solution designs, models): synchronized systems development, integration of a single system into a product line, merge of two product lines, derivation of a single system from a system family, development of a single system based on reuse (through derivation). We have chosen the latter as a nominal case, that we exploit in this paper to define our goals and constraints, which are articulated around the following statements:

- Starting working context of the project:
    1. We assume that the family of systems (FoS) model is defined a priori;
    2. We assume that there is sufficient detail in the FoS description to derive a single complete system specification, as a consequence of previous development efforts
    3. As process input, we assume that a variability definition of the system context (environment, technical context and client services) is ”inherited” from the upper level system analysis (e.g. vehicle variability propagates to subsystems).
- In addition to the typical activities of the systems engineering process, we need to take into account other activities related to reuse:

1. The approach shall enable identification of *system scope reuse opportunities*: for that the engineer has to choose existing assets in relation to high level requirements, environment variability, technical context and commercial offer. These assets, related to the operational analysis viewpoint of the systems engineering process, define and place the system context and its use cases in relation to previously developed systems and and orient subsequent activities towards reuse.
2. The approach shall enable identification of *solution reuse opportunities*. The engineer is required to choose among potential solutions by performing concept synthesis/analysis and trade-off analysis. He may either decide to go for an existing solution, matching a similar ”system scope” and use cases, either develop a new architecture, to satisfy new requirements or to provide more efficient solutions to existing problem definitions.
3. The approach shall enable *identification of component and module reuse opportunities*. The engineer is required to match the system architecture with existing solutions and take advantage of existing components or modules. Thus, in a top-down approach, a common architecture can lead to reusable components. In the opposite case, a bottom up-approach may allow the engineer to match and combine existing components and modules to realize the required system functions.

- The output of the process consists of:

1. Allocation of requirements to components for the selected configurations.
2. Variability description for each component belonging to the selected configurations.

The statements that we presented are compatible with the general systems engineering process described in [3]. Figure 1 illustrates the relation between the activities presented above and system assets : with each reuse decision, variability contained in the description of the family of systems is solved, leading to an intermediate and partial configuration of the system.

Meanwhile, it is possible to introduce new assets for each partial configuration. The *target diversity* is specified, as input to the process, using the *Renault documentary language* [1]. The key activities where reuse opportunities should be evaluated

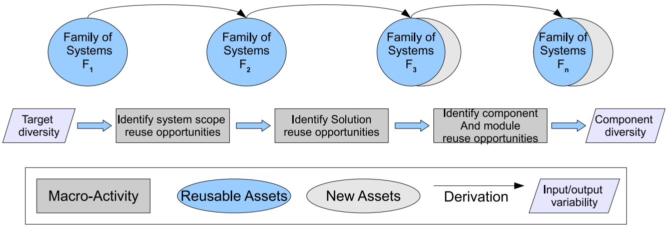


Fig. 1 Reuse related activities in the development process

correspond to transitions between the different viewpoints: operational, functional and physical [3], that correspond to our reference model-based systems engineering architecture framework. At a more general level, these transitions correspond to the passage from higher to lower levels of abstraction (to a more detailed description of the problem) or to a subsystem or subcomponent (e.g. environment to vehicle or vehicle to subsystem). Of course, if the requirements offset is too important in respect to the family of systems and if new requirements cannot be satisfied by existing solutions, the activities will be oriented towards development, instead of reuse.

A *partial system configuration* represents a collection of assets that includes variability and is obtained through partial resolution of variability contained in the family of systems definition.

We define *target diversity* as a partial system configuration of a system that refers only to: the system environment, the system technical context and customer services (commercial offer). It represents the diversity of customer needs that the system shall cover and the diversity of environments a system shall be used in.

Even if most of the system and variability assets are already defined before starting the derivation, the target diversity may have a larger scope than what is already defined. In consequence variability creation and resolution activities may be performed even during the development of the same product.

3.1 The Sequoia Tool

The goal of the Sequoia approach [19], developed by the CEA LIST is to help designers to build product lines based on UML/SysML models. Among its different functionalities, Sequoia supports variability modelling, variability propagation [18], and generation of a decision model.

Variability is expressed through the definition of a UML profile. To specify an optional element, the designer simply adds the stereotype *VariableElement* to the item. The stereotype *ElementGroup* introduces additional information through its properties, such as constraints between variable elements. In Sequoia, the decision model is used as a guide enabling to analyse all available variants and paths leading to a completely defined product. Once the derivation activity is launched, the choices described by the decision model are proposed to the user as a series of questions. The output of this process is a completely defined product and the user is not able to make any kind of modification to the initial model until the derivation step is over.

4 Results and Implementation

We have imagined two scenarios for the derivation of a single system with the Sequoia tool, which are compatible with the context and process related elements presented in section 3.

# Scenario A (*Derivation based on system viewpoints or stakeholders*): we consider the derivation process follows the development process and a certain sequence of choices performed to gradually reduce the model scope.
# Scenario B (*Non-functional requirements as criteria for derivation*): we take into account a derivation process where the final product has to satisfy certain non functional properties (e.g. weight, cost)

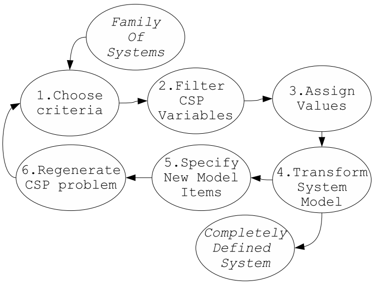
We can achieve these scenarios by performing several partial configuration on the family of systems model. In consequence, each configuration step should take into account only a part of the variable elements, while the rest are to be kept in the resulting model. At the same time the engineer should be able to modify or define new variable or non variable model items between stages.

4.1 Partial System Configuration

In order to perform a partial configuration we rely on the description of a CSP problem (classically defined as a triplet $P = (X,D,C)$, for any variable $x_i \in X$, $D(x_i)$ is the domain of x_i, while $c_{1..n} \in C$ a set of constraints related to $x_{1..k}$). The tool Sequoia allows the user to select among a list of predefined constraints: implication, equivalence, AtLeastOne, AllPosibilities, Alternative, or to define a custom expression. Each variable is related to a model item, which is stereotyped as *variable*, meaning the item can be present, absent or replaceable.

A viewpoint (on a system) is an abstraction that yields a specification of the whole system restricted to a particular set of concerns. [8] Suppose γ denotes a viewpoint restricted to a set of concerns $\phi_\gamma = \{\varphi_1..\varphi_n\}$. We suppose there is a user defined rule of correspondence $f_\gamma : \phi_\gamma \rightarrow X$, which associates to each concern φ a set of variables x from X. An assignment g_V is a function that maps any $x_i \in X$ to a value in $D(x_i) \cup \{\#\}$, using the symbol "#" to denote that the variable has not yet been assigned a value from D.

Table 1 Partial derivation activity flow

0. Define family of system concerns γ for derivation 1. Define viewpoint ϕ_γ. 2. Determine $V_d = f_\gamma(\phi_\gamma)$ the set of variables for partial derivation. 3. Assign values from $D(x_i) = \{0;1\}$ for each variable x_i until g is a complete assignment on V_d, such as $\forall x_i \in V_d, x_i \neq \#$. 4. For each $x_i = 0$ remove model items. 5. Allow the user to define new model items. 6. Regenerate CSP problem : $P' = (X', D, C')$, where $X' = X \setminus V_d$ and C' the set of related constraints.	 Family Of Systems 1.Choose criteria 2.Filter CSP Variables 3.Assign Values 4.Transform System Model 5.Specify New Model Items 6.Regenerate CSP problem Completely Defined System

4.2 *Towards a Scenario Based System Derivation, the Sequoia-R Profile*

The Sequoia profile was modified to support partial derivation, by introducing concepts that enable filtering constraints in respect to particular concerns defined by the user, as described in table 1. The concepts that were added are *DecisionCriteria* and *ProductCriteria*.

The concept *DecisionCriteria* is used to gather the variation groups, that describe constraints between several system assets or independent variable assets, that are not linked by the system specified to any other variation groups. It provides a way to map specific user concerns for derivation to configuration variables and constraints.

We define *variation group categories* depending on the systems engineering development process phases and system stakeholders. For example, we can define a series of DecisionCriteria which follow the different levels of analysis of a system, or DecisionCriteria which regroup choices concerning a certain development phase: requirement elicitation, technical requirement definition, solution alternatives, functional architecture definition physical architecture definition, etc.

One of the ways the system can be partially derived, is to associate the different stakeholders (stakeholder model element) included in the development process and in the system scope. We can perform for example a derivation where we resolve all customer or marketing variations but leave open all detailed design decisions or parameters. A variation group can be referenced by multiple *DecisionCriteria*, which allows for different aspects to be taken into account in the creation of a decision model for the system family.

The element *ProductCriteria* was added to express non-functional properties of the products. For example, it can contain an integer, a string or a boolean value which can be associated to the product.

4.3 *The Electric Parking Brake Case Study*

The electric parking brake (EPB) application is a variation of the classical, purely mechanical, parking brake which ensures vehicle immobilization when the driver brings the vehicle to a full stop and leaves the vehicle. We have chosen this

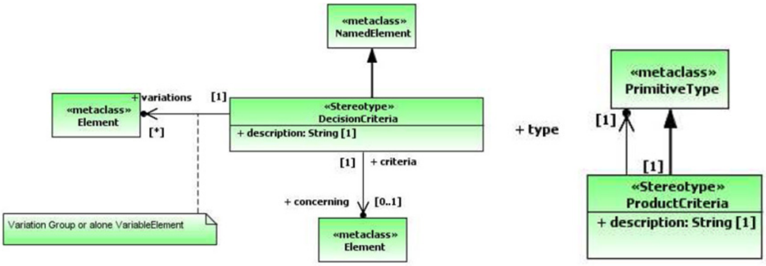

«metaclass» NamedElement
«metaclass» Element
+variations
[1]
[*]
«Stereotype» DecisionCriteria
+ description: String [1]
[1]
+ criteria
+ concerning
[0..1]
«metaclass» Element
Variation Group or alone VariableElement
+ type
[1]
«metaclass» PrimitiveType
[1]
«Stereotype» ProductCriteria
+ description: String [1]

Fig. 2 The stereotypes DecisionCriteria and ProductCriteria

example for its simplicity in respect to other automotive systems and because it contains many variations from the requirements to solution design and component implementation. Table 2 presents the main variations which were identified in the conception of this system, relative to the viewpoints specific to the Renault systems engineering process [3].

Table 2 Electric parking brake system main variations

System View-Point	Occurrence place	Variation point	V_Id	Variants
Operational Analysis	High Level Customer Requirements	Brake Lock	BL1	Automatic
			BL2	Manual
		Brake Release	BR1	Automatic
			BR2	Manual
		Hill Start Assistance	HSA	Present
	System Context	GearBox	GB1	Automatic
			GB2	Manual
		Clutch Pedal	CP	Present
		Vehicle Trailer	VT	Present
	System Environement	Regulation	Reg1	EU
			Reg2	US
Functional Viewpoint	Functional Decomposition	Braking Strategy	BS1	Dynamic
			BS2	Comfort
			BS3	Static
		Effort Monitor On Engine Stop	EMon1	Permanent
			Emon2	Temporary
Engineering Decisions	Technical Solution	Main Design Alternative	PC	Puller Cable
			CA	Caliper Actuators
Physical Viewpoint	Physical Decomposition	Electrical Action	DCM	DC motor
			ACT	Actuators

For the sake of simplicity we do not present all variations relative to the EPB system, but only the most representative ones. Other variations are present, such as specific behaviour relative to the vehicle model or range, depending on the configuration of CAN bus messages, vehicle weight, etc. As an observation, we point out that all variations occurring in the *operational viewpoint* propagate towards the behaviour and physical implementation of the system. At the same time new variations may appear due to alternative design solutions during the problem solving process.

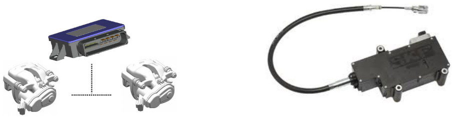


Fig. 3 The Electric Parking Brake system main design alternatives

These variants induce alternative representations of the SysML system model. We used the modeller Papyrus [18] in order to model the system and constraints related to variability, that are supported by its Sequoia add-on [18]. (e.g. VT excludes HSA; GB1 or GB2, BS2 requires CA)

4.4 Example of Derivation Scenario for the EPB

We have tested a derivation scenario, on the electric parking brake example, according to the implementation presented in section 4.2 in the tool Sequoia for managing model based product lines. In order to create the criteria sequence, we consider that the family of systems model is already available along with the required variability.

The first step from here, is to model the derivation criteria within the FoS model. After modelling all needed derivation criteria, we are able to customize the derivation process by selection of the appropriate order and desired criteria through a specific model wizard.

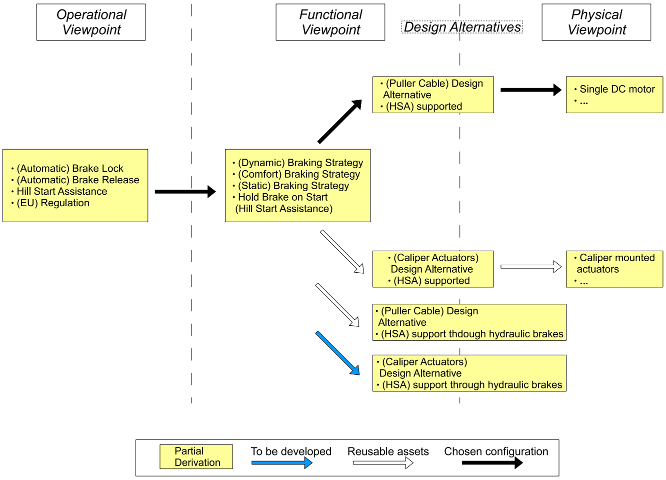
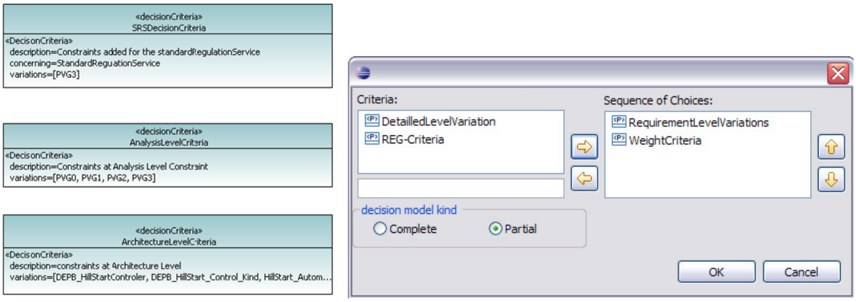

Operational Viewpoint
Functional Viewpoint
Design Alternatives
Physical Viewpoint
• (Automatic) Brake Lock
• (Automatic) Brake Release
• Hill Start Assistance
• (EU) Regulation
• (Dynamic) Braking Strategy
• (Comfort) Braking Strategy
• (Static) Braking Strategy
• Hold Brake on Start (Hill Start Assistance)
• (Puller Cable) Design Alternative
• (HSA) supported
• Single DC motor
• ...
• (Caliper Actuators) Design Alternative
• (HSA) supported
• Caliper mounted actuators
• ...
• (Puller Cable) Design Alternative
• (HSA) support thdough hydraulic brakes
• (Caliper Actuators) Design Alternative
• (HSA) support through hydraulic brakes
Partial Derivation
To be developed
Reusable assets
Chosen configuration

Fig. 4 Derivation scenarios for the EPB system

Figure 4 describes a sequence of choices that lead to the configuration of a single EPB system model. Two types of variability can be seen: one issued by stakeholders or customer requirements (the customer being represented by the marketing representative), and the other belonging to the solution domain, relative to design alternatives. Thus it is possible to reach different designs on the solution level, depending on the selected path and on the initial configuration on problem level. However some of the explored alternatives may not yet be implemented (e.g. a new design leading to cost reduction, increased performance, based on component availability etc.), and reusing assets will only be possible for the operational viewpoints and stakeholder requirements in this case.



«decisionCriteria»
SRSDecisionCriteria
«DecisonCriteria»
description=Constraints added for the standardRegulationService
concerning=StandardReguationService
variations=[PVG3]

«decisionCriteria»
AnalysisLevelCriteria
«DecisonCriteria»
description=Constraints at Analysis Level Constraint
variations=[PVG0, PVG1, PVG2, PVG3]

«decisionCriteria»
ArchitectureLevelCriteria
«DecisonCriteria»
description=constraints at Architecture Level
variations=[DEPB_HillStartControler, DEPB_HillStart_Control_Kind, HillStart_Autom...

Criteria:
DetailledLevelVariation
REG-Criteria

Sequence of Choices:
RequirementLevelVariations
WeightCriteria

decision model kind
○ Complete ● Partial

Fig. 5 Configuration of the partial derivation stages

The dialogue shown in figure 5 enables the user to specify the kind of model generation,partial or complete, along with a sequence of criteria that will impact the order of proposed choices. In the case where the decision criteria only correspond to a subset of all variation groups, after generating a decision model, only the linked choices will be proposed during the derivation phase. The result of a derivation based on a partial decision model will be a new family of systems model containing less variations as the previous one, but not the complete specification of the product.

In the case of a complete derivation, the tool will search the variation groups which were not selected in the derivation sequence, and will propose them as decisions after the treatment of the variations linked to the selected criteria.

In this way, we are not only able to impose a desired sequence of choices, in accordance to the SE process, but also to create several iterations of derivation and modelling, where new assets can be introduced or removed. It allows at the same time to distribute the derivation process between several actors, which could have different preoccupations, without the need to know the details of the product line model.

Since the derivations are successive, there is no problem concerning conflicting decisions, because the generation model is regenerated after each partial derivation. In the case where the derivations are made in parallel however, conflicts can appear, but this aspect is treated (at least for now) on the methodology level, by following a predefined flow of actions.

5 Conclusion and Future Work

We presented in this paper an approach to integrate model based systems engineering activities with product derivation and to render the configuration process more flexible, by reducing complexity and allowing the introduction of new model items at each step of the configuration.

Because the development of systems in the automotive domain often relies on reuse, but requires modification or evolution of previous designs, the process of family of systems derivation needs to address flexibility. Being difficult and error prone, derivation methodological support reduces significantly the possible paths of configuration and simplifies the overall process. We also presented the implementation realized in order to support this approach through the tool Sequoia for managing product lines.

Our future work includes the adoption of an expressive orthogonal variability model that would ease manipulation of variable assets. It includes modelling of automotive domain specific diversity and interaction with the Renault configuration language [1]. We also intend to investigate the issue of guidance during product line derivation and how this could benefit from systems engineering trade-off analysis in the selection or development of variants. While the current approach allows the specification of new assets, we would have to take into account evolution, maintenance and tracking changes for family of system model.

References

1. Astesana, J.M., Cosserat, L., Fargier, H.: Constraint-based Modeling and Exploitation of a Vehicle Range at Renault's: Requirement analysis and complexity study. In: Workshop on Configuration, p. 33 (2010)
2. Bosch, J.: Maturity and evolution in software product lines: Approaches, artefacts and organization. Software Product Lines, 247–262 (2002)
3. Chalé Góngora, H.G., Dauron, A., Gaudré, T.: A Commonsense-Driven Architecture Framework. Part 1: A Car Manufacturers (naïve) Take on MBSE. In: INCOSE 2012 (2012)
4. Deelstra, S., Sinnema, M., Bosch, J.: Experiences in software product families: Problems and issues during product derivation. Software Product Lines, pp. 120–122 (2004)
5. Deelstra, S., Sinnema, M., Bosch, J.: Product derivation in software product families: a case study. Journal of Systems and Software 74, 173–194 (2005)
6. Djebbi, O., Salinesi, C., Diaz, D.: Deriving Product Line Requirements: the RED-PL Guidance Approach (December 2007) (Presented)
7. Djebbi, O., Salinesi, C.: RED-PL, a Method for Deriving Product Requirements from a Product Line Requirements Model. In: Krogstie, J., Opdahl, A.L., Sindre, G. (eds.) CAiSE 2007 and WES 2007. LNCS, vol. 4495, pp. 279–293. Springer, Heidelberg (2007)
8. IEEE Recommended Practice for Architectural Description for Software-Intensive Systems. IEEE Standard 1471-2000, Institute for Electrical and Electronics Engineering, New York
9. Gomaa, H., Shin, M.E.: Automated software product line engineering and product derivation. In: 40th Annual Hawaii International Conference on System Sciences, HICSS 2007, p. 285a (2007)
10. Haskins, C.: Systems engineering handbook. INCOSE. Version. 3.2 (2010)
11. Hunt, J.M.: Organizing the asset base for product derivation. In: 2006 10th International Software Product Line Conference, pp. 65–74 (2006)
12. McGregor, J.D.: Preparing for Automated Derivation of Products in a Software Product Line, pp. 15213–13890. Carnegie Mellon Software Engineering Institute, Pittsburgh (2005)

13. OLeary, P., Mc Caffery, F., Richardson, I., Thiel, S.: Towards agile product derivation in software product line engineering (2009)
14. Perrouin, G., Klein, J., Guelfi, N., Jézéquel, J.-M.: Reconciling Automation and Flexibility in Product Derivation (September 2008) (Presented)
15. Rabiser, R., Grunbacher, P., Dhungana, D.: Supporting Product Derivation by Adapting and Augmenting Variability Models (September 2007) (Presented)
16. Rabiser, R., Dhungana, D.: Integrated support for product configuration and requirements engineering in product derivation. In: 33rd EUROMICRO Conference on Software Engineering and Advanced Applications, pp. 219–228 (2007)
17. Svahnberg, M., Bosch, J.: Evolution in software product lines. In: 3rd International Workshop on Software Architectures for Products Families, IWSAPF-3 (2000)
18. Tessier, P., Gérard, S., Terrier, F., Geib, J.M.: Using variation propagation for model-driven management of a system family. Software Product Lines, 222–233 (2005)
19. Tessier, P., Servat, D., Gérard, S.: Variability management on behavioral models. In: VaMoS Workshop, pp. 121–130 (2008)
20. Tischer, C., Muller, A., Mandl, T., Krause, R.: Experiences from a Large Scale Software Product Line Merger in the Automotive Domain (August 2011) (Presented)

Chapter 16
Towards an Architectural Design Framework for Automotive Systems Development

Hugo G. Chalé Góngora, Thierry Gaudré, and Sara Tucci-Piergiovanni

Abstract. This paper discusses the concepts of Model-Based Systems Engineering (MBSE) and of Architecture Frameworks (AF) and presents some preliminary results of current initiatives at Renault on these subjects. We advocate the adoption of a MBSE approach, i.e., the application of modeling to support a SE methodology covering the SE design process and activities and supporting the methods that are needed to carry out these activities. This results in the definition of an architectural design framework for the automotive systems development currently implemented in a SysML specialization. It is expected that this work will contribute to foster the reflection on an architecture framework for the automotive industry and stimulate discussions across the automotive community.

Keywords: complex systems, architecture framework, SysML, Model-Based Systems Engineering.

1 Introduction

Commercial automobiles have turned into very complex high-technology products in a relatively short time span. Different factors contribute to this complexity. One of them is the increasing number of vehicle functionalities supported by software, electronics and mechatronic technologies, a trend that does not seem to slow

Hugo G. Chalé Góngora
RENAULT, 1 avenue du Golf, 78288 Guyancourt, France
e-mail: hugo.chale-gongora@renault.com

Thierry Gaudré
RENAULT, 1 avenue du Golf, 78288 Guyancourt, France
e-mail: thierry.gaudre@renault.com

Sara Tucci-Piergiovanni
CEA, LIST,
Laboratory of Model-Driven Engineering for Embedded Systems,
PC 174, 91191 Gif-sur-Yvette, France
e-mail: sara.tucci@cea.fr

down. The involvement of carmakers in the development of these functionalities differs from one vehicle domain to the other (chassis, body, powertrain), ranging from black box integration to white-box developments. Another factor is the way in which car manufacturers have evolved from their historical mechanical and manufacturing background to the intricate organizations that develop the automobile products of today. Generally, this evolution has translated into development processes that are not as efficient, flexible and agile as they could or should be. Modern vehicle systems impose indeed a cultural and organizational change from carmakers, in that their development calls for the participation of many different professional fields each having its own language, its own jargon. Besides this multi-disciplinary aspect, vehicle systems are very often developed in a concurrent engineering approach, instead of a linear or cascade process. The advent of the electrical vehicle makes this last two factors even more evident, not only because of the "untraditional" technologies that carmakers need to master but also, and specially, because the arrival of new stakeholders, actors and interests around the electrical vehicle mean that the traditional scope of the automobile has changed.

The need to master these different complexity-inducing factors and improve the efficiency of product development, plus the arrival of the ISO 26262 standard (which besides from safety-related aspects, also raises issues concerning development processes of automotive systems, currently under-formalized) have motivated Renault to deploy Systems Engineering. The first visible results of this deployment were some evolutions of Renault's engineering divisions: first, a list of vehicle systems covering main classical functionalities (e.g. braking, air-conditioning) was built, to improve requirements consistency between the whole vehicle and its elementary parts. Second, it was decided to deploy a standard SE process on certain vehicle systems based on their complexity or innovative character. This process is in line with INCOSE references and adapted to specificities of automotive systems, for instance the management of re-usable system references (Chalé Góngora et al. 2010). As a result, a shared SE data model and process, and standard document templates, were elaborated.

The SE process at Renault has mainly been (and still is) a document-centric one depending largely on testing and prototyping performed at component or specialty domain levels. The authoring of the different objects of the process (e.g. use-cases, requirements, and architectures) is somewhat troublesome and time-consuming. One of the reasons for this is that these objects are elaborated by means of manual transformations of ad-hoc data and information contained in different documents that are transmitted from one activity of the process to the other and from one engineering domain to the other. Another reason is that the methods for elaborating the objects are applied (and to a certain degree described) inconsistently. Still, in order to avoid creating another complexity-inducing factor, all the objects of the systems design process should be described as clearly as possible. A better formalization of processes and of the process objects should certainly contribute to avoid confusion and misinterpretations in the development of systems. This is further stressed by the fact that Renault, like all other car manufacturers, relies heavily on third parties to develop vehicle systems.

One of the main drawbacks in implementing the SE process can then be summarized as a lack of semantic consistency among the different objects and activities of the process and a lack of integration and alignment of both activities and engineering disciplines. While there might not be one single solution to all these problems, we believe that the use of formal and informal (but consistent) models that comply with a common semantic model could facilitate systems engineering activities and avoid the encountered drawbacks of document-centric, error-prone implementations of the process. Our current improvement initiatives are divided into two main activities: The definition of common unambiguous semantics through the formalization of SE terminology in a formal ontology, as presented in (Chalé Góngora et al. 2010 and 2011); and a computer-assisted SE process that formalizes and partially automates the system design activities. This paper presents preliminary results of the latter initiative. The initiative advocates the adoption of a Model-Based Systems Engineering (MBSE) approach, i.e., the application of modeling for the support of SE activities. The envisaged modeling approach has the following features: 1)It complies with architecture frameworks principles. Following these principles, the model of the system is defined and analyzed at different levels of abstraction and from different viewpoints. At each level, the system model is described by a set of views (typically, graphical representations). A view describes the architecture of the system at some level of abstraction and in conformance to a viewpoint. Interestingly, relationships between (i) elements at different level of abstractions and/or (ii) belonging to different viewpoints are explicitly traced by well-defined refinement/decomposition methods. The paper presents how SE activities are formalized through the definition of an architectural design framework and related models. 2) It uses standard models for architectural framework description. In order to build standard models, a standard architecture modeling language is needed. Such a language must support (i) system architecture structuring and decomposition, (ii) abstraction and refinement, (iii) system viewpoints definition, and (iv) traceability between elements of the model. Today, the SysML language (SysML 2010) represents a good and usable choice, being a widely accepted standard for system modeling, especially in industry domains. The paper presents the use and specialization of SysML for the description of the proposed architectural design framework.

The paper is organized as follows: Section 2 discusses concepts of MBSE and of Architecture Frameworks (AF), detailing objectives and motivations of our approach. Section 3 briefly presents the systems design process and outlines the proposed architectural design framework. Section 4 presents the SysML diagrams that implement the different views of the architectural design framework, along with the data model based on a SysML specialization. The paper concludes with a presentation of possible extensions of the proposed architectural design framework to tend toward the definition of a true and complete architecture framework for the automotive industry.

2 Towards MBSE

Systems Engineering (SE) proposes a holistic approach and a set of principles and methods for successfully engineering systems that meet stakeholder needs and maximize the value delivered to stakeholders. MBSE, on the other hand, aims at producing a consistent system model that contains all the information that is required to completely describe, verify and validate the system. The benefits of implementing MBSE include an improvement of quality, through a more rigorous and costless traceability between requirements, design, analysis and testing; an increase in productivity, through the reuse of models and automated document generation (Estefan 2008 and Friedenthal et al. 2008); and an improvement in communication, by integrating views of the system from multiple perspectives (Wand and Dagli 2011). At the heart of MBSE, lies the concept of architecture, which can be defined according to ANSI/IEEE Std 1471-2000 (IEEE 2000) as "the fundamental organization of a system embodied in its elements, their relationship to each other and to the environment, and the principles governing its design and evolution". Therefore, according to the principles of SE, wherever there is a need there should be an architecture and wherever there is an architecture there should be a need (Watkins and Dagli 2010). Furthermore, the results presented in (Elm 2011) and (Hounour 2010 and 2011) show a strong relationship between project performance and product architecture and trade study capabilities, on one hand, and between project success and the relative weight of these same activities with respect to the overall cost of the project, on the other hand. Not surprisingly, in many industrial domains, the set up of system architecture design activities and the use of AF has proven particularly beneficial for the management of large complex system projects involving different organizations (Broy 2009).

An AF can be defined as a standard set of elements that enables and facilitates the description of architectures in a complete, consistent, homogeneous and unambiguous way and provides guidance on how to create such description. An AF helps to understand the way in which a specific domain develops products or services. Some well known examples of architecture frameworks are DoDAF, MoD-AF and TOGAF. These examples provide concepts, guidelines, best practices, and methods that are tailored for the needs of specific domains, respectively, defense and information technologies. They apply similar principles to organize and structure information using different levels of abstraction.

In the automotive industry, a standard AF does not exist yet, although this concept and its potential benefits have been discussed before, like in (Broy 2009). The work presented in this paper is intended to provide the Systems Engineering groundwork for starting the construction of an architecture framework for the automotive industry, by defining an architectural design framework which only supports the system design process, i.e. the main "Technical processes" defined by the ISO/IEC 15288 standard. In the future, the Renault intends to define and share with other automotive actors a true and complete AF for the automotive domain, which will provide, in its final version, an overarching set of concepts, guidance, best practices, and methods that will support all the processes defined by ISO/IEC 15288 standard (Enterprise, Agreement, Project and Technical processes).

The purpose of the AF is to ensure that system architectures can be understood, shared, analyzed and compared across organizational and project boundaries, in order to help decision makers to make key decisions more effectively thanks to structured, organized information sharing. This shall enhance the efficiency of the value creation stream and facilitate the integration of innovations into the vehicle and the reduction of risks. The proposed AF for automotive systems and the systems design process that it supports are described in the following paragraphs.

3 The Starting Point

The Systems Design Process. When defining the systems design process, we tried to answer to a simple question: how do we design our systems? Or, rather, how *would we like* to design our systems? The logical answer to this question was that we should strictly follow Systems Engineering principles (of course). So, not surprisingly, the four main activities of the systems design process are the Elicitation of Stakeholders Requirements, the Elaboration of System Technical Requirements, the Functional Architecture Design and the Organic Architecture Design.

During the "Elicitation of stakeholders requirements" activity, we identify the scope, the environment and the main actors concerned in the development of the system (the system of interest). The original stakeholders needs are captured, analyzed (according to each operational context of the system's lifecycle) and generally structured using a use-case approach. These use-cases are refined in their turn by operational scenarios. The outcome of this activity is a set of (individually) clear stakeholder requirements, including the purpose, missions, objectives and scope of the system, as well as other technical measures of the "success" of the system.

The goal of the "Elaboration of the system technical requirements" activity, is to transform the stakeholder requirements into a set of precise, achievable and consistent requirements, with tradeoffs between inconsistent or contradictory stakeholder requirements being possible. System technical requirements are organized and classified according to their type (behavioral, functional, performance, RAMS, operational, design, withdrawal, recycling).

The objectives of "Functional architecture design" activity are, on one hand, to identify, describe and make more precise (by decomposing them) the functions or transformations that the system must perform in order to satisfy the requirements (mainly functional requirements). And, on the other hand, to specify the flows (i.e., information, energy or material flows) that are produced or consumed by the functions. The internal dynamic behavior of the system is also an outcome of this activity and corresponds to the logic scheduling of the execution of the system functions.

The outcome of the "Organic architecture design" activity is the definition of the components of the system, of their interfaces and of their connections. The system architecture shall fully satisfy the system technical requirements, including non-functional requirements (like cost, weight and size of the system, kind of interfaces, forbidden or authorized use of materials, etc.), and other technical measures or criteria (measures of effectiveness, key performance parameters, etc.). During this activity, the system functions, defined in the previous activity, are allocated to the constituents of the system and, eventually, decomposed again to achieve an optimal allocation. The flows produced and consumed by the allocated functions are associated to the interfaces and connectors that shall transport them. New interfaces can be defined, depending on architectural choices that are made.

"Dependability" and the "Evaluation and Optimization" activities take place during all the systems design process. It is not the aim this paper to deal with these activities. For more information about the Dependability analysis activity, the reader can refer to (Chalé Góngora et al 2010 and 2011), where we focus on the safety aspects of dependability as the main objective is to comply with the ISO 26262 standard. The "Evaluation and Optimization" activity is presented in (Doufene et al. 2012). Similarly, all the activities of the systems design process are the object of "Verification and Validation" activities, such as inspections (of documents, models or plans), simulation, model checking, design reviews, etc.

The systems design process is recursive and iterative, with several design loops that can be repeated partially or completely following the evolution of needs or the emergence of bottom-up constraints, like the impossibility to realize certain components of the system. At the end of (one loop) of the process , sub-systems are developed following either a similar process, if the sub-system is complex in the sense that it calls for different professional fields (disciplines or technologies), or a domain-specific process, if the subsystem calls for only one professional field such as software, electronics, mechanics, etc. The architectural design framework presented below provides the methods and views (or sets of models) that completely support the systems design process presented here.

Background of the architectural design framework. The background of the architectural design framework presented in this paper can be found in the work of (Krob 2009 and 2010) concerning the architecture of complex systems. This work presents an architecture framework derived from the SAGACE method (Penalva 1997), which is structured on three main pillars: a modeling methodology; a graphical representation based on a nine-viewpoint matrix, or analysis grid, consisting of three main viewpoints (operational, functional and structural or constructional), each refined by three temporal perspectives; and a graphical modeling language. When adapting this framework to the systems design process described above, we used the same main viewpoints to support the activities of the process and we added a requirements viewpoint for consistency purposes. The Operational, Requirements, Functional and Constructional Viewpoints respectively support the Elicitation of stakeholder requirements, the Elaboration of system technical requirements and the Functional and Organic architecture design activities. Figure 1 presents an overview of viewpoints of the proposed architectural design framework.

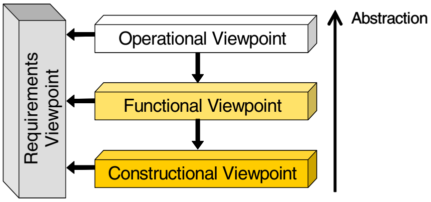


Fig. 1 Viewpoints of the architectural design framework

Table 1 Summary of Methods, Views and Viewpoints of the Architectural Design Framework

Systems design process Activity	Used Methods	Supporting Views	Defining Viewpoints
Elicitation of stakeholder requirements	Systemic Analysis	Maximal System Scope System Environment	Operational
	Definition of the Operational Scope	Operational Context External Interfaces	
	Identification of Use-Cases	Use-Cases	
	Definition of Operational Scenarios	Operational Scenarios System Working Modes	
	Formalization of Stakeholder and High-Level Requirements	Stakeholder Requirements High-Level Requirements	Requirements
Elaboration of system technical requirements	Requirement Analysis	System Technical Requirements	Requirements
	Traceability [1]	---	
Functional architecture design	Functions Identification & decomposition	Functional Breakdown Structure	Functional
	Structuring	Functional Architecture	
	Requirements Allocation on Functions	Allocation on Functions	
Organic architecture design	Components Identification	Product Breakdown Structure	Constructional
	Allocation & Structuring	Organic Architecture	
	Requirements and Functions Allocation on Components	Allocation on Components	

[1] There is not one single view supporting traceability. We use instead the mechanisms proposed by SysML in the other views and diagrams of the architecture framework.

More precisely, each activity of the systems design process calls for specific design and analysis methods, as proposed in (Krob 2009 and 2010). These methods are supported by one or more views or sets of models of the system (implemented as specialized SysML diagrams in this paper), which are related to a viewpoint. We may consider these viewpoints and their associated views as different abstraction levels in which the system is represented at a certain level of detail. Table 1 presents a summary of the methods and views and their relationship to viewpoints and activities of the systems design process.

As we can observe on Table 1, the number of methods and **views** supporting the Elicitation of Stakeholder Requirements activity is higher than those of the other activities. This indicates that the workload involved in this activity is also more important, which seems only logical, since stating the problem is more important than solving the problem in Systems Engineering.

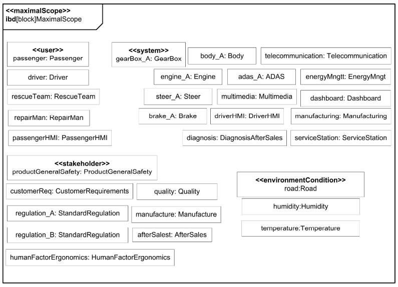
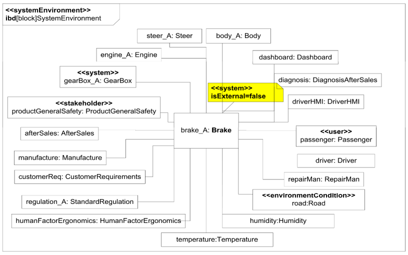
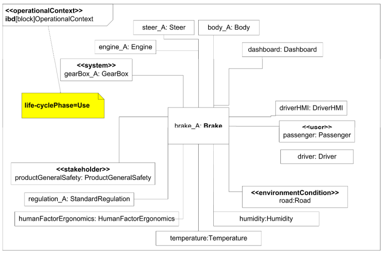
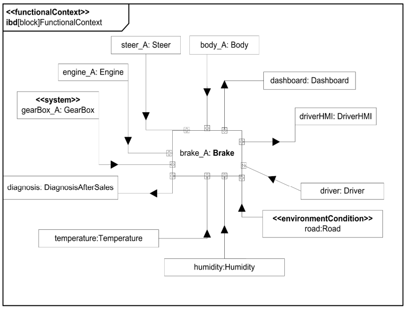
As explained in (Chalé et al. 2012) the data model and the architectural design framework can be implemented in SysML. SysML is a widely accepted standard language that supports well systems modeling, but because it is a multi-purpose language, not a domain specific language (DSL), it is not particularly adapted to the automotive domain. Most importantly, SysML does not support per se the MBSE process and methods that are being deployed in Renault. Some extensions and restrictions of the SysML language are thus necessary. The adaptation of the SysML to support our data model is performed in the form of a specialization (or profile), the standard method used in UML for language extensions. Finally, since this architectural design framework is formalized in a machine-interpretable language, we can perform automatic verifications and calculations on complex system models (this feature is not presented in this article). The methods and the SysML specialization (objects and diagrams) that constitute the architectural design framework are detailed in the following paragraphs.

4 The Architectural Design Framework: Main Views and SysML Specialization

Main Views. This section summarizes the set of views defined for the Operational, Functional, Constructional and Requirement viewpoints along with the SysML diagrams chosen for the view. Views associated to the Operational Viewpoint are presented in Table 2 and Table 3, which presents structural and behavioral views respectively. Table 4 shows views for both the Functional and Constructional viewpoints. Table 5 presents views for the Requirements Viewpoint.

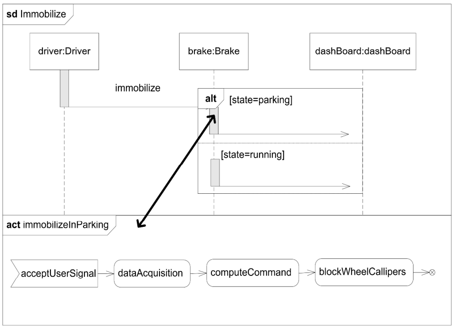
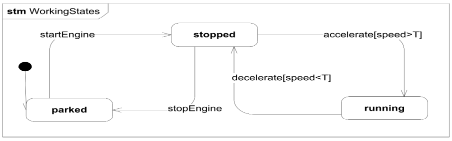
Operational Viewpoint. The set of views in Table 2 represents the most abstract views of the MBSE process. They are the artifacts produced during the first phase of the process where the actors, the system scope, the system environment and high-level interactions have to be identified. The system is here a black box as analyzed by a user's perspective.

Table 2 Structural Views for the Operational Viewpoint

View	SysML Diagram	Modelled Elements
Maximal Scope	Internal Block Diagram  <<maximalScope>> ibd[block]MaximalScope <<user>> passenger: Passenger <<system>> gearBox_A: GearBox body_A: Body telecommunication: Telecommunication driver: Driver engine_A: Engine adas_A: ADAS energyMngtt: EnergyMngt rescueTeam: RescueTeam steer_A: Steer multimedia: Multimedia dashboard: Dashboard repairMan: RepairMan brake_A: Brake driverHMI: DriverHMI manufacturing: Manufacturing passengerHMI: PassengerHMI diagnosis: DiagnosisAfterSales serviceStation: ServiceStation <<stakeholder>> productGeneralSafety: ProductGeneralSafety <<environmentCondition>> road:Road customerReq: CustomerRequirements quality: Quality humidity:Humidity regulation_A: StandardRegulation manufacture: Manufacture temperature:Temperature regulation_B: StandardRegulation afterSalest: AfterSales humanFactorErgonomics: HumanFactorErgonomics	Maximal scope actors: all the stakeholders, users, systems and environment conditions potentially impacting the system under study. Maximal scope categories: set of actors' categories to include in the maximal scope (at least one actor of each category).
System Environment	Internal Block Diagram  <<systemEnvironment>> ibd[block]SystemEnvironment steer_A: Steer body_A: Body engine_A: Engine dashboard: Dashboard <<system>> gearBox_A: GearBox diagnosis: DiagnosisAfterSales <<system>> isExternal=false <<stakeholder>> productGeneralSafety: ProductGeneralSafety driverHMI: DriverHMI brake_A: Brake afterSales: AfterSales <<user>> passenger: Passenger manufacture: Manufacture driver: Driver customerReq: CustomerRequirements repairMan: RepairMan regulation_A: StandardRegulation <<environmentCondition>> road:Road humanFactorErgonomics: HumanFactorErgonomics humidity:Humidity temperature:Temperature	System boundary: distinction between external actors and system actors (system actors: actors being part of the system of interest). High-level interactions: interaction points and connections between external actors and system actor(s).
Operational Context	Internal Block Diagram  <<operationalContext>> ibd[block]OperationalContext steer_A: Steer body_A: Body engine_A: Engine dashboard: Dashboard <<system>> gearBox_A: GearBox life-cyclePhase=Use driverHMI: DriverHMI brake_A: Brake <<user>> passenger: Passenger driver: Driver <<stakeholder>> productGeneralSafety: ProductGeneralSafety <<environmentCondition>> road:Road regulation_A: StandardRegulation humanFactorErgonomics: HumanFactorErgonomics humidity:Humidity temperature:Temperature	Subset of the system environment elements related to a specific life-cycle phase. Life cycle-phase: phases of the system project life-cycle; i.e., development, manufacture, delivery, assembly, use, maintenance, recycling.
External Interfaces	Internal Block Diagram  <<functionalContext>> ibd[block]FunctionalContext steer_A: Steer body_A: Body engine_A: Engine dashboard: Dashboard <<system>> gearBox_A: GearBox driverHMI: DriverHMI brake_A: Brake diagnosis: DiagnosisAfterSales driver: Driver <<environmentCondition>> road:Road temperature:Temperature humidity:Humidity	Refinement of actor/system interactions and interaction points previously identified, represented by two contexts: Functional Context: refinement through specification of flows circulating to/from the system (matter, energy, information). Physical Context: refinement through the specification of the physical means the flows circulate through (mechanic, electric, network).

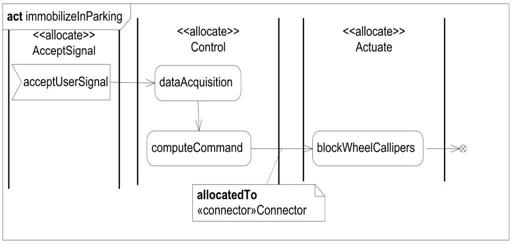
The set of views in Table 3 builds from the structural operational views. Once actors, the system boundary and high-level interactions have been identified, the expected behavior of the system must be specified. To this end System Use Cases are used. System Use Cases are then refined in Sequence Diagrams, to describe needed behavior in terms of expected interactions (actor requests and system operations). The next step consists in adding complementary views to interactions. To this end Activity diagrams and State Machines are employed. Ideally, for each operation an activity diagram should be defined. State machine can help in completing the activity diagram view by defining conditions (upon transitions) to refine alternative paths of expected interactions.

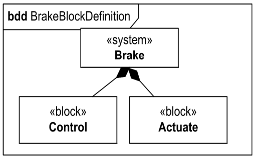
Table 3 Behavioral Views for the Operational Viewpoint

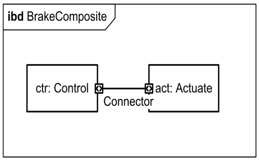
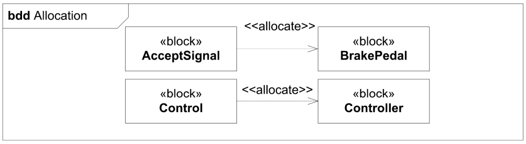
View	SysML Diagram	Modelled Elements
Operational Scenarios	Sequence Diagram for operations and actor requests, Activity Diagram for internal behaviours, actions and flows 	Detailed interactions between the system and external systems/user/environment to realize the use cases. Actor requests and Operations the system should perform upon requests. Actions to refine/detail each operation. Control/data flows between actions.
System Working States	State machines 	State machines to describe alternative conditions for operational scenarios.

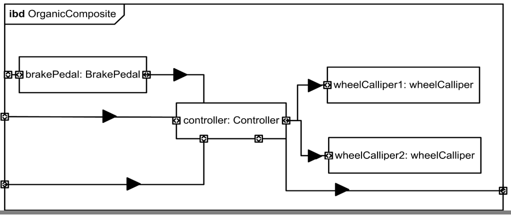
Functional and Constructional Viewpoints. The set of views in Table 4 is constructed from the whole set of Operational Views. First, system functions are obtained by properly grouping or refining Activities (actions) and allocating them on SysML Blocks. Ports and connectors should convey a type of flow compatible to flows of external interfaces and to object flows identified in Activity Diagrams. A further allocation (or grouping) of system functions into physical components yields the construction of the organic architecture. One particular characteristic of the automotive industry is a strong carry-over constraint (mostly justified by cost constraints). This implies that the physical components of the system are most of the time identified from existing Bills of Materials (BOM).

Table 4 Views for Functional and Constructional Viewpoints

<table>
<tr><th>View</th><th>SysML Diagram</th><th>Modelled Elements</th></tr>
<tr><td>Functional Breakdown Structure
Functional Architecture</td><td>Activity, Block Definition and Internal Block Diagrams


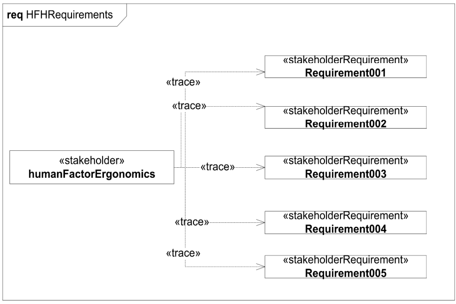
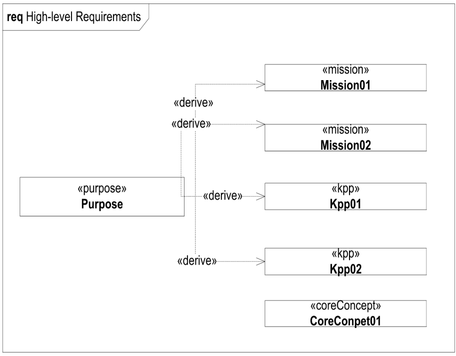
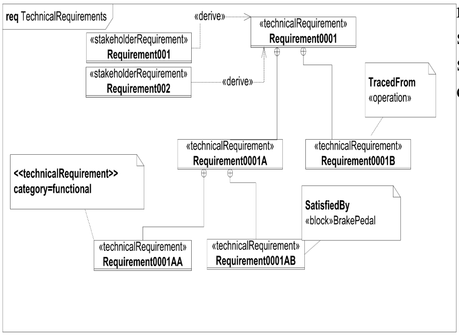



</td><td>System functions: actions identified in the Operational Scenario views can be regrouped or refined in system functions. Ports and Connectors connecting used blocks will conform to control/data flows identified in Operational Scenarios views. Type of flow: type of flow flowing across a connector (matter, energy, information).</td></tr>
<tr><td>Product Breakdown Structure
Organic Architecture</td><td>Block Definition and Internal Block Diagrams



</td><td>Physical components: defined by allocation of system functions.
Ports and connectors connecting used components are identified. Circulating flows are derived from allocated functions.
Type of physical connector: type of physical means the flows circulate through (mechanic, electric, network).</td></tr>
</table>

Requirements Viewpoint. Table 5 shows views for the Requirement viewpoint. As already pointed out, the Requirement viewpoint is orthogonal to other viewpoints (Figure 1). Nonetheless, each requirement view has a relationship with views of other viewpoints. In particular the Stakeholder requirement view is built at the very beginning after the identification of actors and system boundary. The whole set of stakeholder requirements is the starting point for identifying High-Level Requirements. The Technical Requirement view, instead, is typically built after the whole set of Operational Views is available. For instance, functional requirements typically stem from operations identified in sequence diagrams.

Table 5 Views for the Requirements Viewpoint

View	SysML Diagram	Modelled Elements
Stakeholders Requirements	Requirement Diagram  req HFHRequirements «stakeholder» humanFactorErgonomics «trace» «trace» «trace» «trace» «trace» «stakeholderRequirement» Requirement001 «stakeholderRequirement» Requirement002 «stakeholderRequirement» Requirement003 «stakeholderRequirement» Requirement004 «stakeholderRequirement» Requirement005	Stakeholder requirements: requirements as issued by stakeholders.
High-Level Requirements	Requirement Diagram  req High-level Requirements «purpose» Purpose «derive» «derive» «derive» «derive» «mission» Mission01 «mission» Mission02 «kpp» Kpp01 «kpp» Kpp02 «coreConcept» CoreConpet01	Purpose: rationale of the system, i.e. the reasons why the system should be realised Mission: what the system should do to fulfil the purpose. Measures of effectiveness or Key Performance Parameters (KPP): criteria defining how well the system achieves the intended purpose. Core Concept: statement defining the main constraints on system design
Technical Requirements	Requirement Diagram  req TechnicalRequirements «stakeholderRequirement» Requirement001 «stakeholderRequirement» Requirement002 «derive» «derive» «technicalRequirement» Requirement0001 TracedFrom «operation» «technicalRequirement» Requirement0001A «technicalRequirement» Requirement0001B <<technicalRequirement>> category=functional SatisfiedBy «block»BrakePedal «technicalRequirement» Requirement0001AA «technicalRequirement» Requirement0001AB	Functional, performance, interface requirements: unambiguous, consistent, verifiable set of stand-alone statements that identify system's characteristics or constraints.

SysML specialization and usage. This section presents the SysML specialization needed to model views defined so far. It is worth to be noticed that typical views, as the functional and constructional views, can be easily constructed in standard SysML. In particular the "allocation" concept allows grouping Activity actions or Blocks to other Blocks in straightforward manner. This feature allows supporting the construction of the functions (Blocks) from activity actions and of physical components (Blocks) from functions (Blocks). The component model, moreover, is very rich and supports physical and functional flow descriptions circulating between components. Note that object flows (flows in Activity Diagrams) can also be easily allocated on connectors (internal block diagrams) or associations (in block definition diagrams). Concerning requirements traceability, SysML already offers numerous relationships (refine, derive, verify, and satisfy) to describe traceability between design objects and no specialization has been necessary on this side. The only specialization has been introduced to model the concept of stakeholder, technical and high-level requirement as explained below. Concerning the behavioral views of Operational Scenarios, we mainly rely on those parts of SysML shared with UML (Interactions, Activity and State Machines). The language on this side is quite rich and no extension/specialization has been necessary so far. On the other hand, to model top-most views of the MBSE process (structural operational views and system use case views) a heavy specialization has been necessary. In the following subsections, SysML specialization is presented in the form of profiles to support above-defined views and diagrams construction. Guidelines for the SysML specialization usage are also presented.

Sysml specializarion and usage for structural Operational Views. In order to model structural views for the Operational Viewpoints, stereotypes have been introduced to model respectively: the different contexts in which actors play a role (from Maximal Scope to the Functional/Physical Context) and the actor concept itself. Figure 2 shows stereotypes introduced for contexts. They are applied to the enclosing blocks in the associated Internal Block Diagrams (see Table 2). Note that for the Operational Context, the attribute lifeCyclePhase identifies to what life-cycle phase the Operational Context refers to. Figure 3 shows specialized SysML ItemFlows. They are applied to connectors in IBDs for Functional/Physical Contexts in order to properly refine actor/system interactions.

In Figure 4 stereotypes for actors have been introduced. The introduction of this set of stereotypes has been mainly motivated by the fact that actors always appear as properties (or parts) of enclosing blocks (maximal scope, system environment, etc.) in Internal Block Diagrams. As in UML/SyML an Actor is always a Classifier and not a Property, a special stereotype called <<ActorPart>> has been introduced to be applied to properties of enclosing blocks. We've also defined an ad-hoc usage of the types of actor parts. The idea is to have each actor part typed by an actor category, i.e. a SysML Block stereotyped with the <<ActorCategory>> stereotype, herein introduced. The set of pre-defined categories can be also specialized by establishing a Generalization between the new Block (representing the newest sub-category) and its parent.

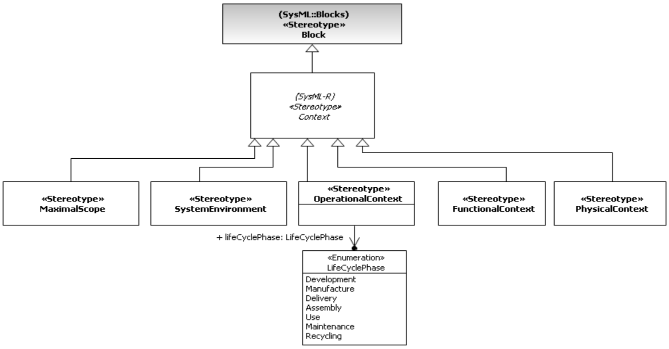


Fig. 2 The different contexts extending SysML Block

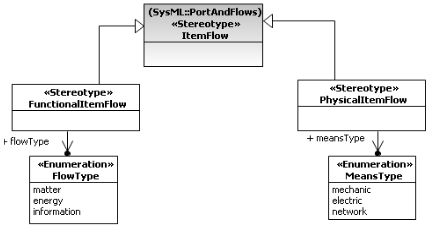


Fig. 3 Specialized flows for Functional/Physical Contexts

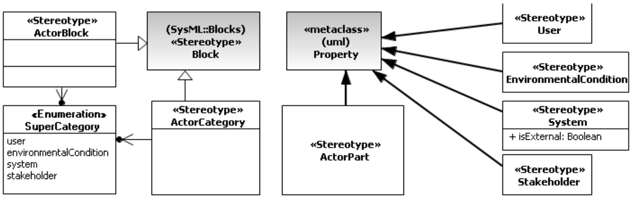


Fig. 4 Stereotypes for actors and actor categories

For practical reasons, it is useful that the topmost categories can also be applied as stereotypes to actor parts, to highlight the topmost category the actor falls in. This is why the stereotypes <<User>>, <<EnvironemantalCondition>>, <<System>>, and <<Stakeholder>> have been introduced. These stereotypes when applied to actor parts must comply with the super category set for the actor's type. Note also that the <<System>> stereotype has Boolean attribute isExternal. This attribute is set to true by default. During the system environment construction, it will be set to false to identify all actor parts included in the system boundary. The meaning of the stereotype <<ActorBlock>> will be explained in the next section.

SysML specialization and usage for high-level behavioral Operational Views. In order to model behavioral views for the Operational Viewpoints, stereotypes have been introduced to model, System Use Cases and Use Case Conditions, respectively (see Table 3). No specialization has been necessary right now to model operational scenarios.

System Use Cases are UML Use cases empowered by a set of Use Case Conditions. Figure 5 shows the introduced stereotypes to apply to Use Cases. As shown, Use Case Conditions are defined by creating UML Constraints stereotyped <<UseCaseCondition>>. Note that a use case condition can be written in natural language and/or may refer to more formal conditions called sub-life-cycle phases. Sub-life cycle phases are states of the vehicle/environment refining a given life-cycle phase.

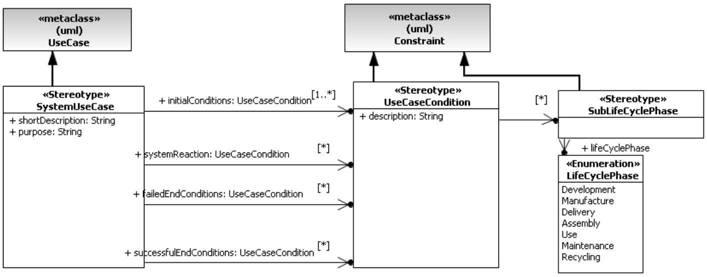


Fig. 5 Stereotypes extending Use Case and defining Use Case Conditions

In the previous sections we explained that through the definition of the System Environment, actor parts are identified as part of "system" or purely "external actors". Actors used in System Use Cases must be chosen among the actor parts identified in System Environment. However, the use of a standard UML Use Case poses some problem on this side. In UML Use Case diagrams, the interaction between the actor and the system is represented by an UML Association. An Association cannot be established between properties, i.e. between an actor part and the system. To solve this problem we introduced a constraint in our meta-model. This constraint establishes that each actor part must have a counterpart represented by a Block, stereotyped <<ActorBlock>>. These two elements represent conceptually the same actor: (1) they should be in one-to-one correspondence, (2) they should have the same name and (3) the <<ActorBlock>> specializes also the category of the actor part (its type) to inherit category's attributes (if any). Note that the same problem affects the example used in the SysML specification, annex B (OMG 2010), but no specific actions are suggested to solve the problem, an actor with the same name of the actor part is used in use case diagrams, without further constraints.

SysML specialization and usage for Requirement Views. In order to model the different set of requirements defined in views for the Requirement Viewpoint, stereotypes have been introduced to model respectively Stakeholders requirements, High-level requirements and Technical Requirements. Figure 6 shows introduced stereotypes to be applied to SysML Requirements.

Two types of requirements have been identified: high-level requirement and general requirement. Both of them are abstract and cannot be directly applied. High-level requirements summarize at a high level of abstraction the main goals and expected functionalities of the system (see Table 5). General requirements are traditional requirements enriched with several attributes as category (functional, performance, interface, etc.) and safety level/flexibility. For stakeholder requirements a special attribute is added, which is not shared with technical requirements. This attribute is called *isUseCaseRelated* and considers the fact that the requirement, expressed in natural language, must be later refined in a Use Case. A technical requirement instead, has another specific attribute called *prescribedOnExternalSystem.* Note that each technical requirement must have at least on "*derive from*" or "*trace*" relationship with at least one element, e.g. a stakeholder requirement, a use case, a scenario, a signal or an operation defined in a Sequence diagram, etc. This relationship witnesses that the technical requirement is formulated starting from the analysis of formalized user/system interactions. Each technical requirement must also have at least one "*satisfiedBy*" relation with at least one element amongst the external actors if *prescribedOnExternalSystem=true or* elements inside the system boundary if *prescribedOnExternalSystem=false.*

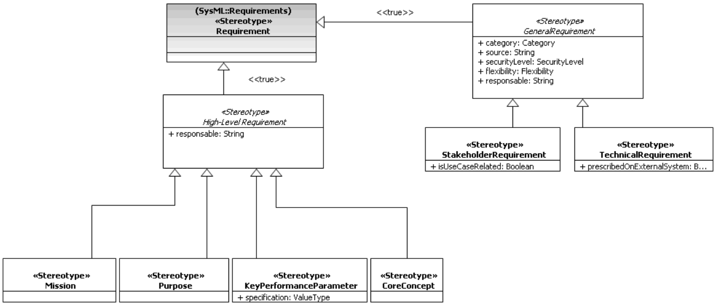


Fig. 6 Stereotypes for Requirements

5 Conclusion

In this paper, we presented the first results of the work started by RENAULT and CEA List on the specification of a architectural design framework and the specialization of the SysML language to support the RENAULT systems and system of systems design process.

The final goal of this activity is the elaboration of a complete, integrated AF for the automotive industry and the development of the MBSE environment which will support it. This AF will be able to support: Verification and Validation activities through simulation and optimization techniques; Product Line Management, to fill the gap between top-down systems design and bottom-up integration of legacy elements (components, technologies); Safety Processes, as defined by ISO 26262 standard; a seamless transition to Software Design environments using UML; Project, Agreement, and Enterprise processes such as defined by the ISO/IEC 15288 standard.

This kind of initiative would require more discussion and dissemination and, particularly, a wider adoption by the actors of the automotive industry. The work presented here will hopefully contribute to foster the reflection on this subject and stimulate exchanges across the automotive community.

References

Broy, M., Gleirscher, M., Kluge, P., Krenzer, W., Merenda, S., Wild, D.: Automotive Architecture Framework: Towards a Holistic and Standardised System Architecture Description. White paper, IBM Corporation. Technical Report, Technische Universität München. TUM-I0915 (2009)

Chalé Góngora, H.G., Gaudré, T., Taofifenua, O.: A Process and Data Model for Automotive Safety-Critical Systems Design. In: Proceedings of the 20th Annual International Symposium of the INCOSE (Chicago, IL). INCOSE, San Diego (2010)

Chalé Góngora, H.G., Guadré, T., Taofifenua, O., Topa, A., Levy, N., Boulanger, J.L.: Reducing the Gap Between Formal and Informal Worlds in Automotive Safety-Critical Systems Design. In: Proceedings of the 21st Annual International Symposium of the INCOSE (Denver, CO). INCOSE, San Diego (2011)

Chalé Góngora, H.G., Dauron, A., Guadré, T.: A Commonsense-Driven Architecture Framework. Part 1: A Car Manufacturer's (naïve) Take on MBSE. Submitted to the 22nd Annual International Symposium of the INCOSE, Rome, ITALY (2012)

Estefan, J.A.: Survey of Model-Based Systems Engineering Methodologies (MBSE) - Rev. B. In: International Council on Systems Engineering. INCOSE, Seattle (2008)

Friedenthal, S., Moore, A., Steiner, R.: A Practical Guide to SysML- The Systems Modeling Language. Morgan Kaufmann OMG Press (2008)

ANSI/IEEE (American National Standards Institute / Institute of Electrical and Electronic Engineers), ANSI/IEEE 1471-2000. IEEE Recommended Practice for Architectural Description of Software-Intensive Systems (2000)

Elm, J.: A Study of Systems Engineering Effectiveness: Building a Business Case for SE. In: Proceedings of the 21st annual International Symposium of the INCOSE, Denver, CO. INCOSE, San Diego (2011)

Honour, E.: Systems Engineering Return on Investment. In: Proceedings of the 20th annual International Symposium of the INCOSE, Chicago, IL. INCOSE, Chicago (2010)

Honour, E.: Sizing Systems Engineering Activities to Optimize Return on Investment. In: Proceedings of the 21st annual International Symposium of the INCOSE, Denver, CO. INCOSE, San Diego (2011)

Doufene, A., Chalé Gongora, H.G., Krob, D.: A Commonsense-Driven Architecture Framework. Part 3: Extension to Multi-Objective Optimization - A case study for Electric Vehicle Powertrain. Submitted to the 22nd Annual International Symposium of the INCOSE, Rome, Italy (2012)

Krob, D.: Eléments d'architecture des systèmes complexes. In: Appriou, A. (ed.) Gestion de la Complexité et de l'information Dans les Grands Systèmes Critiques, pp. 179–207. CNRS Editions, Paris (2009)

Krob, D.: Enterprise Architecture, Modules 1-10". Internal Communication, Ecole Polytechnique, Palaiseau, France (2009-2010)

Penalva, J.M.: La modélisation par les systèmes en situations complexes. PhD Thesis, Université de Paris 11, Orsay, France (1997)

OMG (Object Management Group), OMG Systems Modeling Language (OMG SysML^{TM}), Needham, MA, US (2010), `http://www.omg.org/spec/SysML/1.2`

Wang, R., Dagli, C.H.: Executable System Architecting Using Systems Modeling Language in Conjunction with Colored Petri Nets in a Model-Driven Systems Development Process. Journal of Systems Engineering 14(4), 383–409 (2011)

Watkins, C.B., Dagli, C.H.: Understanding the Implementation of System Architectures in the Context of Distributed Cognition. In: Proceedings of the 20th Annual International Symposium of the INCOSE, Chicago, IL, INCOSE. San Diego (2010)

Chapter 17
Human Emotional Interaction and Hydrotherpy: Industrial Design Keys

Sergio Gago and Joaquim Lloveras

Abstract. This article focuses on bathroom design and more specifically on the sub-sector commonly known as “wellness”. This sector is characterized by its essential element, water. It is possible to generate several types of sensations through water by utilizing several variables such as composition, temperature, and pressure. The leading companies in this sector are allocating more and more resources in the design phase of development. These products need to meet a more demanding market, not only in regards to the product, but also the environment.

This is a growing and complex market because its purpose is not just selling a physical product, but also the types of sensations and emotions that are generated. This set of sensations should result in positive emotions for the user. In short, the aim is to sell "wellness". The sensory design plays a key role in this process. It’s also essential to consider the product aesthetics, both the presence aesthetic and the aesthetic in human interaction.

A critical issue in today’s market place is how to obtain the feedback from users. This feedback contributes vital information that is essential in order to improve the features of design for each product. This information can bridge communication barriers between the end user and the product design team: retail, sales department, technical after sales .. each of these units filters the user feedback with a self-interest that lead to the discovery of concrete needs. Therefore providing specific designs that could be vastly improved in terms of interaction and emotional design.

The study of sensory characteristics and user feedback for current products create the basis for future ¨wellness¨ designs. The interaction of users with these

Joaquim Lloveras
Technical University of Catalonia (UPC)
C/ JORDI GIRONA, 31 08034 BARCELONA
e-mail: j.lloveras@upc.edu

Sergio Gago
Departament de Projectes d'Enginyeria
Universitat Politècnica de Catalunya (UPC)
Escola Tecnica Superior d'Enginyeria Industrial de Barcelona (ETSEIB)
Av. Diagonal, 647. Planta 10. 08028 Barcelona
e-mail: sergio.gago@estudiant.upc.edu

products and their experience strongly influence research and development. The wellness products are designed to improve the physical and mental health of users through hydrotherapy. By creating more effective sensory and emotional relationships, not only will we achieve greater success in the market, but also enhance the quality of life for the users as well.

Keywords: design, emotion, wellness, hydrotherapy, interaction.

1 Introduction

It is very important to understand that aesthetics, interaction and the way that we perceive the products can deeply affect the user's emotions. The world, the environment and even everyday objects can be perceived in a distinctive way, making people change their way of doing things [1]. The material world that surrounds us affects our emotions. By understanding these emotions and the interaction with the products, the users' quality of life can be improved by engineers and designers, making the designs better without having to increase the cost [2].

Both, current products and new products that are being designed should be analyzed in market, functional, and ergonomic studies in order to obtain more information. The goal is to improve how the products are developed and designed without increasing cost and still make an impact in the market.

This article is focused on studying the products that represent the wellness sector. The actual design of these kinds of products doesn´t always involve studying the deep emotional values that people share with these products. Understanding these emotional factors is essential in order to have success and a significant share of the market.

In addition, we want to ensure these products have a positive social impact. Very often these important factors are overlooked because this is the world of artists and not engineers. Although these kinds of products should center on the user and their feelings, the actual industry is focused on functionality and not interaction. There are times when the product ¨feels good¨ emotionally and isn´t practical. In addition, most industrial companies do not have an effective way of communication between the design team and the final user. The real emotional effect of the end user is never communicated to the designers. Therefore many of the design specifications don't take into account the real customer feedback. People are complex and complicated. They are not another machine that will operate as programmed. Some people will appreciate the functionality of the design and others will feel that it is impersonal [3]. The aim is to obtain the important information from customers to merge these two worlds and create products that can appeal to people both on a functional level as well as aesthetically.

2 Aims of the Study

This article attempts to identify characteristics that determine new areas for improvement in the design process. These improvements are based on the users' reactions and needs as a result of their interaction with the product. The main objective is to understand human emotional responses when designing these products. With this goal in mind, we are studying products that have a special interest in stimulating the senses.

In addition, it’s very important to analyze the positive and negative characteristics of current products in terms of emotional design through feedback from users. The purpose is to identify action points to substantially improve the interaction between people and these products. Through an online system the end users can easily access they can communicate their opinion a satisfaction with the product. This system makes it possible for users to reach a set of key questions that make them to think about what feelings the product is creating in them. The goal is to get feedback from the sensory and emotional response from the user’s experience. Through this feedback we can obtain new information directly from the user. This feedback will contain specific information and could be analyzed by the designer to get a new perspective on the specifications which are more focused on the feelings and users' emotions and opinions.

3 Wellness Products Studied

We are studying the leading products in the wellness sector. The goal is to look at the products that interact with the user at different points in the emotional experience. We also need to understand the different techniques and technologies involved with this interaction. These products are also more complex to design, since they offer more features than others in the sector. We can divide these types of products into two categories: the spa and the shower.

Depending on the type of product, the user can enjoy many different functions. Among these, we highlight the hot tub, aromatherapy and / or chromotherapy, music and waterfalls [4].

3.1 The SPA

The current concept of the Spa in product engineering refers to a type of pool or bathtub, according to size, equipped with different nozzles for hydromassage (water and air). More sophisticated models also have chromotherapy and systems that produce a variety of fragrances. One can also find products with integrated Hi-Fi audio equipment.

The spa industry has experienced rapid growth in recent years. People who purchase a spa are looking for, above all; pleasure. The ability to affect emotions while stimulating the senses plays an important role in the overall user experience. The combination of these factors is the key to building lasting relationships with users. Products that simulate the experience of visiting a wellness center have grown tremendously in the past several years. This has led to a surge in the range of commercially home spa products available in the market today. The figure 1 shows an example of a home spa.

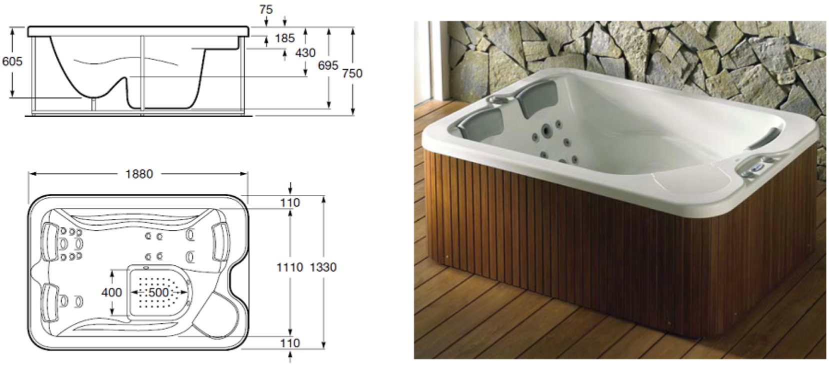


Fig. 1 Example of general measures of a home spa

3.2 The Shower

The shower is defined as a type of bath in which the water falls on the subject who is standing upright. There is no accumulation of water as it is directed to the drain. Jets and water fountains are usually fixed in the cabin above the user and also arranged vertically in order to massage the subject from different perspectives.

Showering is part of a daily routine. The main objectives are to promote cleanliness and prevent odors, diseases and infections. Advances in science and medicine in the nineteenth century helped society to realize the benefit of regular bathing and how it affected people’s health. As a result, all modern cultures encourage a daily regimen of body hygiene. The wellness industry has transformed

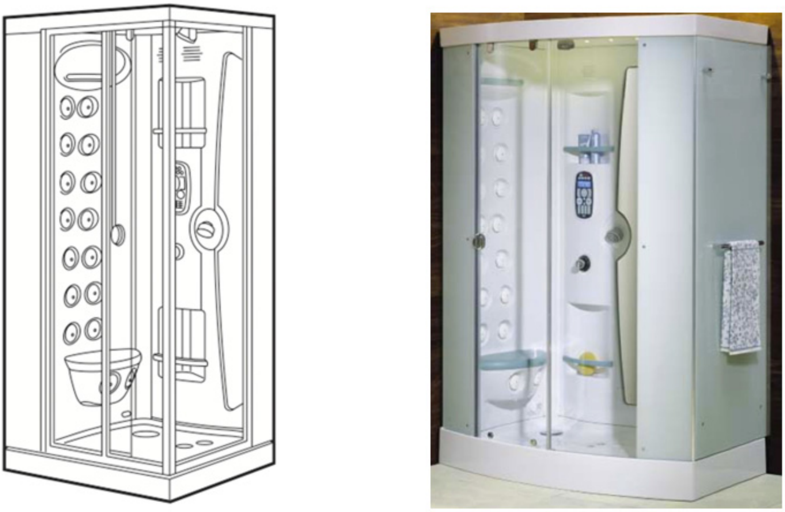


Fig. 2 Example of shower cabin with digital control for their functions

the classic concept of showering into a relaxing and therapeutic activity as well. Thus the concept of "cabin shower" was created. This includes a number of features added to the classic shower design, and has become an integral part of most people's lives. One of the latest advances in cabin shower design (see figure 2) is the use of specific computers which open new possibilities to stimulate the senses and user's pleasure.

4 Sensorial Study

Wellness products studied in this article have more intimate contact with the user than their peers. Due to this intimate contact, sensory design plays a major role in designing these products. The design should focus on the many variables that will affect the users' sensory perception. These variables are important because they directly impact the user's emotions and their feelings about the product [5]. The sensory attributes can be found in both the product and the user.

4.1 Sensory Attributes of the Product

Wellness products are composed of several elements. The basic element of communication is the facade, ie what the user can perceive through sight and his physical interaction with the product. In most cases, this part contains the control interface. It is important to note that the facade of the product is the first contact with the user and, therefore, creates the first impression [6]. The elements that a user is able to perceive through his senses define the set of sensory characteristics and special attributes for each product.

4.2 Sensorial Perception of the User

Sensory values attributable to a product may be perceived by the user in two ways:

- Objective way (physical): we are talking about physical and precise attributes which do not require a high level of interpretation. For example the color, texture, noise, etc...
- Subjective way (psychological): We are talking about all those perceptions or feelings that are related to social, cultural and personal experience from the past. These intrinsically have a deeper value for the user.

4.3 Users' Physical Perception

The surface elements of the product are perceived mainly by sight and touch, but there are a number of other elements that can be perceived by other senses. In wellness products much of the importance is focused on the senses of touch, scent and hearing. The goal is that the user will be able to receive the message of harmony and well-being through all their senses.

4.4 Users' Psychological Perception

In this section we reflect on those attributes that are more subjective and interpretive. We are talking about affective characteristics, symbolic, cultural and those characteristics that get the user to emotionally bond with the product.

The psychological perception of the user is determined by those human characteristics that satisfied or not by sensory attributes; still affect a user's emotional tie to the product [5]. This is the reason it is important to analyze the specific sensory characteristics of the product. These sensory characteristics should be a kind of language that communicates with each user, trying to transmit the information and feelings that the user needs every moment. If we could quantify the effect of these attributes, it would be possible to design products that could communicate the feeling or the particular effect desired by the user [7]. However, this effect can only be offered through the physical stimulation of the user's sensory attributes: tactile, visual, olfactory and auditory senses.

We could classify the sensory elements attributable to a wellness product through the figure 3. Furthermore, it is illustrated in table 1 with examples of pleasant emotions.

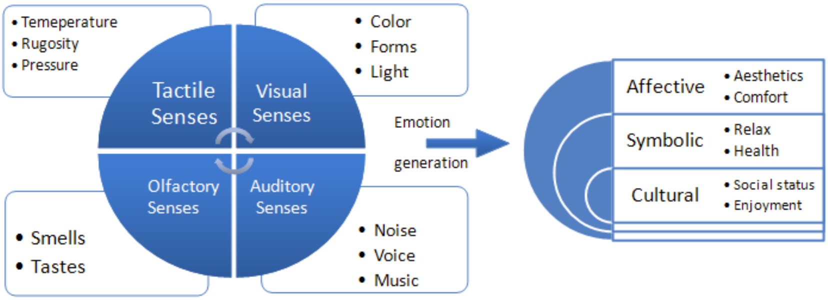


Fig. 3 Users' physical perception and emotion generation on wellness

Table 1 Type of pleasantness and examples for emotion

Type of pleasantness	Example of happiness/joy
Visual pleasantness	A beautiful external form or color combination
Auditory pleasantness	Silence and calm song while using functions
Olfactory pleasantness	Uses of aromatic fragrance o soaps in water
Tactile pleasantness	Pleasant bubbly feeling of an air jet under water

5 Research Methodology and Data Collection from Users

Part of the research involves an important data collection task consisting of a set of online semi-structured interviews with the people who have purchased a wellness product, followed by an analysis of the interview data related to the product features. A set of questions was designed to create interactive feedback from the users. Questions tackled the following themes that corresponded to the purpose of the study:

- The overall satisfaction about the product and about each of its functions;
- The relevance of these functions for the user;
- The main reasons for acquisition;
- The effect caused by product attributes;
- A final open field for suggestions for improving the product.

The interviewees were shared with the most prominent retail Spanish companies in the wellness industry between March 2011 and February 2012. We estimated the number of products sold during a year for these companies in Spain, which resulted in a total of 6,600 products for 2011 (6,000 from shower cabins and 600 from home spas). From these purchases we were able to interview a total of 327 users. About 20% of these interviewees were users of shower cabins, 50% from home spas and the remaining percentage used both products. This sample contains people of different features. We only interviewed consumers who were users of these kind products for more than 6 months.

Once the data was collected, it was analyzed according to the type of product acquired: spa or shower. This study aims to identify those sensory characteristics that are important to the user and which are not as well as product features that do not fulfill the user's wishes. The goal is to find the most important characteristics or functions that enable users to enjoy a more pleasant interaction with the product.

This kind of information should create new ideas and provide ways to innovate the design process. The idea is to be able to apply the results to the specifics products and not only to come up new improvements on design for a specific product, but to understand the way of thinking of most of users and their needs and wishes about interaction and emotion with these kind of products. In this way, we should be able to establish a general guide for the designers about the emotional design of wellness.

6 Calculation Parameters for the Result Analysis

The analysis and choice of each of the points to consider in the interviews with users has been the key to obtaining satisfaction values that are indirectly related to some of the parameters involved in designing the product. One of the most difficult tasks from this study was to trying to calculate a value that defines the effectiveness of providing satisfaction to users [8]. Given the complexity of wellness products studied, it was necessary to guide users from functions and general considerations to specific design parameters.

The first step is to assign a value to the main users' wishes who want to be satisfied. We group these desires into 4 general families in table 2.

Table 2 Relevance of the users' desires

Desire	Relevance -Spa users (%)	Relevance - S.C. users (%)
Relaxation / comfort / enjoyment	87	62
Health / Hydrotherapy	61	78
Product Aesthetics	58	62
Social recognition	36	26

The second step is to detect and study the main points of dissatisfaction. These points should be weighted according to the importance of the family they belong to, which was calculated in the first stage. To elaborate this weighting must also take into account the characteristics of users and the type and model of product on which the experience is based. The points raised can be grouped into the areas of study in table 3.

Table 3 Relevance of the areas of study

Areas of study	Relevance - Spa users (%)	Relevance - S.C. users (%)
Effectiveness (Functionality)	96	93
Ecological	46	75
Intuitive	59	82
Comfortable / ergonomic / accessible	87	62
Value for money	67	86

The study of all the parameters involved in stages 1 and 2 allowed us to detect major points of action. However, we need to go beyond these parameters to analyze and identify the physical characteristics involved in the possible improvement of sensorial stimulation [9]. In this case, the physical characteristics that have been addressed in the study are showed in table 4.

Table 4 Feature parameters and studies involved

Feature parameters	Studies involved
Water temperature	Effectiveness, comfortable, hydrotherapy, enjoyment
Water pressure	Effectiveness, comfortable, hydrotherapy, enjoyment
Rugosity of materials	Effectiveness, comfortable, accessible
Smells / tastes	Effectiveness, comfortable, enjoyment, value for money
Color / shape	Intuitive, comfortable, ergonomic, accessible
Lighting system	Effectiveness, comfortable, intuitive, enjoyment, value for money
Noise	Comfortable, intuitive, value for money
Music	Effectiveness, comfortable, intuitive, enjoyment, value for money

These final values will have a specific weight depending on the product type and model referred to. Therefore, as this the last step, the appropriate parameters should be chosen to compare them with the functions that are impeding the user's satisfaction. This task is one of the most complex and one of the most important as well. The ratio of sensitive parameters to improve the product functions requires a deep understanding of the product, such us features offered, how they are designed, functional limitations, range, etc.

Obtaining the final conclusions requires, not only precise calculations of the data regarding user's feedback but also interpreting the values with the designers team to identify areas for improvement and its feasibility, economic and technological. We could illustrate an example of the steps of this process through the figure 4.

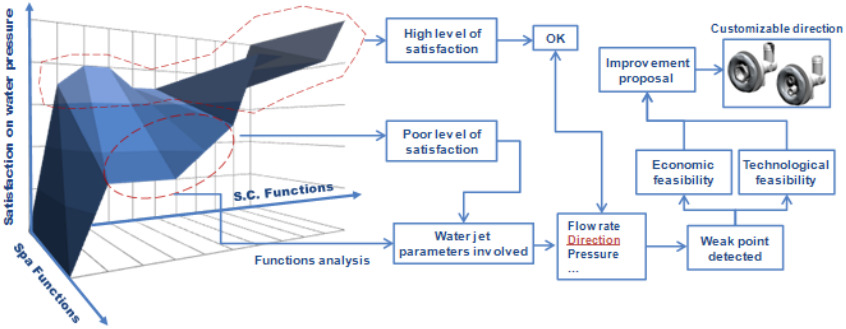


Fig. 4 Connection between sensorial characteristics and designing parameters

7 Weak Design Points Detected

Both the current market research and product feedback from users, could determine most points of design deficiencies. These issues directly affect the user experience. A number of considerations as a result of this study are shown below. These considerations have been reported and analyzed with design managers from leading companies involved and should be the basis for improving the existing designs of these types of products.

7.1 Control Interface

Products exhibit poor quality in regards to the control interface, when compared with the other components of the product. Only very high-end spas and showers feature a tactile interface with clear and total control over how the spa or shower functions. In addition, most users do not believe the design or the location of the control interface is convenient or intuitive. The interface must be located in an area in which the user can interact with the device without having to move their position.

7.2 Customizing the Pressure and Position of the Water Jets

Users find the hydromassage function one of the more important features offered especially the position and pressure of the water in relation to the body. Weak or incorrect regulation of these two variables, position and pressure, could result in an injury or simply no pleasure for the user. We propose developing specific software that could calculate the optimal value for these factors in the hydromassage. At least, these values could be used as a basis for the user's ability to adjust these parameters as desired. The basic information inputs that the user should enter to the control interface would be age, weight, height and physical ailments. The software could easily calculate, according to user characteristics, the position and intensity of application of the water outlet. The goal of hydromassage is to soothe the user's points of discomfort. The correct location of trigger points is a key for hydromassage. By applying constant pressure at a certain temperature at these points, we could alleviate bodily ailments effectively. The Figure 5 shows the 14 major trigger points considered in massage treatments. [10]

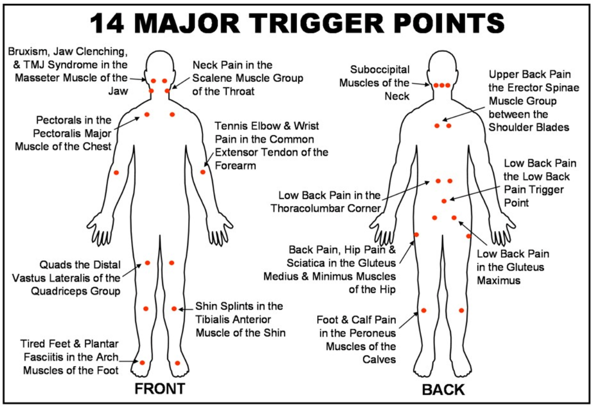

14 MAJOR TRIGGER POINTS
Bruxism, Jaw Clenching, & TMJ Syndrome in the Masseter Muscle of the Jaw
Neck Pain in the Scalene Muscle Group of the Throat
Pectorals in the Pectoralis Major Muscle of the Chest
Tennis Elbow & Wrist Pain in the Common Extensor Tendon of the Forearm
Quads the Distal Vastus Lateralis of the Quadriceps Group
Shin Splints in the Tibialis Anterior Muscle of the Shin
Tired Feet & Plantar Fasciitis in the Arch Muscles of the Foot
FRONT
Suboccipital Muscles of the Neck
Upper Back Pain the Erector Spinae Muscle Group between the Shoulder Blades
Low Back Pain the Low Back Pain Trigger Point
Low Back Pain in the Thoracolumbar Corner
Back Pain, Hip Pain & Sciatica in the Gluteus Medius & Minimus Muscles of the Hip
Low Back Pain in the Gluteus Maximus
Foot & Calf Pain in the Peroneus Muscles of the Calves
BACK

Fig. 5 Position of the water jets to cover major trigger points

7.3 Thermal Sensation

One should differentiate between general temperature and the temperature that contacts the user's body. General temperature is calculated theoretically and it is intended to have a value close to user's body temperature. It is important that the first contact with the user with water isn't unpleasant. This temperature cannot be calculated in a theoretical way. We need to ask the user about their sensation in order to identify what is too cold or too hot.

- Impact on the shower cabin: In terms of engineering, we should determine how to eliminate the water in the cabin installation (pumps and pipes) after the last use. The temperature of standing water cannot be controlled and it usually results in a very unpleasant sensation to the user at the start of the next use.

- Impact on the spa: The thermal sensation of the most users when they are entering a spa is usually the sensation that it is too cold. In some cases the opposite has been reported, ie the user senses a sensation that it is too hot. To avoid this negative response in users, we should customize an initial temperature for each individual user based on their use and, gradually, the system could identify the ideal target temperature for the user.

7.4 Floating and Avoiding Water in User's Facial Area

We should think about how unpleasant water can be in the users' airways and eyes, especially spas using salt water. That water may make the user's breathing difficult, destroying any sense of wellbeing that had been created. The water could also produce other adverse effects such as the horizontal displacement of the user, because water currents can push the user from one side to another.

We propose two new aspects in the spa design. One major change should be the intensity of the upper jets of the spa. They should be regulated independently of the others, avoiding excessive bubbling in areas close to the user's face. Furthermore, the existing headrest cushion should be redesigned. These pads not only have to offer the support to various points but should also allow users to position their body so they are not displaced by the water.

7.5 Environmental Awareness

Bath industry leaders have become increasingly more responsible about the environment and water use. That's why the design trend opts for water treatment and recycling existing after supplies in wellness products. Many big spas work as pools so, after being filled with water, it is not necessary to waste more water. We can reuse it several times, depending on the length of use and other additional concerns.

The operation of shower cabins is another area that should be rethought when thinking about water reuse. We should differentiate between the two stages of use. The first phase should be considered as cleaning and thermal preparation for the user's body in order to start the second phase. This second phase should allow the user to enjoy all functions that the cabin provides without worrying about water consumption, since in this phase the water could be reused through a recirculation loop and aided by pumps.

7.6 Time Control

The use of any product to relax should provide automatic control of time. This means that the product should warn the user when the desired time is close to finishing. This time should be configured before the product use and the warning should be done in an unobtrusive way.

7.7 Ergonomics

The current aesthetic in design tends to be straight lines. This is contradictory to the shape of the human body. It doesn't make sense to create attractive forms for the user if that results in discomfort for them. We should differentiate between the two in design; the external part that could be designed more with visual aesthetics in mind and the internal part that could be adapted to be more functional while thinking about accommodating the user's body in an ergonomic design.

7.8 Ambient Noise

The noise created by hydromassage products is a result of the vibration of the pumps. This noise has a much greater effect on the spa, since the supply of water in the shower cabin is exterior and the falling water produces a sound much greater than water flowing through the pipes. Therefore, it is important to concentrate on improving the acoustic insulation of the spa so it does not produce noise that could interfere with the process of relaxation.

7.9 Accessing the SPA

The entry and exit from the spa are responsible of many falls and injuries in bathroom. This is especially true for advanced age users who have been exposed to a prolonged use. Research has discovered that a large number of products are unsafe to enter or exit. We should consider redesigning them, regardless of size of the product, to address this problem. In addition, we should design a fixed structure to help support the user while they are entering or leaving the spa.

8 Limitations

The findings presented in this paper originate from data that was collected from a small group of users who bought a product in the Spanish retail market from popular companies in the wellness industry. Therefore, it would be a stretch to claim that the answers collected can be translated on a greater scale. Moreover, there is always the possibility that the participants may not have answered truthfully or are representative of users from others countries or cultures.

For the improvement of the external validity of this study, more groups of users from different countries need to be interviewed. Given that this study is a first step towards developing a guide for the emotional design of wellness products, we believe that the results are valuable and are give a clear direction for future studies.

9 Discussion

Clear deficiencies have been identified that cause an unpleasant users experience while we were looking for design aspects that could provide an improved user experience. It is important to pay attention to these deficiencies, since every adverse sensation persists in users' memory and could easily override any positive feelings and result in the user’s rejection of the product. This concept is represented through a connected-gears system as is shown in the Figure 6. One can see how locking any gear box prevents movement of the rest causing the whole system to fail.

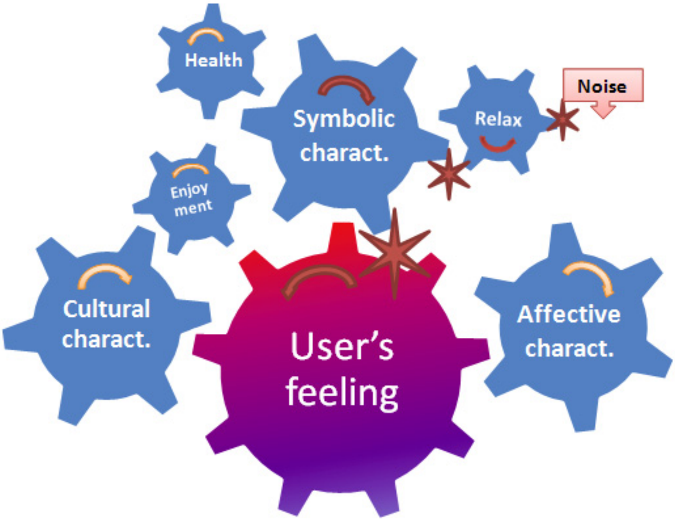


Fig. 6 Representative diagram of user’s product rejection

Users of wellness products are looking for satisfaction or relief, ie pleasure. This sense of well being can be classified as the satisfaction of two general needs: physical treatment of bodily ailments and alleviating psychological stress.

Physical treatment should be applied through the physical sensory attributes of the user; visual, tactile, auditory and olfactory. In addition, the user's feelings need to be satisfied through the physical sensory attributes. These sensory attributes affect the user's emotional bond with product. This is why it essential to analyze these sensory needs in an effective way. The sensory characteristics should be a kind of language through which the product can communicate with each user, trying to transmit the desired information and feeling of every moment. This results in psychological treatment as well as physical. This treatment could offer enjoyable sensations and help relieve stress.

If we are able to quantify the effect that causes each sensory attribute, we could clearly identify those design aspects that are vital to creating an emotional bond with the user. That is the best way to improve the user experience.

10 Conclusions

This research demonstrates the importance of getting feedback from users and making adjustments to the design. This information is needed to avoid those characteristics that encourage users to reject the product. This product rejection occurs when even one aesthetic or negative interaction characteristic is sufficiently bad for the user, regardless of how good everything else may be. So it is very important to detect what users want and need through their feedback as well as those product characteristics that we can correct or improve.

In addition, the meetings with several departments of the leading Spanish companies in the wellness sector show that, in most cases, an effective and direct way of communication between "designers" and "final users" does not exist. They seem two different and distance worlds which are trying to converge, but they do not succeed completely in doing it.

References

[1] Norman, D.A.: Emotional Design: People and Things, jnd.org (2004), http://www.jnd.org/dn.mss/emotional_desig.html

[2] Norman, D.A.: Emotion & design: attractive things work better, jnd.org (2005)

[3] Heller, D.: Aesthetic and Interaction Design – Some Preliminary Thoughts. Interactions 12, 48–50 (2005)

[4] Wellnessbyroca web site (March 28, 2011), http://www.wellnessbyroca.com

[5] Bedoya, D.: Diseño sensorial. Las nuevas pautas para la innovación, especialización y personalización del producto. Thesis doctoral, UPC (2007)

[6] Desmet, P., Hekkert, P.: Framework of product experience. International Journal of Design 1(1), 57–66 (2007)

[7] Desmet, P., Overbeeke, K., Tax, S.: Designing Products with Added Emotional Value: Development and Application of an Approach for Research Through

[8] Hassenzahl, M.: Emotions can be quite ephemeral. We cannot design them. Interactions 11, 46–48 (2004)

[9] Desmet, D.: A Basic Typology of Product Emotions, City (2004)

[10] Medicinenet web site (Febreuary 26, 2012), http://www.medicinenet.com/fibromyalgia_pictures_slideshow/article.htm

Chapter 18
Synchronisation Phenomena in Electrical Systems: Emergent Oscillation in a Refrigerator Population

Enrique Kremers, José María González de Durana,
Oscar Barambones, and André Lachaud

Abstract. Under-frequency load shedding (UFLS) is a technique in which the power grid frequency is used as an indicator to shed down electrical loads at critical periods, which can increase the resilience of the system. We present a multi-scale agent-based model to simulate the impact of UFLS on an energy system. The model allows a representation from a complex system point of view, allowing for cross-scale interactions which is not evident in standard grid simulations. Each refrigerator is modelled individually but the population has a heterogeneous configuration. The refrigerators are coupled to a simplified energy system model to represent the grid frequency. Using a simple UFLS strategy we discovered synchronisation effects among the refrigerator population. An emergent oscillation which would be fatal for the system was detected. The model allows to explore the origin of this phenomenon as well as to determine the phase transition which occurs in the system.

1 Introduction

Interconnected continental power grids such as the European or the North-American power grid offer a large degree of resilience due to their extension and large amount

Enrique Kremers
European Institute for Energy Research (Electricité de France & Karlsruhe Institute of Technology), Emmy-Noether-Strasse 11, 76131 Karlsruhe, Germany
e-mail: kremers@eifer.org

José María González de Durana · Oscar Barambones
E.U. de Ingeniería de Vitoria, Universidad del País Vasco, Nieves Cano 12,
01006 Vitoria, Spain
e-mail: jtpgogaj@ehu.es

André Lachaud
Laboratoire CEDRIC du CNAM, 292 rue Saint-Martin, 75003 Paris, France
e-mail: alac@free.fr

of control mechanisms. The frequency of these grids is usually largely stable and shows only small variations during regular operation. On island systems however, the stability is drastically reduced. Small, self-sufficient electrical systems have been traditionally supplied with high reactive, rather small or medium sized production units, frequency instabilities are a common issue. A support of the spinning reserve by demand side management (DSM) could help to improve the stability of the system.

1.1 The Demand Side as a Support for Frequency Stability

In the upcoming, smart electrical system, an active role of the demand side is being discussed. By regulating demand within certain limits, not only production is being governed. This allows a more flexible balancing of the system.

These mechanism can support the integration of fluctuating renewable energy sources such as wind power [9]. Further, the grif frequency can be stabilised by reactive load in- or decreases, allows to support the primary or spinning reserve, which is usually handled by the production side only. The frequency as an instantaneous indicator of the balance between production and demand allows to quickly react to unplanned imbalances of the grid.

1.2 UFLS Based on Local Refrigerator Management

In order to support the primary reserve, a highly reactive demand response is needed. As soon as an abnormal frequency drift is detected, the demand side can react by adding or reducing the power consumption. In contrast to classical reserve mechanisms, usually when talking about the demand side we are dealing with a large number of small consumers, rather than with a big dispatchable power plant. In this example we will choose refrigerators, as they are appliances with a very high penetration rates and, even without having high peak powers, it is usually continuously plugged to the grid. Refrigerators offer the possibility to operate as a thermal storage medium, and this can help to respond in a flexible way to the demand of the energy system.

Only a few simulations on individual based models of large populations can be found on refrigerators. Some approaches on demand side management concerning refrigerators have been proposed during the last years. Notably [12] uses a simplified refrigerator model in order to simulate a large number of them, proposing a dynamic operation strategy by varying the thermostat threshold linearly the grid frequency. However, in any of these approaches, the effect of the different demand side management strategies at system level and with a large penetration has been analysed through a detailed model. Only a few high resolution studies and models exist on domestic demand [17, 15], and they are not coupled to an energy system model. In the following, a model which aims to represent a large population of frequency controlled refrigerator is presented, which shows unexpected impacts on the system [5].

2 Modelling a Refrigerator

The refrigerator is a device based on a compressor which works cyclically and cools down the inner cell which keeps the content at a fresh temperature. The model of the refrigerator agent is shown in Figure 1.

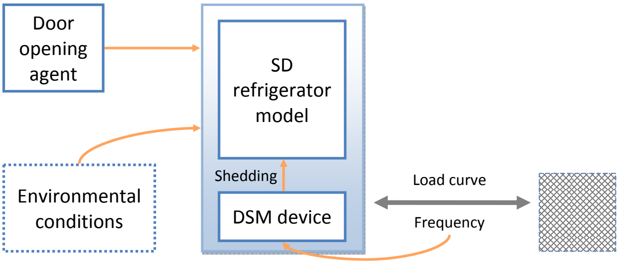


Fig. 1 Refrigerator model structure: The refrigerator is coupled to a demand side management (DSM) device, which is connected to the grid and can sense the frequency. Further, the refrigerator has heat exchanges with the environment, and a door opening agent adds *human behaviour*.

The refrigerator model used in this study is based on the heat fluxes of its internal cell, the evaporator and the exterior. These fluxes are described by a set of differential equations which characterise the heat transfer processes. The principles of the models are describe in [3]. This model was extended with door opening and the effects of the content. Door openings have a large impact on the internal air temperature, especially when the refrigerator is situated in a warm environment. The food and drinks charged in the refrigerator also are relevant for the dynamics and inertia of the inner cell temperature, functioning as a heat storage. The evolution of the internal temperature is described by a differential equation that takes into account the heat transfer process between the external environment and the refrigerator cell and the cooling contributions by the evaporator plate, as well as direct air exchanges due to door openings and heat exchanges with the charge:

$$\frac{dT_i}{dt} = \frac{T_e - T_i}{\tau_{e-i}} + \frac{T_p - T_i}{\tau_{p-i}} + \frac{T_l - T_i}{\tau_{l-i}} + \frac{\delta_d (T_e - T_i)}{\tau_d} \tag{1}$$

where T_i is the internal cell temperature, T_e the temperature of the environment, T_p the temperature of the evaporator plate. τ_{e-i}, τ_{p-i} and τ_{l-i} are time constants. T_l is the average temperature of the assumed content in the refrigerator. The opening of the refrigerator door creates an immediate heat exchange between the environment air temperature and the interior. The term is characterised by δ_d which expresses the heat exchange due to door openings. $\delta_d = 1$ only if the door is open; if it is closed it equals zero. The time constant τ_d represents the heat transfer by air exchange between the environment and the inner cell.

The variation of the temperature of the evaporator plate is obtained by

$$\frac{dT_p}{dt} = \frac{T_i - T_p}{\tau_{i-p}} - \frac{\delta_c Q_f + Q_{ext}}{C_p} \tag{2}$$

where τ_{i-p} is a time constant and Q_f is the cooling capacity of the plate, which is triggered by the switching control variable δ_c. C_p is the thermal heat capacity of the evaporator fluid and Q_{ext} an exponential function of time which describes the effect of released thermal power after the compressor switches off and is explained in detail in [3].

The thermostat controlling the compressor turns it on when the temperature drops above $T_{p,max}$, and doesn't turn it off until the temperature falls below $T_{p,min}$. Hence, the thermostat is a device that is modeled as a state machine with two states, `off` and `on`.

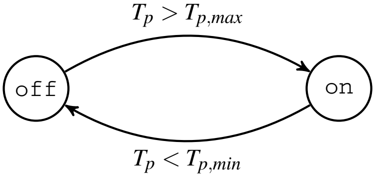


Fig. 2 Thermostat state machine

Further, the effect of the content (food or beverages) in the cell is described by a heat transfer process between the internal cell temperature and the temperature of the content T_l:

$$\frac{dT_l}{dt} = \frac{T_i - T_l}{\tau_{i-l}} \tag{3}$$

where τ_{i-l} is a time constant.

2.1 Inclusion of Social Behaviour through Door Opening

In order to recreate the uncertainties of door openings, a stochastic model was included to represent the human behaviour causing door openings. This door opening agent is able to control the refrigerator door and is embedded in the refrigerator. The model is based on a discrete event state-chart, which triggers δ_d for the door heat transfer process in Equation 1.

The agent model represent a random opening of the door distributed exponentially over the day, which is characterised by an an average number of door openings per day n. The duration of an opening is distributed normally with a mean time $\mu_{doDuration}$ and the standard deviation $\sigma_{doDuration}$ of the door opening.

The values chosen by default were taken from a survey study [7]. An average of $n = 15$ openings per day was taken, with an average duration of $\mu_{doDuration} = 20$ s and a standard deviation of $\sigma_{doDuration} = 50$ s.

3 Implementation of the Model through an Agent-Based Approach

We use an agent-based model [16] in order to represent this complex system, in which we aim for an individual modelling of a large population of refrigerators, based on the model described above. An advantage of agent-based modelling is that it can be easily combined with other approaches, which are used to describe the behaviour of the agents. So, we introduce a classical differential equation to reproduce the behaviour over time of an agent.

The model is implemented in Anylogic, a multi-approach simulator. To represent the differential equations concerning the refrigerator behaviour, the System Dynamics (SD) notation is used. In [14] it is described how SD can be used to model classical electromechanical systems which are continuous models based on ordinary differential equations (ODE). Anylogic includes a very reliable hybrid solver [2] which is able to combine these continuous systems with discrete models, which create discontinuities in solving of ODEs over time.

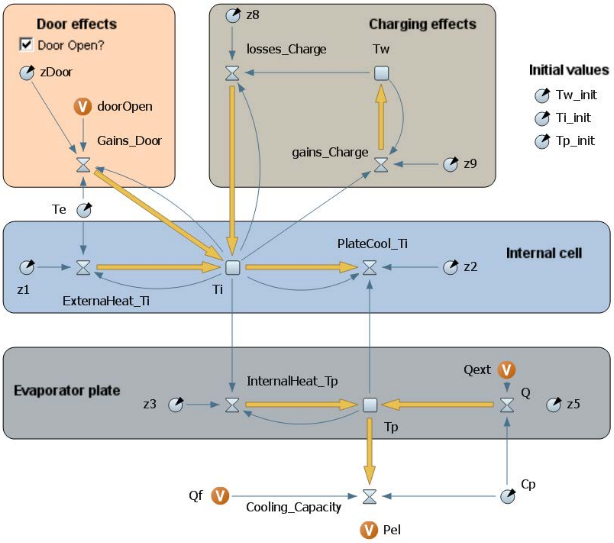


Fig. 3 Refrigerator model implemented in system dynamcics. The different parts of the system can be recognized, as well as the flows among them. Each stock (square) is defined by one of the differential equations given above.

Figure 3 shows the refrigerator model describe above in the SD notation, showing clearly the different heat flows and losses of the system. The refrigerator parameters were calibrated using the OptQuest optimising engine based on measurements realised on a real refrigerator. This allowed us to verify the validity of the model.

4 A Multi-agent Simulation of a Population

Multi-agent simulations consist of many interacting agents. Considering the fact that the electrical system is composed of many heterogeneous entities, a multi-agent approach seems reasonable. To represent this heterogeneity, each agent can be parametrised in a different way, and so reflect a variety of agents among the population. Each agent is modelled individually, thus no averaged or higher level model is used. This requires on the one hand a greater computational power, but on the other allows high resolution models of the system, where individual values of each entity can be recovered, to be analysed individually or statistically over the whole or parts of the population. In this case, the refrigerator agents will be replicated and interact with an energy system model, which allows representing the grid frequency.

When individual loads are added, they can coincide in different degrees. If many of the loads have their peak at the same time, this will be reflected on the aggregated load curve. If the peaks are distributed and do not coincide, the load curve is smoother. The coincidence of the loads at a given time t is $c(t) = \frac{P(t)}{P_{nom}}$, where P is the instantaneous load and P_{nom} the total installed load (maximum possible load if all individual loads would coincide).

5 A Simple Energy System Model for the Grid Frequency

To model the response of the frequency of the system to load changes, we use the System Frequency Response (SFR) model [1]. It is a simplified model which describes the behaviour of an interconnected system when dealing with a large disturbance. The SFR model allows to understand how the system parameters affect the frequency response. In a detailed model of the electrical system, this is more difficult because frequency response there is a very complex function of many system variables. The model omits many details, providing an approximation the system frequency performance which has been used by different authors in the field [8, 4]. The parameters of the SFR models were set to typical values shown in [1].

Based on the refrigerator population of refrigerators described we create an integral model (Figure 4) coupling the population to the SFR model, which is implemented also in SD. A simplified behaviour of an energy system was considered. The total production of the system was assumed to be constant, as well as the rest of the demand. In fact, the only powers varying in this test system are the refrigerators. This simplification allows us to analyse the direct impact of a refrigerator population on the frequency.

The disturbance power is defined by $P_d(t) = P_g(t) - P_o(t) - s_f \cdot P_f(f)$, where P_g is the total generated power and P_o the consumption of all other loads and P_f the refrigerator load and s_f a scaling factor. A positive disturbance power therefore means an over-generation, and a negative disturbance a over-load of the system.

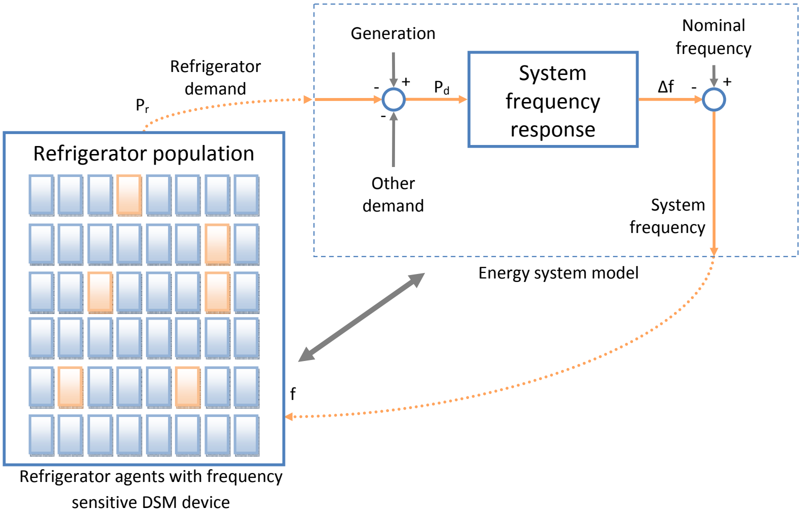


Fig. 4 Integral system model: the multi-agent model of the refrigerator population interacts through power and frequency with the simplified energy system model. The aggregated load curve of the refrigerators affects the frequency of the system, which is again used at the refrigerator level in order to perform UFLS. The model represents the system interactions in a dynamic way, as both effects of frequency and power are inherently coupled.

6 Inclusion of a Frequency Based Load Shedding for the Refrigerators

In order to observe the effects of a demand side management mechanism on the refrigerators, a simple and distributed load control was added to each agent in order to perform an UFLS. The operation principle is simple, the refrigerator is unplugged from the grid when a certain frequency threshold f_{off} is reached, and reconnected when a second threshold f_{on} is passed over. This aims for disconnecting the load when there is a frequency drop, usually caused by an production failure or a sudden, unforeseen load increase. The refrigerators are disconnected completely from the power source, so the compressor is not able to run any more. As they get reconnected, dependant on the thermostat (the evaporator temperature) will the compressor starts working again or not (see Figure 2).

7 Simulation Results and Discussion

In the first simulation, we show the normal operation of the refrigerator population. 250 refrigerators are simulated with an installed power of 120 W each. All the parameters of the refrigerators were varied by 5% in order to recreate a realistic population, in which each refrigerator has a slightly different configuration, as well as different initial conditions (internal temperature, etc.). This creates a heterogeneous population in which each refrigerator has its own state and therefore the individual load curves are particular for each agent. The average total load of the population is $\overline{P_f} = 11.5$ kW during normal operation, which corresponds to a coincidence of around $c = 40\%$. In order to achieve an impact on a realistic energy system, the population load is scaled up by $s_f = 1300$, which leads to an average scaled up load of 15 MW (for the refrigerators). The rest of the demanded load of the system was set to $P_o = 285$ MW, and the nominal generation power of the system to $P_g = 300$ MW.

7.1 A UFLS Scenario

Production failure is simulated through a sudden drop of 15 MW, which get restored linearly in the following 10 minutes (see Figure 5. This recreates a failure in a plant with a subsequent reaction of the system (other generators take the dropped load over following the secondary reserve mechanisms). In the first simulation, no UFLS on the refrigerator side was performed. We can observe the aggregated load curve which is rather smooth. The production failure causes the frequency to fall under 49 Hz for a short time, while the restoration of the power stabilises it within the following 10 minutes.

In the UFLS case the refrigerators disconnect themselves from the system when the frequency falls under $f_{off} = 49$ Hz, and we reconnect it as soon it drops above $f_{on} = 50$ Hz. As it can be seen in Figure 5, as soon as the frequency falls below the threshold, the complete refrigerator load is released. However, after some time, the load of the refrigerators increases up to twice its average value (around 30 MW, which means that almost 80% of the refrigerators are working at the same time). This rebound effect is due to the hysteresis process which controls the thermostat of the refrigerators. This happens even if the disconnection is only for a few seconds. The system frequency increases in consequence of the disconnection, and within 6 s after the disconnection reaches up to 50.4 Hz. However, due to the increase of the refrigerator demand (in comparison to normal operation), the frequency recovery is slower than without UFLS.

This very simple UFLS strategy only based on frequency thresholds therefore can help to restore the frequency quickly after an event, but can create a rebound effect due to the coincidence of the loads after the disconnection. This load increase is counter-productive for the frequency stabilisation.

7.2 Emergent Synchronisation and Oscillation of the System

A further scenario shows the effect of the same load shedding strategy, when increasing the impact of the refrigerator power on the total demand. This was achieved by increasing s_f from 1300 to 1760. As the refrigerators are shed and reconnected a short time after (when frequency drops above 50 Hz), they tend to coincide in a higher degree, as describe before (Figure 6), which is a *rebound effect*. The coincidence has shown to be greater after a load shedding. Yet, no oscillation appears.

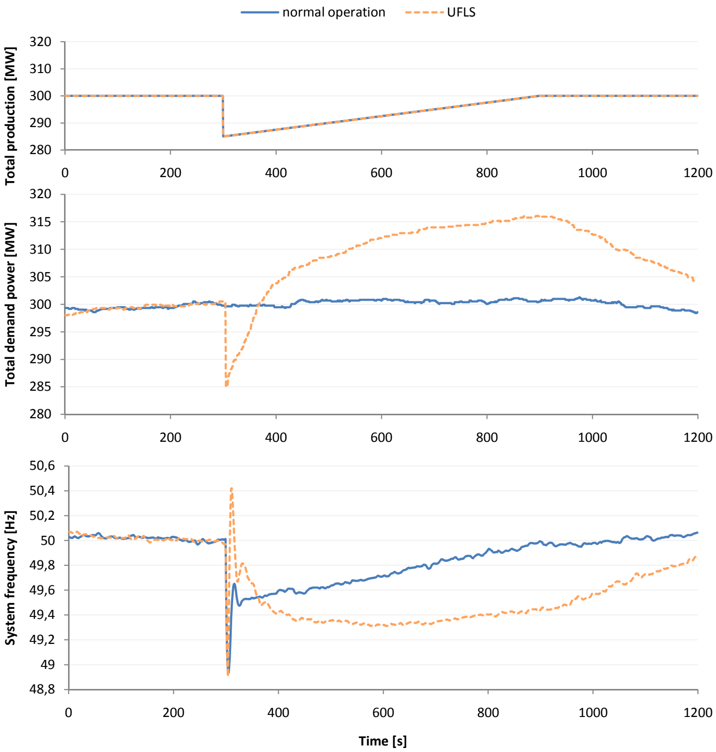


Fig. 5 Simulation results for a production failure scenario with and without UFLS

But as they reconnect again and again, a second under-frequency event takes place, which is not due to a production failure, but to the demand increase of the refrigerators. After some time, the system begins to oscillate with a period of 2-4 s in the simulated cases.

The appearance of synchronisation apparently does not depend on the number of individual refrigerators, but rather on the aggregated load that is shed. We could reproduce the effect with relatively small populations of refrigerators (50-300) by scaling up the load. This means that if the total load of the refrigerators which are managed by an UFLS is high enough in relation to the energy systems frequency response, a risk of synchronisation might exist.

Through the simulation model we found that the main reason for the emergence of oscillation is the rebound effect which is created after the first disconnection. This rebound effect emerges due to the hysteresis process on which the thermostat of the refrigerator works. The pulses tend to coincide more after each disconnection, as the thermostats cycles tend to synchronise. Similar effects have been described by [13] concerning fireflies and their light pulses, or synchronised hands clapping [10] in an audience. The refrigerator can be seen as a pulse-based oscillator. A typical

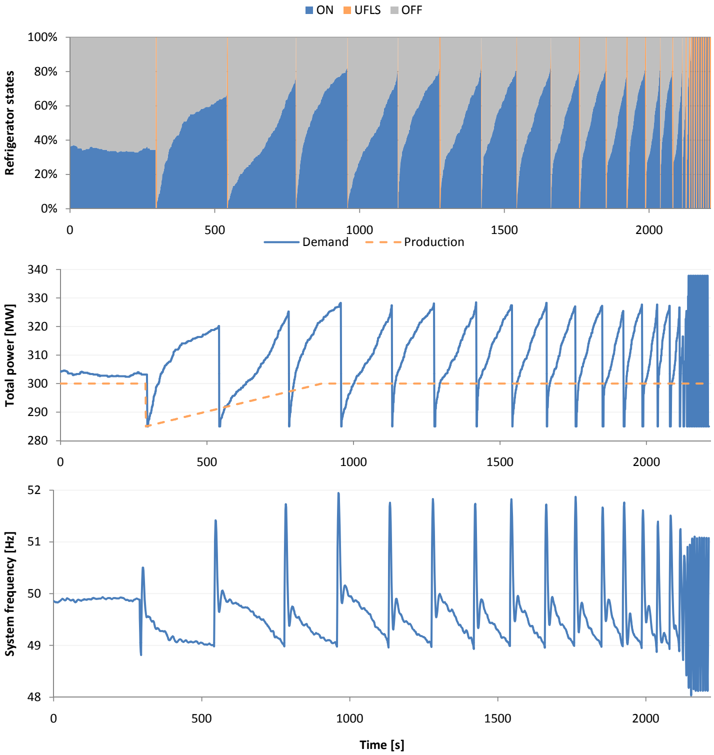


Fig. 6 Simulation results for a production failure scenario with UFLS. The system begins to slowly synchronise itself up to a complete oscillation of frequency and loads.

refrigerator cycle is described by an interval without consumption (in which the refrigerator warms up) and a second interval in which a constant power is consumed to cool down the internal cell through the heat pump mechanism. The length of this pulse depends on many factors, such as the technical parameters of the device, and its usage.

The phenomena of an oscillation is not predictable from the individual models of the refrigerators. It emerges in a rather unstable way, which is related to the state of the population (describe by the individual states of each refrigerator), which has a very large degree of freedom. This is typical for complex systems. Due to the used of an agent-based model we were able to recreate a population which is already capable to represent the system as a whole, in a disaggregated way, with its interactions among different levels.

The simulation environment allows to test UFLS strategies virtually in a large scale environment an analyse the impact at a systemic level. Individual models or non-interacting models where there is no feedback between the electrical grid and the load management are not able to capture these effects. It has to be noted that this effect only appears with a large number of refrigerators, and thus can only be difficulty analysed in real experimental cases, as the impact on the energy system must be large enough.

8 Conclusions

Through a multi-agent simulation model we represented a population of refrigerators which was coupled to a simplified electricity system model, recreating the frequency response of the grid. A calibrated and individually validated refrigerator model was taken as a base. A frequency based demand side management unit was added to the refrigerator, which unplugs the refrigerator at under-frequency events marked by a threshold, and reconnects the appliance when the frequency surpasses a second threshold. The model allows for cross-scales interactions by relating individual loads to a systemic model representing the frequency response.

A rebound effect was found using a very simple UFLS strategy. It leads to a synchronisation of the loads, which could be captured and analysed. An oscillation of the system if the refrigerator disconnected refrigerator load is high enough could be observed and should be avoided in real devices. A phase transition between a stable regime and an oscillating regime could be observed, in which the system tends to be unstable. In this intermediate state, it is not clear if the system will oscillate or return to stability. The non-determinism of the implemented model allows to explore these different states of the model.

9 Outlook

In other domains, emergence has been studied already (biology, social sciences), however, few applications of these theories has been made to engineering or energy systems. The properties of these effects should be further analysed. The simulation suggests that the probability of achieving an emergent oscillation depends on the amount of load (number of refrigerators) that are managed, the properties of the system (frequency response). Further, the model permits to test other UFLS strategies which could avoid these effects. The period of the oscillation for example, suggests to be a function of the characteristics of the population of the refrigerators, as well as the frequency response. As all refrigerators have different properties, it can't be deduced that there is a common period for all and that this is the reason for the oscillation. Moreover, the phase transition from a stable regime (no oscillation) towards a completely synchronised regime should be analysed more in detail, as the phase shift has shown to occur sometimes very quickly (some minutes), and others taking much longer (up to an hour).

A relation with the Kuramoto and Nikitin models [6, 13, 11], which are models for the behaviour of a large set of nearly identical coupled oscillators, would be worth studying.

Acknowledgements. We would like to thank the support of the colleagues of EDF who contributed to the measurements on the refrigerator.

References

1. Anderson, P.M., Mirheydar, M.: A low-order system frequency response model. IEEE Transactions on Power Systems 5(3), 720–729 (1990)
2. Borshchev, A., Kolesov, Y., Senichenkov, Y.: Java engine for uml based hybrid state machines, pp. 1888–1894. IEEE (2002)
3. Cerri, G., Palmieri, A., Monticelli, E., Pezzoli, D.: Identification of domestic refrigerator models including cool storage. In: International Congress of Refrigeration 2003, Washington D.C., USA (2003)
4. Kottick, D., Blau, M., Edelstein, D.: Battery energy storage for frequency regulation in an island power system. IEEE Transactions on Energy Conversion 8(3), 455–459 (1993)
5. Kremers, E., González de Durana, J.M., Barambones, O.: Emergent synchronisation properties of a refrigerator demand side management system. Applied Energy (in press, 2012) (accepted on July 18, 2012)
6. Kuramoto, Y.: Chemical oscillations, waves, and turbulence. Dover Pubns. (2003)
7. Laguerre, O., Derens, E., Palagos, B.: Study of domestic refrigerator temperature and analysis of factors affecting temperature: a french survey. International Journal of Refrigeration 25(5), 653–659 (2002)
8. Lee Hau Aik, D.: A general-order system frequency response model incorporating load shedding: analytic modeling and applications. IEEE Transactions on Power Systems 21(2), 709–717 (2006)
9. Moura, P.S., de Almeida, A.T.: The role of demand-side management in the grid integration of wind power. Applied Energy 87(8), 2581–2588 (2010)
10. Neda, Z., Ravasz, E., Brechet, Y., Vicsek, T., Barabasi, A.L.: Self-organizing processes: The sound of many hands clapping. Nature 403(6772), 849–850 (2000) 10.1038/35002660
11. Nikitin, A., Néda, Z., Vicsek, T.: Collective Dynamics of Two-Mode Stochastic Oscillators. Physical Review Letters 87(2) (2001), http://link.aps.org/abstract/PRL/v87/e024101
12. Short, J.A., Infield, D.G., Freris, L.L.: Stabilization of grid frequency through dynamic demand control. IEEE Transactions on Power Systems 22(3), 1284–1293 (2007)
13. Strogatz, S.H.: Sync: The Emerging Science of Spontaneous Order, 1st edn. Hyperion (2003)
14. Viejo Garcia, P., Gonzalez de Durana, J., Barambones, O., Kremers, E.: Quantitative system dynamics: Comparison of modeling techniques for the simulation of electromechanical systems. In: International System Dynamics Conference, Washington (2011)
15. Widen, J., Wackelgard, E.: A high-resolution stochastic model of domestic activity patterns and electricity demand. Applied Energy 87(6), 1880–1892 (2009)
16. Wooldridge, M.J.: An introduction to multiagent systems. Wiley, Chichester (2009)
17. Wright, A., Firth, S.: The nature of domestic electricity-loads and effects of time averaging on statistics and on-site generation calculations. Applied Energy 84(4), 389–403 (2007)

Chapter 19
Orchestrating Situation Awareness and Authority in Complex Socio-technical Systems

Guy A. Boy

Abstract. Systems engineering is developing everywhere in industry, but human issues incrementally emerge. In particular, this systemic technology-centered approach to engineering rigidifies organizations and lead to surprises. People involved are not fully aware of what is going on. This paper identifies situation awareness and authority issues in complex systems design and management, and discusses possible solutions. It more specifically focuses on an Orchestra model of socio-technical systems.

1 The Problem

Systems engineering was born in the 1940s at Bell Telephone Laboratories (Schlager, 1956). Systems engineering focuses on large complex systems. The main goal was to figure out properties of the whole system often composed of other systems interacting among each other. We typically talk about systems of systems (SoS), e.g., the National Airspace System (NAS). Large organizations, such as NASA and DoD, contributed to develop the systems engineering discipline, which was intended to provide solutions for handling changes and growing complexity of large projects and programs. Several methods emerged such as Quality Function Deployment (QFD), also called the House of Quality, which was developed to decrease design and development time, as well as team efforts (Hauser & Clausing, 1988).

Systems engineering practice tends to generate lots of data that need to be handled correctly and timely. Awareness is a crucial issue. People working together need to be able to have some level of shared knowledge about each other's activities. It is not because reports are generated that they are necessarily read and understood. Quality techniques generate counterproductive behaviors; for example,

Guy A. Boy
Florida Institute of Technology
Human-Centered Design Institute (HCDi)
150 West University Blvd., FIT BOX 6044
32901, Melbourne, USA
e-mail: gboy@fit.edu

people tend to spend more time reporting, i.e., writing reports, than really working together. In too many places, reporting has become more important than work itself. Another factor is the dichotomization of work. Work has been divided into pieces that support financial management, but not necessarily technological management, and even less human resource management. Most problems come during integration.

Heavy-reporting techniques block **creativity**, and introduce rigidity and tremendously decrease motivation of the personnel involved. It is clear that large projects and programs must be highly structured, but our highly software-immersive organizations and products require different design thinking. Our world evolves very fast from three main interrelated points of view, **technology, organization and people** (TOP). It is extremely difficult to handle this TOP evolution with rigid techniques. Therefore, we need to find different approaches and models that provide more flexibility for detecting abnormal situations, handling changes and supporting human involvement.

Situation awareness (SA) and authority are crucial human processes that require more attention. They can be at the level of an individual or collective. SA has been extensively studied in human factors and ergonomics at the individual level and in complex collaborative environments. Today, design and operations evolve from individual performance to network-centered approaches. Who needs to be aware of what? Who is in charge? The question of authority sharing is becoming crucial. The main issue is the lack of distributed SA of systems within systems of systems. Most of the time, SA issues emerge from SoS complexity, in the same way as attractors in chaos theory. The Orchestra model was developed to better understand design and management of socio-technical systems as an alternative to traditional army models.

2 Situation Awareness and Authority in Complex Socio-technical Systems

Many researchers and practitioners have developed fundamental work on situation awareness (Beringer & Hancock, 1989; Billings, 1995; Endsley, 1988, 1995, 1996). In aviation for example, the aircrew needs to be aware of external situations (e.g., weather conditions, traffic, terrain) as well as internal situations (e.g., systems states, failures, fuel level). Another example is the management of a large company that needs to keep acute awareness of market and competition, as well as articulation of internal services and suppliers performance.

Why is **situation awareness** (SA) so crucial? The never-ending integration of software into systems not only creates a bigger distance between people and physical machines, but also the very nature of human-machine interaction changed and continues to change. Software enables the emergence of new interaction styles and experience. The main issue is to provide the right interaction capability at the right time and in the right way. This is difficult. Why? The SA concept includes perception of the situation, its comprehension and the necessary means of projection (Endsley, 1995) in order to act safely, efficiently and comfortably. Having the right machine affordances and useful skills to be learned supports SA.

To be more specific, flying an airplane requires constant situation understanding in order to perceive and anticipate upcoming events. For fuel management, the pilot must be aware of the state of the fuel level in a selected tank; must understand the rate variation of fuel use, detect abnormalities and reorganize the way engines are fueled; and must determine if he or she is unlikely to reach destination due to the unexpected rate of fuel use and divert to an alternate airport prior to engine failure (Spirkovska, 2010). We see in this example that pilots need to have the right level of information at the right time to decide and act appropriately. Designers have to find solutions that take into account limited amounts of physical space for presenting the situation and potentially provide appropriate guidance. Usability engineering (Nielsen 1993) and more generally human factors and ergonomics (Meister, 1999) provides methods to this end, but without creativity and expert continuous testing such complex systems cannot be properly designed.

SA is intimately related to authority: "I need to be aware of what is going on because I am in charge." **Authority** is defined from two main perspectives:

- **control** in the engineering sense (i.e., who is in charge and competent for a given task and function), and
- **accountability** in the legal sense (i.e., we are always accountable to someone else, and accountability includes responsibility).

In this paper an Orchestra model will be introduced to support both SA and authority sharing among agents. Both SA and authority can be shared and distributed. For that matter, SA and authority can be shared and distributed (Artman & Garbis, 1998). Distributed SA is therefore "an emergent property of collaborative systems, something that resides in the interaction between elements of the system and not in the heads of individual operators working in that system." (Salmon et al., 2009).

3 The Orchestra Model

The Orchestra Model was designed over the years (Boy, 1991) and finally refined during a study carried out from 2006 to 2008 on **authority sharing** in the airspace system, the PAUSA[1] project (Boy et al., 2008; Boy & Grote, 2009). One of the major results from the PAUSA project was the definition of four main processes that support authority allocation in the overall organization (Boy, 2009). These processes are explained as follows.

- First, the process of **sharing** refers to information, knowledge and resources involved among a set of actors who need to have a shared awareness of the situation. Human and machine agents should interact using the same frame of reference. Cooperative authority sharing requires such a mutual understanding of

[1] French acronym for Authority sharing in the airspace (Partage d'AUtorité dans le Système Aéronautique). Nine major organizations, twenty-six researchers and practitioners were involved. We deliberately focused on the identification of human and organizational factors involved in Air Traffic Management automation today, in 2015 and after 2020.

the situation. The human factors community has widely addressed this issue of shared SA (Endsley & Jones, 1997), but the point here is to provide a framework of reference, like a music theory for all agents (i.e., a shared ontology).

- Second, the process of **distribution** refers to cognitive function[2] allocation. Each agent in the overall system has a role. We say that authority is distributed among actors with respect to their roles. Roles are defined according to competency, goals, contexts and resources. The pie-chart metaphor can be used to represent the distribution/allocation of individual chunks (functions) to agents. Organizational automation tends to format jobs, in the sense that each actor has a very specific cognitive function to execute (Boy & Grote, 2011). This distributed cognition phenomenon was first introduced by Edwin Hutchins (1995),
- Third, the process of **delegation** refers to contract setup (i.e., an actor sets up a contract with another actor). In this case, authority is delegated from an agent to another (i.e., the latter will execute the terms of the contract for the former, and the former will have to manage contract execution).
- Fourth, the process of **trading** mainly refers to negotiation among actors. Trading is commonly performed in real-time, as opposed to being planned (even if some patterns could be anticipated in order to facilitate trading operations), and requires the best mutual understanding among competent actors. Authority is traded among actors according to possibly conflicting goals (or roles), contexts and resources.

Problems arise from task definition, coordination and ability of all actors to articulate their work with others. Even if each actor has a well-defined cognitive function, its context of use is often too rigidly defined for appropriate articulation and intersubjectivity[3] among actors. This is why a model that enables the study of socio-technical evolution and emergence of new practices is needed.

3.1 Orchestra Components

No simulation can be purposefully and efficiently carried out without a **conceptual model**. In this scenario-based design approach to function allocation, the Orchestra model is an alternative to the traditional army-type model that supports a hierarchical decomposition of functions. Five categories of entities must be defined.

(1) **Music theory** that supports various information-flows and provides a common frame of reference for all agents in the environment.
(2) **Scores** that agents are required to use in order to support their assigned functions during operations. **Composers** typically develop scores and articulate

[2] A cognitive function is typically defined by its role, context of validity and resources that support its execution (Boy, 1998).

[3] Intersubjectivity typically refers to a psychological relation between people, whether it is common sense, an agreement or a divergence.

them among each other to coordinate overall performance of the musicians. These composers still remain to be identified correctly in the ATM case.
(3) **Conductors** who provide the operational timing patterns, and consequently will be responsible for effective information flows (i.e., the overall symphony performance) among musicians and the audience (i.e., end-users).
(4) **Musicians** themselves who are required not only to perform what their scores say, but also to articulate their own performance with the others.
(5) The **audience** that includes customers of the symphony.

3.2 Three Models of Interaction among Agents

There is a vast amount of research in the domain of shared (or not shared) mental models (Kanki & Foushee, 1989; Walz et al., 1993; Malone & Crowston, 1994; Faraj & Sproull, 2000; Mathieu et al., 2000). However, there is still a large amount of research to be conducted on the difficult problem of representing and simulating interactions among agents. Awareness of the organizational environment is crucial here. Two human beings interacting between each other incrementally **adapt** their own awareness of the other as conversation goes on (i.e., each of them learns from the other and consequently his or her model he or she has of the other evolves during conversation). Interaction with or through machines is a different matter. In a socio-technical organization such as an orchestra, agents are interrelated with respect to three kinds of interaction models (Boy, 2002). These models are distinguished by knowledge each agent has of others in the organization.

(1) When agents do not know each other, the best way to interact safely, efficiently and comfortably is to be supervised. **Supervision** is the first interaction model. None of the supervised agents has the authority to decide what to do; a supervisor does it for them.
(2) **Mediation** is the second interaction model. Agents have a common frame of reference (CFR) through which they are able to interact. They still do not know each other deeply, but they know that they can interact between each other through the CFR. In addition to CFR, there are mediating agents who facilitate interactions. In WYSIWYG user interfaces for example, in addition to desktop metaphors, there are mouse-sensitive help lines that pop-up on demand. In this model, authority is distributed among the agents.
(3) The third interaction model is **cooperation by mutual understanding**. This is what people usually do when they interact with each other. This model assumes that agents are able to construct a mental model of the others in order to perform better in future interactions. People interacting among each other do this naturally. Very simple instances of such a model have been developed and used so far on computers. For example, some pieces of software are able to learn a user's habits and incrementally provide smart options or suggestions. This is the case of current text processors that are able to learn a user's specific lexicon from frequent uses of words. Web browsers remember frequently used links, etc. In this model, authority is traded between the agents. In human-human interaction via machine agents, related technology

should provide appropriate SA means to enable sustainable and symbiotic communication.

To summarize, there is a continuum from the supervision model of interaction where authority follows a top-down army-type model, to the mediation model of interaction where authority follows a transversal orchestra-type model, to cooperation by a mutual understanding model of interaction where authority follows a more-chaotic trade model. These interaction models are very useful to support the way cognitive functions are implemented in complex software not only from a human-computer interaction point of view, but also from an internal subsystem-to-subsystem point of view. In particular, they also provide an articulated way to validate large object-oriented software. These three interaction models implicitly assume that agents have the same global goal or local goals compatible the global goal (e.g., performing the symphony). It may happen that some agents have competing goals (e.g., competing for a limited set of resources). Competition is an orthogonal dimension to the three interaction models where supervisor would become referee, mediator would become counselor, and competing agents could mutually understand each other to improve gains. Therefore, the three interaction models can be non-competitive or competitive.

A multi-agent system becomes mature when each agent understands its role in the right context using available resources (i.e., cognitive function definition). **Maturity** is guided by the type of interaction agents that are able to perform in a given context (i.e., supervision, mediation or cooperation by mutual understanding). In particular, it is clear that whenever cooperation by mutual understanding is possible, interactions are very advanced among agents, but we also need to know when human and/or machine agents need mediators, and even more specifically supervisors. Maturity is then tested in terms of level of autonomy (i.e., an agent is said to be autonomous when he/she/it is able to handle its allocated function independently). In some cases, an agent is not able to be autonomous and requires assistance in the form of mediation or supervision. It is important to realize soon enough during the design process when this kind of interaction model might be predominant. Surprises arise when this kind of test is not performed in advance. Note that supervision and mediation are perfectly acceptable models when they are well understood and accepted by the various team players. Like musicians who are collaborating within an orchestra, agents need to accept either the lead from other musicians or to be mediated through scores, which themselves were coordinated by the composer. The maturity process may take some time before the multi-agent system reaches an acceptable maturity level. It is based on competence of each agent, their abilities to work with others in the "orchestra" and a clear understanding of the various interaction models that are required to run the overall "symphony".

4 Authority Sharing Illustrated in the Airspace Domain

Authority sharing is one of the major themes of the next generation of air traffic management system (ATM), and flight deck automation in particular. The fact that we will have more aircraft in the sky (i.e., air traffic capacity increase), and we want to enhance safety, requires deepest research on the way various functions are being reallocated among various agents. We need to better identify pilots' information requirements and communication needs to perform tasks currently managed by air traffic control (ATC), which will greatly increase the needs for a pilot's awareness of the surrounding airspace, (human and system) failure identification and recovery, and unexpected-event handling in this dynamic and complex multi-agent infrastructure.

Therefore, we need to co-design and co-adapt both technological and organizational support. Avionics software is now highly sophisticated, enabling many machines to be considered as agents (i.e., having cognitive functions as humans have). Human and machine agents are more interconnected in the airspace than ever before, and their inter-relationships are often crucial to understand, master and support. This evolving ATM multi-agent world is critically situated and context identification is a primary concern. In particular, flight deck automation will have to be designed taking into account that pilots will gain autonomy, thus changing coordination requirements.

Consequently, function allocation needs to be addressed during the whole life cycle of all ATM systems. Cognitive function analysis is typically used to support the analysis, design and evaluation of such function allocation. More specifically, cognitive processes, such as authority sharing, distribution, delegation and trading, must be addressed. While there are human cognitive functions that can be predicted during design, there are some that will only emerge from use. This is why **scenarios** should be extensively developed and **human-in-the-loop simulations** (HITLS) carried out.

Spacing and merging (S&M) in dense traffic is a difficult problem; in particular, the sequencing of arrival flows through a new allocation of spacing tasks between air and ground. Today, air traffic controllers (ATCOs) solely manage aircraft S&M in busy airspaces. They control both the sequencing decisions and manage the merging routes, airspeeds and altitudes, guiding each aircraft. Controllers are aided by today's tools, which range from simple Letters of Agreement (LOA) and standard navigation aids, to more advanced systems like today's GPS approaches and integrated Flight Management Systems (FMS). The new Required Navigation Performance (RNP) procedures are the latest improvement down the traditional path of providing pilots with standard procedures and more accurate ways to follow them. While this approach is an important one, it alone will not solve the future problems of airspace congestion because it addresses only execution and does not address the major issue, which is **coordination**. Today, ATC is a centralized army-type decision point (i.e., all decisions must pass through this point and be distributed in a serial manner to all pilots within the managed airspace). This is a clear bottleneck that is highly dependent on skills of the controller to analyze the situation, make decisions, and then communicate required information to each aircraft as necessary.

Future ATM systems will enable pilots to be more autonomous and consequently will require more coordination among agents. They will have contracts like musicians have scores. Consequently, these contracts will have to be coordinated by some sort of planners, like the composers do. From this point of view, the main difference between ATM and a symphony is that contracts may change during performance, like the play of a Jazz orchestra. Authority trading will be a major issue. Situation awareness of each agent remains a central emergent cognitive function to investigate and identify during design and development. In fact, agent's authority and SA are intimately coupled, and their identification determines the type of interaction model an agent will have with other agents that are relevant in the operational context. Sometimes supervision is the only interaction model that is possible, and agents will need to refer to a conductor. In other situations, they will be able to interact via contracts (scores) and trust this mediating means. Finally, it will happen that agents will perfectly understand what the others are doing, and therefore will communicate directly.

Off-nominal situations are infrequent, but have a tremendous impact when they do occur. They typically induce **dynamic function allocation**:

- appropriate agents will have to be aware of the situation change (resulting in a different common frame of reference, or music theory and style if we take the Orchestra metaphor),
- contracts will have to be redefined and coordinated (composer role), and
- consequent operations will have to coordinated (conductor role).

For example, it may happen that an aircrew is not able to make it off the runway at the high-speed exit and take a full-length landing. In a congested terminal area, the following aircraft will have to perform a go-around maneuver. First, the aircrew must realize they are not going to make the exit (SA cognitive function), they must manage the landing (safety-assurance and action-taking cognitive functions), and find time to let the controller know (coordination cognitive function). Consequently, the ATCO must inform the trailing aircraft and potentially all other aircraft sequenced on the approach (coordination cognitive function). All these cognitive functions must be implemented at the right time, which might not be the case given the extra workload during this kind of operations. Information flows are highly dynamic and can only be managed by well aware and knowledgeable agents, and possibly new technology. For example, the ATCO re-sequencing traffic may also discover there is an aircraft that is low on fuel and requires an emergency landing. Creative decision-making is consequently the appropriate cognitive function that is at stake for the ATCO. On this very simple example, we see that authority must be shared in a timely manner among appropriate agents.

One way of managing this coordination problem is to develop appropriate automation. Automation can be used to detect when an aircraft will not make the exit and automatically signal the controller, elevating this burden from the pilot who is likely under high workload already. That same signal could automatically be sent to all the trailing aircraft. This kind of additional agent is expected to create more SA among involved agents and therefore increase their mutual understanding of the situation (thus promoting the third interaction model). In addition, the ATCO, as a conductor, could make a single call confirming the situation and requesting reduced speeds. Each aircraft could acknowledge through their flight displays instead of using radio communications and ATCOs would see each response on their own screens. If this kind of solution seems to simplify the job of the various agents, it is mandatory to make sure that they are properly trained or fine-tuned, using the right cognitive functions.

5 Discussion

Following-up on the illustrative example described above, systems such as air-air surveillance capabilities (ADS-B) and cockpit automation (ASAS: Airborne Separation Assistance System) are being designed to enhance authority sharing between the flight deck and the ground. The evolution between what is currently done and the next generation of air-ground environments requires careful scrutiny of function allocation and keeping automation as simple as possible, in terms of flexibility for the actors. Aircraft spacing and merging (S&M) technology remains immature, requiring further investigation and development. In terminal areas, S&M currently relies on ATCOs' skills and experience and is affected by weather conditions, rates of runway use, ground congestion and other factors. In the perspective of authority delegation to the flight deck, new approaches to S&M need to be invented, especially in high-density traffic situations.

It is now important to identify required functional evolutions and properties that emerge from this evolution, taking into account a representative environment with very high traffic. Referring to the Orchestra Model, new approach procedures and terminal area patterns are part of the common frame of reference (i.e., a music theory analog). Generic contracts, as scores, needs to be defined according to cognitive functions that will emerge from both new automation and organizational rules, mainly coordination rules. Contract coordination should be both anticipated (composer role) and managed (conductor role). Finally, function allocation should be thought in terms of authority sharing in the sense that several agents share responsibility and control in context. It could be a priori defined (i.e., each function represented by a contract is allocated to an appropriate agent). It should also be dynamically defined (i.e., cognitive function may be allocated with respect to an ongoing situation). As already seen, dynamic function allocation requires appropriate SA (i.e., there is a constant need to look for potential hazards and understand perception and cognitive limits of various agents in order to compensate with additional cognitive functions and maintain an appropriate cognitive stability). Such cognitive functions could be additional resources in the form of supervisors, mediators or automated links that provide a better mutual understanding. Of course, their implementation and operational costs should be evaluated with respect to relevant human and technological factors. The choice of their effective implementation in the real world depends on these evaluations.

Other approaches, such as cognitive systems engineering/joint cognitive systems (Hollnagel & Woods, 2005), consider the growing complexity of socio-technical systems, problems and failures of clumsy technology, and the limitations of linear models and the information-processing paradigm. They also recognize the need for cognitive function (Boy, 1998) "in the mind" (i.e., processes that mediate responses to events). In fact, this anthropological approach of cognition was already started with the identification of situated actions (Suchman, 1987) and distributed cognition (Hutchins, 1995). All these contributions emphasize context as the main research issue. In fact, people are both goal-driven and event-driven; they are opportunistic according to context. This is why context is so important to identify and take into account.

Context is an extremely difficult subject to grasp and identify since it is directly associated to the persistence of situations and events (Boy, 1998); some are long enough to be captured, and some others are too short to even be perceived. This is why a scenario-based approach carried out by domain-expert professionals is necessary. The Orchestra Model is a metaphoric model that enables handling context in a functional and structured way, since the cognitive function representation includes a context attribute by construction. Identification and categorization of the possible connections and interactions among agents through their cognitive functions enables us to better understand various relevant issues of SA. In fact the way we identify and categorize the world is crucial in the perception of context when acting. It is clear that all metaphors are very limited, and the Orchestra metaphor has limitations when we use it to describe socio-technical systems. However, it incrementally emerged as an acceptable model of the evolution of our software-immersive multi-agent environment, and the ATM environment in particular.

6 Conclusion

Since context is a major concern in the design of appropriate socio-technical systems, scenarios are very good tools to support the elicitation of emergent cognitive functions. Scenario-based design requires support by a strong conceptual model. The Orchestra Model is a good conceptual tool to categorize cognitive functions in air traffic management problems, their allocation among human and machine agents, as well as various relevant relationships between them. We presented an operations application, but the Orchestra model also applies to design and manufacturing (Boy, 2012).

The Orchestra model defines relationships between agents, supported by contracts that are very similar to scores in music. In addition, when there are several agents to coordinate, these contracts (scores) need to be coordinated; this is typically the role of a composer in music. Despite initial planning (i.e., initial coordination of contracts), there are always events that are not anticipated either because they are intentions from agents that differ from the original plans, or unexpected external events. These events require dynamic re-allocation of functions, and therefore modification of initial contracts. This is typically the role of a conductor. Agents, as musicians, need not only to be competent in performing their functions; they also need to understand what the other agents are doing. This is why we also need interaction models, based on a common framework (music theory). In the best case, agents communicate between each other by mutual understanding, but they may require being supervised or mediated when they do not have acceptable situation awareness.

References

Artman, H., Garbis, C.: Situation Awareness as Distributed Cognition. In: Proceedings of ECCE 1998, Limerick, Ireland (1998)

Beringer, D.B., Hancock, P.A.: Exploring situational awareness: A review and the effects of stress on rectilinear normalization. In: Proceedings of the Fifth International Symposium on Aviation Psychology, vol. 2, pp. 646–651. Ohio State University, Columbus (1989)

Billings, C.E.: Situation awareness measurement and analysis: A commentary. In: Proceedings of the International Conference on Experimental Analysis and Measurement of Situation Awareness. Embry-Riddle Aeronautical University Press, Florida (1995)

Boy, G.A.: Advanced interaction media as a component of everyday life for the coming generation. Invited Speech at he. JMA Marketing World Congress, Tokyo, Japan (1991)

Boy, G.A.: Cognitive Function Analysis. Ablex. distributed by Greenwood Publishing Group, Westport (1998) ISBN: 1-56750-377-2,

Boy, G.A., Salis, F., Figarol, S., Debernard, S., LeBlaye, P., Straussberger, S.: PAUSA: Final Technical Report. DGAC-DPAC, Paris, France (2008)

Boy, G.A., Grote, G.: Authority in Increasingly Complex Human and Machine Collaborative Systems: Application to the Future Air Traffic Management Construction. In: The Proceedings of the 2009 International Ergonomics Association World Congress, Beijing, China (2009)

Boy, G.A., Grote, G.: The Authority Issue in Organizational Automation. In: The Handbook of Human-Machine Interaction. Ashgate, UK (2011)

Boy, G.A.: Orchestrating Human-Centered Design. Springer, U.K. (2012)

Endsley, M.R.: Situation awareness global assessment technique (SAGAT). In: Paper Presented at the National Aerospace and Electronic Conference (NAECON), Dayton, OH (1988)

Endsley, M.R.: Toward a theory of situation awareness in dynamic systems. Human Factors 37(1), 32–64 (1995)

Endsley, M.R.: Automation and situation awareness. In: Parasuraman, R., Mouloua, M. (eds.) Automation and Human Performance. Theory and Applications, pp. 163–181. Laurence Erlbaum, Mahwah (1996)

Endsley, M.R., Jones, W.M.: Situation awareness, information dominance, and information warfare (No. AL/CF-TR-1997-0156). United States Air Force Armstrong Laboratory, Wright-Patterson AFB, OH (1997)

Faraj, S., Sproull, L.: Coordinating Expertise in Software Development Teams. Management Science 46(12), 1554–1568 (2000)

Hauser, J.R., Clausing, D.: House of Quality, Harvard Business Review Article (May 1, 1988)

Hollnagel, E., Woods, D.D.: Joint cognitive systems: Foundations of cognitive systems engineering. CRC Press-Taylor & Francis, Boca Raton (2005)

Hutchins, E.: How a cockpit remembers its speeds. Cognitive Science 19, 265–288 (1995)

Kanki, B.G., Foushee, H.C.: Communication as Group Process Mediator of Aircrew Performance. Aviation, Space, and Environmental Medicine 20(2), 402–410 (1989)

Malone, T., Crowston, K.: The Interdisciplinary Study of Coordination. ACM Computing Surveys 26(1), 87–119 (1994)

Mathieu, J., Goodwin, G.F., Heffner, T.S., Salas, E., Cannon-Bowers, J.A.: The Influence of Shared Mental Models on Team Process and Performance. Journal of Applied Psychology 85(2), 273–283 (2000)

Meister, D.: The History of Human Factors and Ergonomics. Lawrence Erlbaum Associates, Mahwah (1999) ISBN 0805827692

Nielsen, J.: Usability Engineering. Academic Press, Boston (1993)

Salmon, P.M., Stanton, N.A., Walker, G.H., Jenkins, D.P.: Distributed Situation Awareness: Theory, Measurement and Application to Teamwork. Ashgate (2009) ISBN: 978-0-7546-7058-2

Schlager, J.: Systems engineering: key to modern development. IRE Transactions EM-3 (3), 64–66 (1956)

Spirkovska, L.: Intelligent Automation Approach for Improving Pilot Situational Awareness. NASA Ames Research Center (2010)

Suchman, L.A.: Plans and Situated Actions. Cambridge University Press (1987)

Walz, D.B., Elam, J.J., Curtis, B.: Inside a Software Design Team: Knowledge Acquisition, Sharing, and Integration. Communications of the ACM 36(10), 63–77 (1993)

Chapter 20
A Collaborative Distributed Control and Building Performance Simulation Based on Systems Engineering Practice

Azzedine Yahiaoui and Abd-El-Kader Sahraoui

Abstract. Building itself is like a complex system with a number of physical processes that interact with each other and with the environment. From the control point of view, this system is composed of multi dynamic subsystems that must be monitored and controlled in order to achieve occupants' well-being at lowest energy use possible. However, environmental and occupants change in a building, and this increases the complexity in applying control systems. For this reason, this paper presents a framework for the application of systems engineering (SE) concept – a systematic approach for adapting procedures, tools, and standards to all practical problems – in the design of control systems by run-time coupling for building performance applications. Although this is based on SE good practice and corresponding SE standards, the development lifecycle of control systems are covered ranging from the operational needs to implementation, operation and disposal. As an essential step toward this goal, namely using computer simulations to evaluate the impact of advanced control systems on buildings indoor operation and energy efficiency, this paper describes a collaborative distributed simulation that was developed and implemented between different simulation tools such as ESP-r and Matlab/Simulink. The paper also addresses a study of verification and validation (V&V) issues within this framework: inconsistency checking, traceability issues and all requirements related in SE standards, especially EIA-632.

Azzedine Yahiaoui
Center for buildings and Systems
Technische Universiteit Eindhoven (TU/e)
5600MB Eindhoven, The Netherlands
e-mail: a.yahiaoui@bwk.tue.nl

Abd-El-Kader Sahraoui
LAAS-CNRS, 7 avenue du Colonel Roche, F-31077 Toulouse, France
Université de Toulouse; UPS, INSA, INP, ISAE;
LAAS, F-31077 Toulouse, France
e-mail: sahraoui@laas.fr

Keywords: systems engineering, complex systems, distributed simulation, control systems, building performance simulation, verification and validation.

1 Introduction

With today's environment changing concerns, advanced control systems must be applied to buildings Heating, Ventilation, Air-Conditioning and Refrigeration (HVAC&R) equipment and lighting components in order to achieve occupants' well-being and comfort aspects at lowest energy use and greenhouses gas emissions possible. Such developments need to be designed, implemented and deployed rapidly in order to reduce time involved in improving building HVAC&R equipment and lighting components for new deployments. In [21], it has been stated that building performance applications are substantially influenced by the quality of automation and control systems provided in it. Therefore, the application of advanced control systems in building performance operations has for objective to promote sustainable buildings as well as the development and the next generation issues of economic strategies. Hence, sustainable buildings or "high-performance buildings", also called "green buildings" are buildings with complex systems, which must have minimum impacts on the built and natural environment. One of the major challenges today is the protection of the ecosystem. Although the building is designed and constructed for a long period, its life cycle can vary from some months to hundreds of years. During the period of its usage, the building can progressively consume significant amounts of natural resources, produce large quantities of gas emissions and affect the ecosystem in many different ways. As a result, there is a need to apply advanced control systems in order to make buildings more sustainable and efficient while improving occupants' well-being and indoor operations as well as reducing their impacts on environmental quality.

The dwelling aspects concerned with sustainable buildings are comfort, well-being and energy efficient of occupants. The history shows that the protection of human health, in which the ecosystem protection is associated with, is much more closely related to comfort issues (thermal and visual comfort as well as indoor air quality). For this reason, this paper presents a framework for the application of SE concept in design of control systems for building performance applications by run-time coupling. While recognizing that there are many other aspects involved in achieving lower energy consumption, greater satisfaction and higher productivity of the occupants, the work described in this paper, focuses on modeling and simulation for better design of control systems for the indoor environment in buildings.

The benefits of integrating advanced control systems in building performance applications would consist of offering an enhanced functionality of building indoor operations and of building HAVC&R equipment and lighting components resulting in better buildings performance, as well as an improved reliability with further reduction in energy costs. To well-establish such a vast diversity of control functions for all the building HVAC&R equipment and lighting components that operates within the building environmental performance, it is necessary to take into account several factors, which may particularly include indoor environment variable (or processes), occupants' requirements, and economic parameters in the form of microeconomic and macroeconomic levels. It is then important to mention that such sophisticated improvements would occur as a result of automated building HVAC&R equipment and lighting components (or more precisely, as the art of automating buildings). Therefore, automated building refers to the automation and control of building HVAC&R equipment and lighting components, which have been the subject of Building Automation and Control Systems (BACS) or Building Automation Systems (BAS) since last century.

However, a variety of simulation tools for building performance and energy analysis have been developed, ranging from the simple and estimated to the difficult and detailed use. Among them, a small number of programs have given up because they became either useless or too restricted. For example in [2], it has been pointed out that the energy consumption of new buildings can be reduced by as much as 50% with little or no impact on the cost of ownership through the use of SE concept, as shown in Figure 1. Detailed specifications are translated into test procedures, design, and user documentation.

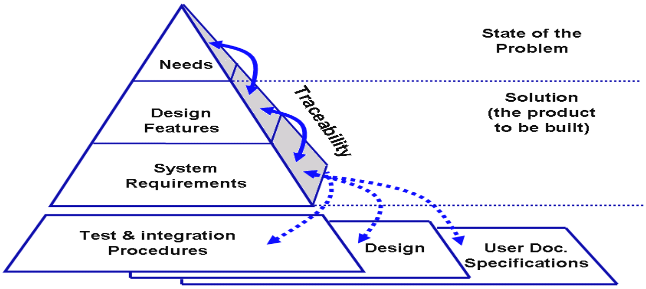


Fig. 1 Approach to integrating control systems in building performance applications

2 Context, Problem Statement and State of the Art

2.1 Context and Application

The context in this work is related to the development and implementation of advanced control systems in buildings, especially in building environmental performance. The integration of building science engineering, architecture, construction management and risk assessment for new construction projects and existing buildings becomes a must. Building applications require a lifecycle of development as shown in Figure 2.

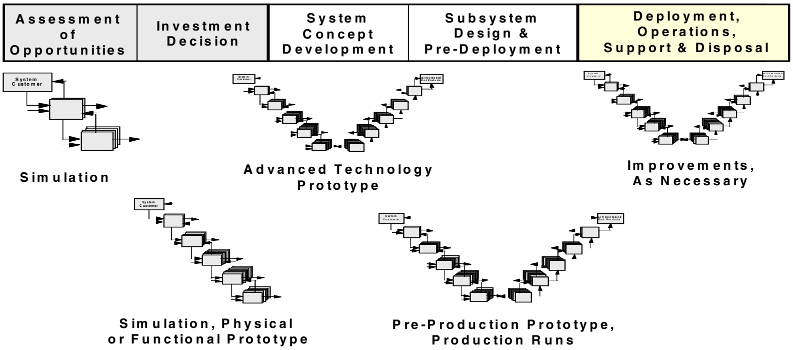

Assessment of Opportunities
Investment Decision
System Concept Development
Subsystem Design & Pre-Deployment
Deployment, Operations, Support & Disposal
System Customer
Simulation
Advanced Technology Prototype
Improvements, As Necessary
System Customer
Simulation, Physical or Functional Prototype
Pre-Production Prototype, Production Runs

Fig. 2 Enterprise-based life cycle phases

In this context, this study is concerned from the systems development (including subsystems design and deployment operation) to the deployment operations concerning the multiple development and implementation of the initial systems as a final system or a prototype.

2.2 Problem Statement

There are numerous approaches to the integration of new technologies in the buildings domain. However, a comprehensive framework where a building is seen as a system is not yet industrialized. Such developments using SE practices require to define requirements ranging from users/stakeholders needs to institutions, standards, local, national regulations, etc., where the limit of such integration cannot be defined if a global approach is not used.

2.3 State of the Art

Up to our knowledge, there are not yet establishment mythologies and approaches linking SE practices with the appropriate use of building HVAC&R (equipment lighting components. However, some research projects, such as in [2], use SE concepts to home buildings as advanced framing and insulation methods to increase efficiency and comfort while decreasing construction and energy costs.

3 Systems Engineering Concept and Design Methodology Process

3.1 Systems Engineering Practice

SE practice is not new, but the discipline is. It started with large-scale programs in USA, mainly in aeronautics [10], in space [11] and particularly in defense [4], [6]. Furthermore, it is getting popular in country having a well-established aeronautic and military industries; it has been since the 1990's deployed in manufacturing, automotive [7] and recently in Society of Manufacturing Engineers (SME) [8]. As a simple definition: Systems Engineering (SE) is an interdisciplinary approach encompassing the entire technical effort to evolve and verify an integrated and life cycle balanced set of system people, product, and process solutions that satisfy customer needs. SE encompasses (a) the technical efforts related to the development, manufacturing, verification, deployment, operations, support, disposal of, and user training for, system products and processes, (b) the management of the system configuration, (c) the translation of system definitions into work breakdown structures, and (d) information for management decision making

3.2 Define Systems of Systems

The deployment of SE product can be carried out in a comprehensive approach by separating the final product (i.e., building) from the enabling product (i.e., control systems) and development product (simulation tools, etc.). This can be best illustrated by the following Figure 3.

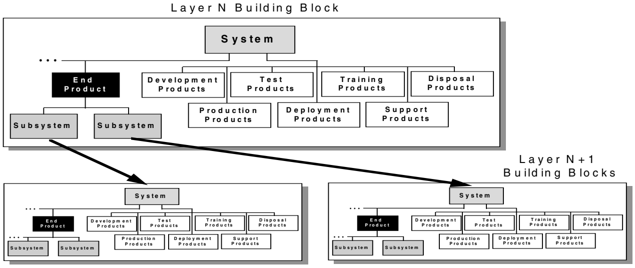


Fig. 3 Hierarchy of building blocks

A single block will really define the complete solution to a complex problem more typical of the design project. When an end product sub-system requires further development it will have its own subordinate building block. Once the descriptions of the end product of the initial building block are completed, and preliminary descriptions of the end product subsystems are defined, the development of the next lower layer of building block can be initiated.

3.3 SE Deployment and Standard

SE concept is a diagram that includes the known processes that are daily in use. However, there are many models of popular SE standards, such as: ISO-15288, ANSI/EIA-632, IEEE-1220, SP-6105, ECSS-E-10A, but certain phases of these diagrams are frequently similar to each other [12]. Although the objective of this research is to apply the deployment strategy to a building model, the EIA standard [3] is applied to the level concerned. While recognizing that the EIA-632 standard is applied in diverse industries with success, it is certainly well suited for developing strategies such as simulation, prototyping, and benchmarking for resolving uncertainties and optimization issues in the building design.

3.4 Development and Implementation Issues

For the analysis and design of integrated control systems for building performance applications, a number of issues can be addressed, among them are the followings: 1) Although most building HVAC&R equipment and lighting components have been developed with an idea of improved performance, their developed control systems that regulate their capacities in buildings are basic and not established properly. As a result, it requires developing a systematic knowledge in the best use of their potential benefits through appropriate control systems of their performance characteristics under transient climatic conditions and occupants needs. 2) When building HVAC&R equipment and lighting components are used in a building, the simple addition of individual best performances does not warrant the best performance of the entire building. As a result, it requires developing control systems through integrated SE concept in order to provide improved benefits to buildings. 3) When requiring simulating a building performance application and its control system is that frequently certain building equipment and/or components can be modeled in one simulation software while some models are only available in other simulation environment. Hence, there is on the one hand domain specific building performance simulation, which is usually relatively basic in terms of control modeling and simulation capabilities (e.g. ESP-r). On the other hand, there is a domain dependent control modeling environments, which is very advanced in control modeling and simulation features (e.g. Matlab/Simulink), but still limited in building simulation. Marrying the two approaches by run-time coupling would potentially enable building performance assessments by predicting the overall effects of innovative control systems in a building indoor environment [13].

3.4.1 Development Lifecycle

The development lifecycle, in this study, includes several phases during which utilized software tools for BPS and CME must work together separately. In effect, the SE concept, which specifies functions, sequence and the interrelationships between various phases of both the building model and its remote control system, is extremely imperative so that their integration into the same entity will be successful. Although, there exists a number of lifecycle diagrams (spiral, waterfall, V, etc.), the V diagram is used in this study for the application of SE concepts to the development of complex projects. The V diagram is based on the interactions that take place between the links of decomposition or analysis (\) and construction, which means physical integration, or synthesis (/) of the design project [16]. When applying the cycle V, all the components of the project, as shown in figure 4, are deduced by decomposing and reconstructing the design concept.

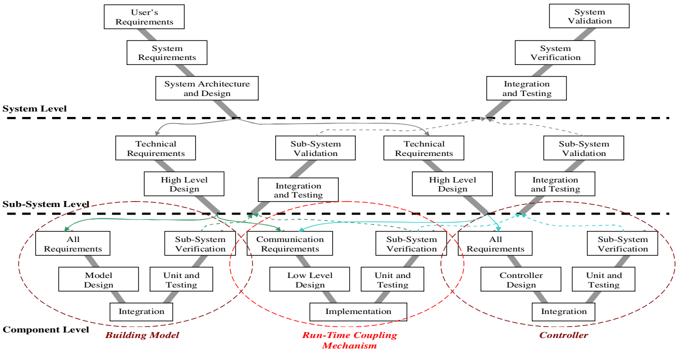


Fig. 4 Development lifecycle model

At top level, the building is represented in the total form of the V diagram, in which at low level this building consists of two sub-systems that stand for the building model and its remote control system, which at lower level are represented separately to work together by run-time coupling. As in BACS architecture, building HVAC&R equipment and lighting components are located in the field level and their control systems are located in the automation level, and they exchange data through a network. By similarity to BACS architecture, a distributed dynamic simulation mechanism was developed and implemented between BPS and CME in order to simulate building control applications, as happen in a real situation. While building models (including zones, HVAC&R equipment and lighting components) are built on ESP-r, their remote control systems are modeled on Matlab/Simulink. Both PBS and CME can run either on the same computer or on different computers connected by a network. . In addition, both software environments should exchange data by run time coupling in order to obtain simulation results as accurate as possible. In consequence, there are three V diagrams at component level, as shown in figure 4, where the V diagram in between concerns run time coupling between a building model and its control system [17].

3.4.2 Distributed Dynamic Simulation Mechanism

The most critical issues facing the design of run-time coupling between Matlab/ Simulink and ESP-r include heterogeneity, interoperability, and parallelism. In previous work [13], [14], [15], run-time coupling between ESP-r and Matlab/ Simulink was developed based on the development of an Inter-process Communication (IPC) mechanism using Internet sockets. This performs a distributed simulation using network protocols including transmission control protocol (TCP) and user datagram protocol (UDP) by exchanging data between building models and their control systems, as occurs in BACS architecture.

During simulation, commands and data are exchanged between a building model built ESP-r and its remote control system modeled on Matlab/Simulink

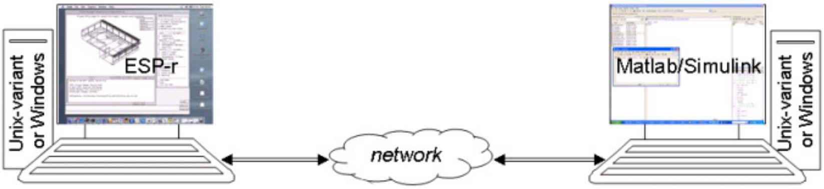


Fig. 5 A distributed dynamic mechanism for building control and performance simulation

at every time step. Furthermore, the iteration continues with the same way until the simulation is completed. Figure 5 shows how a distributed dynamic simulation mechanism is implemented between ESP-r and Matlab/Simulink over a network.

In addition, this dynamic mechanism of distributed building control and performance simulation is also implemented to support data exchange in ASCII, bi-nary, and XML formats as well as synchronous, partially synchronous and asynchronous communications modes.

3.4.3 Application of SE to Design of Control Systems for Buildings Automation

A very structured way of designing advanced control systems for building performance applications by run-time coupling between ESP-r and Matlab/Simulink is by involving SE practices in order to translate the occupant needs (or requirements) into a control system specification and realization that most efficiently meets these needs. Figure 6 shows how to design an advanced control system through a simple V-diagram while applying SE principles.

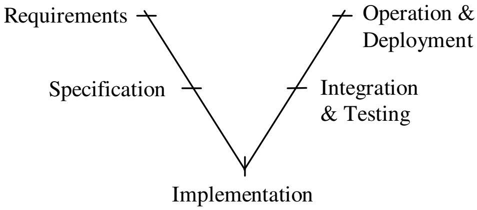


Fig. 6 A well-established way of designing advanced control systems

As shown in Figure 6, the V-diagram is simple as it consists of a small number of phases ranged from requirements to specification, implementation, integration and testing, and operation and deployment. The requirements phase defines the occupant needs and determines the building processes and/or dynamical models of plants to control and to be used for instance to derive requirements of the subsystem level of abstraction from requirements of the system level of abstraction.

4 Building Control Application

This case study concerns the test cell – an experimental room of 3.15*3.85*2.6 m3 located at Delft University of Technology – that is built with light construction materials in order to investigate different phenomena that influence the indoor environment of passive buildings. These phenomena include infiltration, radiant or solar heat gain and heating loss gains. Figure 7 shows a complete representation of the test-cell and its monitoring room. The test-cell contains several sensors and actuators, see Figure 7 (left), while its monitoring room is equipped with a PC and Data Acquisition, see Figure 7 (right). The PC is used for monitoring the actuators (i.e., HVAC equipment and lighting components) of the room from Matlab/ Simulink, and through a Data Acquisition that assures remote control process and data transfer bus connected to all sensors and actuators mounted in the test-cell.

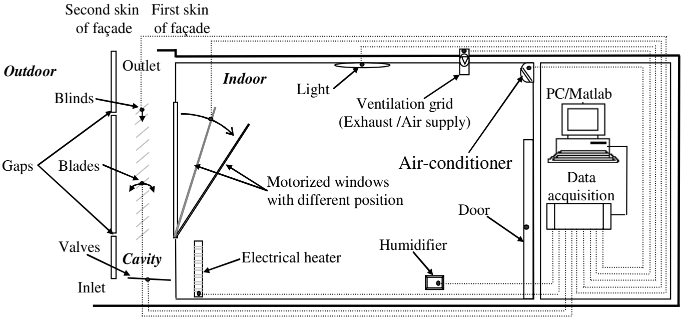


Fig. 7 The test-cell application

Reducing the energy use in buildings contributes to the reduction of greenhouse gas emissions. For this application, the heating plant is an electric heater of 1750 (W), the cooling plant is an air conditioner of about 60 (W). While the heating plant was set to operate when the indoor air-temperature went below 22°C and to stop when it reaches 22°C, the cooling plant was set to operate when the indoor air-temperature rose above 26°C and to stop when it reaches 24°C. As the experimental tests were performed during the winter period, the indoor air-temperature would not raise above 26°C. Therefore, this case of application concerns only the heating process. Hence, a requirements document for building heating process is well documented and described in [1], [5].

As shown in Figure 8, control system switches on the heating plant of the test-cell when the room air-temperature is below the heating set-point, and switches it off when the indoor air- temperature is above the heating set-points. Figure 9 shows a simple graphical environment of how an embedded control system modeled on Matlab/Simulink is run-time coupled to its building model built on ESP-r.

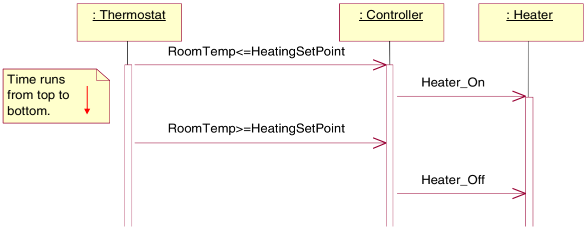

: Thermostat
: Controller
: Heater
Time runs from top to bottom.
RoomTemp<=HeatingSetPoint
Heater_On
RoomTemp>=HeatingSetPoint
Heater_Off

Fig. 8 Sequence diagram for building heating mode

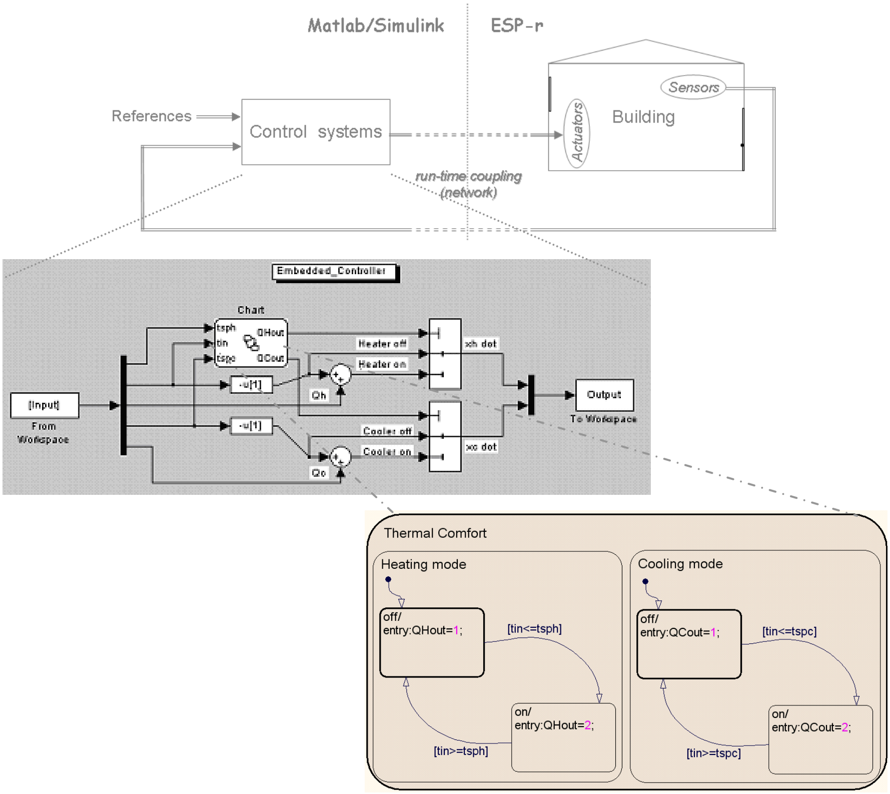


Fig. 9 An embedded control system for a building zone and plant model

The test-cell model, shown in Figure 7, is actually modeled on ESP-r with the real dimensions, construction properties and characteristics, and using the real climate file in simulation. Although the simulated results are obtained by run-time coupling between Matlab/Simulink and ESP-r, Matlab/Simulink is synchronously launched by ESP-r at every time-step as a separate process. Figure 10 shows the simulation results obtained for the test-cell's case study.

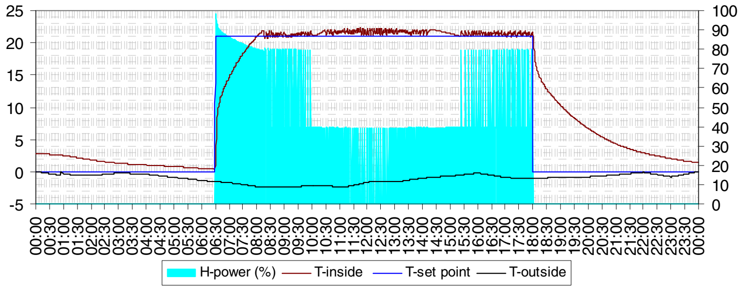


Fig. 10 Simulation results

The obtained simulation results show that there are small changes in the control system responses due to the climate data that highly influence the outside air-temperature. In winter time, the outside temperature influences the indoor air-temperature over time and the control system is not designed to suppress sensitive input noise causing small chattering at short intervals of few seconds. However, this control system can be enhanced to filter noises by an estimator to simultaneously cancel disturbances. In addition, hybrid systems can be easily coupled with any model-based modern control methods to eliminate disturbances [16].

5 Verification and Validation of a Distributed Simulation Mechanism

The initial work was on formalizing requirements for V&V of a distributed dynamic simulation mechanism. However, the V&V of such complex systems is not trivial as it requires developing the concepts from the V&V perspectives.

V&V automation is a "dream" among the test community. Most V&V are actually man-based. We cannot automate all the process, but there is a possibility through the use of specific methods in requirements. Two alternatives are proposed for formalizing requirements. The semi-formal approach is carried out with RDD-100 and Statemate based on the precedent notation. However, for critical systems, the use of formal method is the adequate alternative that is proposed; we use for trial the VDM method (Vienna Design Method).

V&V issues in the execution model proposed by the INCOSE requirement engineering working group. This model is defined in [9] as shown in Figure 11.

The co-simulation offered a more pragmatic approach in the sense the operative part is under simulation with a dedicated simulation tool and the control part with also a dedicated tool. Scenarios of simulation can be conducted with respect requirements of architects and civil engineering on monitoring and adjusting variable and systems parameters

- A resulting hierarchy of requirements composed of collections of requirements that correspond to various parts of a distributed simulation mechanism.

- Wherever a derived requirement exists some analysis was involved. Such analysis makes use of accumulated knowledge, namely shared tasks as run-time coupling between ESP-r and Matlab/Simulink.
- As requirements are generated or revised the flow of requirements to collections must be done in an orderly, gated manner. The development and implementation of run-time coupling between ESP-r and Matlab/Simulink was established in such a way to fulfill all the requirements involved. Some requirements in collections do not pass to other collections or analyses but instead are implemented. That is, for example, a part is built or tested, or assembled from lower level components of a distributed dynamic simulation mechanism.
- In addition to requirements themselves, SE concept deals with a hierarchy of feasibilities, possibilities, and queries that are associated with all requirements. If the requirements are considered to flow "downward" then these items flow "upward". As result, run-time coupling between ESP-r and Matlab/Simulink is designed using system-level synthesis.

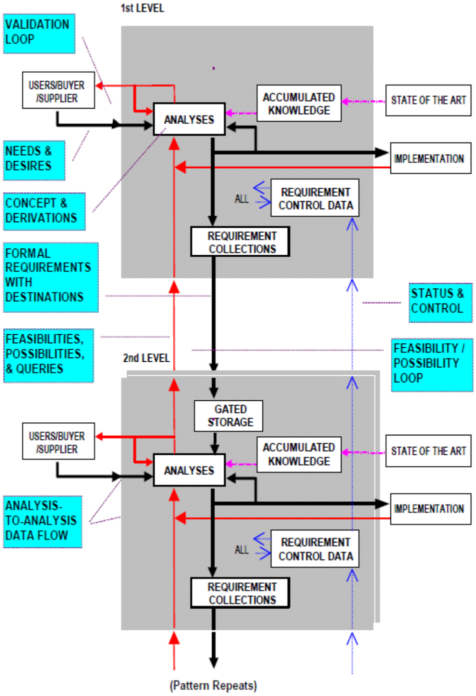


Fig. 11 Foregoing analysis of V&V issues of a distributed system

6 Conclusion

A system-level design of an embedded control system for a building performance application is presented and tested using a collaborative distributed dynamic simulation between ESP-r and Matlab/Simulink by run-time coupling. One of the main objectives of this study attempted to apply SE concept to the integration of advanced control systems in building environmental performance. This has been demonstrated by a procedural design approach to the development of embedded control systems for a building model. Therefore, the importance of using a distributed dynamic mechanism between ESP-r and Matlab/Simulink to enable the integration of control applications in building performance simulation by run-time coupling over TCP/IP has qualified that any control system can now be implemented and deployed for any integrated building model.

References

[1] Booch, G.: Object-Oriented Design with Applications. Benjamin Cummings, Redwood City CA (edition 1991)

[2] Energy Efficiency and Renewable Energy, the EERE's webpage (2010), http://www.eere.energy.gov/buildings/building_america/

[3] EIA-632, Processes for Engineering a System (ANSI/EIA-632), Proof Copy by Electronic Industries (edition 1998)

[4] DSMC, Systems Engineering Management Guide, Proof Copy by Defense Systems Management College, Fort Belvoir, VA (edition 1990)

[5] Hatley, D.J., Pirbhai, I.A.: Strategies for Real-Time System Specification. Technical Proceedings of Dorset House, New York (1988)

[6] Hoang, N., Jenkins, M., Karangelen, N.: Data Integration for Military Systems Engineering. In: Proceedings of IEEE Symposium & Workshop on Engineering of Computer Based Systems, USA (1996)

[7] Loureiro, G., Leaney, P.G., Hodgson, M.: A systems engineering environment for integrated automotive powertrain development. Society for Design and Process Science 34(41) (1999)

[8] Sahraoui, A.E.K., Buede, D., Sage, A.: Systems engineering research a roadmap. Journal of Systems Science and Systems Engineering 17(3), 319–333 (2008)

[9] Sahraoui, A.E.K., Jones, D.: V&V towards systems engineering framework. In: International Conference on Systems Engineering, Las Vegas (1999)

[10] Mathers, G., Simpson, K.J.: Framework for the Application of Systems Engineering in the Commercial Aircraft Domain. Report Version 1.2a, American Institute for Aeronautics and Astronautics, USA (2000)

[11] Shishko, R.: NASA Systems Engineering Handbook. Proof Copy by National Aeronautics and Space Administration, USA (edition 1995)

[12] Sheard, S.A., Lake, J.G.: Systems Engineering Standards and Models Compared. In: Proceedings of 8th Symposium of INCOSE, Vancouver, Columbia (1998)

[13] Yahiaoui, A., Hensen, J.L.M., Soethout, L.L.: Integration of control and building performance simulation software by run-time coupling. In: Proceedings of IBPSA Conference and Exhibition 2003, Netherlands, vol. 3, pp. 1435–1441 (2003)

[14] Yahiaoui, A., Hensen, J., Soethout, L.: Developing CORBA-based distributed control and building performance environments by run-time coupling. In: Proceedings of 10th ICCCBE, Germany (2004)

[15] Yahiaoui, A., Hensen, J.L.M., Soethout, L.L., Van Paassen, D.: Interfacing of control and building performance simulation software with sockets. In: Proceedings of IBPSA Conference and Exhibition Montreal, Canada (2005)

[16] Yahiaoui, A., Hensen, J., Soethout, L., Paassen, D.: Simulation based design environment for multi-agent systems in buildings. In: Proceedings of 7th Internat. Conference on System Simulation in Buildings, Belgium (2006)

[17] Yahiaoui, A.: A Systems Engineering Approach to Embedded Control System Implementation in Buildings. In: Proceedings of 18th Annual International Symposium of INCOSE, Netherlands (2008)

Chapter 21
An Organizing Taxonomy of Procedures to Design and Manage Complex Systems for Uncertainty and Flexibility

Michel-Alexandre Cardin

Abstract. This paper presents a five-phase taxonomy of procedures to support the design of complex engineering systems for uncertainty and flexibility. A review of major contributions was done to identify relevant procedures to support initial design generation, uncertainty recognition and modeling, concept generation and identification, design space exploration, and process management and representation. The taxonomy integrates contributions from surveys, articles, and books from the literature on engineering design, manufacturing, product development, and real options analysis obtained from professional e-index search engines. The organizing principles of the taxonomy were developed keeping in mind the intended user: the engineering designer. The taxonomy aims to provide guidance to designers in selecting appropriate tools at relevant design stages for both industry application and engineering education. It also aims to provide a framework to identify and organize ongoing research activities in this emerging research area.

1 Introduction

This paper presents a five-phase taxonomy of procedures to support the design of complex systems for uncertainty and flexibility. The taxonomy is geared specifically for engineering systems, such as large-scale telecommunications, defense, energy, housing, and transportation systems. Such systems are characterized by a high degree of technical complexity, social intricacy, and elaborate processes fulfilling important functions in society [1]. Dynamic socio-technical elements like markets, operational environment, regulations, and technology play a significant role in their success and/or failure [2]. Crucial decisions have to be made in early conceptual phases of the design, regarding strategic and long-term evolution.

Michel-Alexandre Cardin
Department of Industrial and Systems Engineering,
National University of Singapore
Block E1A, #06-25, 1 Engineering Drive 2, 117576, Singapore
e-mail: macardin@nus.edu.sg

The taxonomy hopes to provide guidance to designers in choosing the relevant design procedures to support conceptual design activities both in industry and academia. It aims to organize the latest research contributions into a unified and coherent framework. It also aims at guiding future research developments. This is done by building upon organizing principles from state-of-the-art design processes for uncertainty analysis and flexibility [3], as well taxonomies of design procedures in engineering, manufacturing, and product development [4].

Flexibility in design enables a system to change in the face of uncertainty [5]. It is associated to the notion of real options, providing the “right, but not the obligation, to change a project (or system) in the face of uncertainty” [6]. Real options exist “on” a system, involving higher-level managerial decisions like abandoning, deferring until favorable market conditions, expanding/contracting/reducing capacity, deploying capacity over time, switching inputs/outputs, and/or mixing the above [6]. Real options “in” a system are technical engineering components enabling options in deployment and operations [7]. Real options – also referred here as *flexible design concepts* – are characterized by a *strategy* (or type) and *enabler* in design (or mechanism) [8]. One example of a flexible engineering system is the HCSC building in Chicago [9]. In the 1990s during the economic uncertainty prevailing in the real estate market, this skyscraper was carefully designed to accommodate 27 additional stories on top of an initial vertical development. The real option to expand capacity would be exercised when there would be a need for additional office space. A few years ago the company exercised this expansion option, with the second phase completed in 2011.

2 Motivation

The main motivation is that designing complex systems for uncertainty and flexibility can improve lifecycle performance of complex systems significantly [3,10]. On the other hand, it is not an easy process to follow, and it is not widespread in industry and design education. It often requires guidance from industry lessons and recent research developments. For complex systems, it is not clear what uncertainty sources to address, where to focus the design effort for flexibility, how much flexibility is worth, how much it costs, and how much flexibility is enough.

There is a body of work providing the organizing principles for the proposed taxonomy. The structure of the taxonomy is inspired from the four-step process suggested by de Neufville and Scholtes [3]. It includes, however, several procedures to support concept generation and identification not presented by these authors (e.g. DSM and decision-based procedures). The methodology presented by Cardin et al. [11] is lengthy, so fewer organizing principles would be favored. The taxonomy by Tomiyama et al. [4] provides valuable organizing principles, but does not account explicitly for procedures focusing on uncertainty and flexibility. The reviews by Sethi and Sethi [12] focus mainly on manufacturing, and not complex engineering system as a whole. The survey by Saleh et al. [13] organizes the field of design for flexibility more thoroughly, but does not provide a clear account of existing procedures to support different phases of the design process.

3 Approach

The approach for developing the taxonomy first required a review of major research contributions to identify relevant procedures to support design for uncertainty and flexibility. A thorough review of surveys, articles, and books from the literature on engineering design [4,13], manufacturing [12], and real options [3,6,14] was done. Keywords like "flexibility in engineering design" and "real options in engineering" – or different combinations of individual words – were used in e-index search engines like Scopus, Engineering Village 2, and IEEE Xplore. Second, the organizing principles of the taxonomy were developed keeping the intended user in mind: the engineering designer. They emerged from a synthesis of an existing taxonomy of design procedures [4] and process for flexibility [3].

4 Taxonomy of Design Procedures

The phases of the taxonomy in **Fig.** 1 are detailed below. They represent the phases that designers should cover as they go through the design process for flexibility.

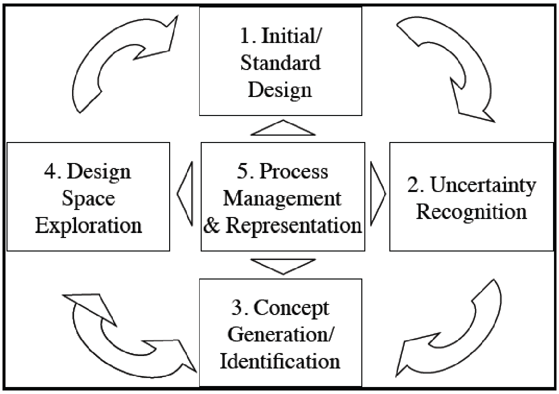

1. Initial/ Standard Design
4. Design Space Exploration
5. Process Management & Representation
2. Uncertainty Recognition
3. Concept Generation/ Identification

Fig. 1 Overview of the proposed taxonomy of design procedures for flexibility

4.1 Phase 1: Initial/Standard Design

Designing complex systems for uncertainty and flexibility starts from an existing design configuration (referred here as *baseline concept*). It is difficult to design flexibility in a system from scratch. Many design variables and parameters need to be considered, leading to a large number of possible design configurations. This situation is exacerbated when a wide range of uncertainty scenarios are considered. Most likely, no design configuration will be optimal for all possible scenarios [15]. In addition, there is a large number of decisions rule in the management of the system during operations, also adding to the complexity of the problem.

User Guidance

The aim of using the above procedures is to generate an initial design to work from. A procedure like Pahl and Beitz [16] provides a systematic way to create such design, relying on a step-by-step mechanism constructed from wide industry experience. TRIZ [17] provides principles based on a holistic review of successful product patents.

Designers interested in analyzing their engineering system for flexibility may not find current procedures best suited for this purpose. One reason is that many procedures do not account explicitly for uncertainty and flexibility. For example, one of the early steps in Pahl and Beitz is to define and freeze design and functional requirements from the customer domain as early as possible [16]. The design is then optimized for deterministic projections of future operating market conditions, requirements and constraints, even though those are prone to change [2,3,10]. Such approach pre-empts early considerations of flexibility.

4.2 Phase 2: Uncertainty Recognition

The procedures in phase 2 help designers identify and model major uncertainty sources affecting lifecycle performance. The reader is directed to the taxonomy by de Weck and Eckert [18] for more details. The authors suggest formal and practical approaches to quantify, characterize, and model uncertainty for more rigorous design analysis, based on the work by Halpern [19]. Formal approaches include standard probability theory, statistics, and Bayesian theory. Part of this group is the Dempster-Shafer theory, relying on separate pieces of evidence to construct the probability of events [20,21]. Possibility theory also provides an alternative to standard probability theory based on fuzzy logic [22]. Practical procedures to model uncertainty include diffusion models – mostly based on stochastic simulations – decision trees, and scenario planning [23].

User Guidance

Formal approaches are most useful to elicit, characterize, and quantify uncertainty. Because of a natural tendency to be over optimistic and confident about the future [24], these tools help designers consider explicitly optimistic as well as pessimistic scenarios. When historical data (e.g. demand, price) is available, statistical techniques are most useful, using for example regression to determine the mean growth rate and volatility of a stochastic process. It can be difficult, however, to determine the relevant data range for estimating parameters, the appropriate model to use (e.g. linear vs. polynomial), and appropriate stochastic models – e.g. geometric Brownian motion (GBM) vs. mean reversion. Probability, Bayesian, and possibility theories are useful when knowledge is available about the underlying phenomenon for inferences (e.g. calculation of joint probability distributions, prior probability updates of dam failure). When data is not readily available, designers may rely on expert knowledge to elicit probability distributions and scenarios, for instance using Dempster-Shafer or Delphi methods. Designers should be careful, however, not to bias questions when eliciting probability distributions and scenarios [24].

Practical approaches are most useful to represent stochastic processes over time. Although any approach can be used for continuous or discrete processes, diffusion models are better for continuous stochastic processes (e.g. market demand, price). Decision tree and scenario-planning methods are better suited for discrete events (e.g. government voting emission standards policy A vs. B).

4.3 Phase 3: Concept Generation and Identification

The procedures in this section provide guidance to determine where to focus the design effort for flexibility. Concept generation involves cognitive tasks to generate flexible strategies and design alternatives (i.e. flexible design concepts). These alternatives may be significantly different than the baseline design concepts generated in phase 1. Enabler identification is about identifying where to embed flexibility in the mechanical design and managerial processes. This will enable managers to exercise the flexibility strategies in future operations.

4.3.1 Concept Generation

Canonical Real Options Strategies
The real options literature provides six canonical strategies to help designers generate flexible design concepts [6]. One strategy is to defer capital investment until favorable market conditions arise. Another strategy is to stage asset deployment strategically over time instead of deploying all capacity at once. Altering operating scale, by expanding or contracting output production capacity, is another strategy. Abandoning a project doomed to fail with the possibility of reselling assets at salvage value is suggested. This value is often not accounted for in standard discounted cash flow (DCF) and net present value (NPV) analyses [6]. Switching production output and/or input can also bring performance improvements. Another possibility is to invest in R&D, which gives the right but not the obligation to capitalize on future technology and additional cash flows if successful.

<Mechanism, Type> Characterization
Mikaelian et al. [8] suggest a systematic approach based on the <mechanism, type> characterization of real options in enterprises. This favors generation of flexibility within and outside typical enterprise “silos” (e.g. strategy, process, product, IT). A *type* is akin to one of the canonical real option strategies above (e.g. expand, switch). A *mechanism* is an action, decision, or entity enabling the real option (an enabler in design). The process is initiated by focusing on major uncertainty sources affecting performance, and suggesting real option types to deal with these uncertainties. From there, the real option types are mapped to mechanism patterns (e.g. modularity, redundancy, buffering, staging) enabling the real options. The explicit mapping of real option types to different mechanism patterns stimulates flexibility generation.

Explicit Training and Prompting
Cardin [25] suggests a technique integrating a short lecture on the topic of flexibility with a structured prompting mechanism to guide teams of designers in creative

concept generation. The lecture helps designers be more aware of the effects of uncertainty on lifecycle performance. It describes generic sources of uncertainty, and explains why flexibility can improve lifecycle performance if considered early on. It discusses strategies for crafting valuable flexible design concepts, and provides real-world examples from industry. The prompting procedure helps scaffold the thought process for flexibility, building upon the structure in Fig. 1. This mechanism is simple, and useful to stimulate creativity in early collaborative activities [26]. Asking direct questions may trigger collective discussions more effectively than relying on industry guidelines and canonical strategies alone.

User Guidance

Canonical real option strategies provide useful checklists to generate flexible strategies and design concepts quickly. They do not provide, however, a systematic framework to determine the form this concept will take, depending on the system at hand. The approach based on the <mechanism, type> characterization brings more structure to this analysis. On the other hand, the first two methods are not crafted for direct use in the collaborative design process. They do not provide explicit support to designers to identify major uncertainty sources first, before eliciting flexibility strategies. The explicit training and prompting procedure integrates more systematically uncertainty thinking, real options strategies, and enabling in design within a collaborative design process. It builds upon collaboration engineering techniques to minimize productivity loss and stimulate creativity [27]. It is not yet clear, however, how to best use the technique in the design process. It could be used at the beginning of conceptual design activities, or periodically until the detailed design phase, reminding designers to consider these important issues.

4.3.2 Enabler Identification

Design Structure Matrix

The DSM (also called dependency structure matrix) introduced by Steward [28] can represent design tasks as a sequence of network interactions. A DSM is a square matrix where the rows and columns list all the relevant design and management components of a system [29]. The DSM encodes and represents graphically an engineering system. Matrix entries represent how the design and management components are connected, and how the information flows from one another. The framework has been studied to support flexibility in design. Change propagation analysis (CPA) and sensitivity DSM (sDSM) were developed to identify specific areas where to embed flexibility in complex systems design.

Change Propagation Analysis

CPA looks at change multipliers as potential areas to insert flexibility. These are design elements creating more change in other design variables then they absorb when a design or functional requirement is changed [30]. Making such variables more flexible reduces the amount of change created elsewhere in the design. CPA was applied enable flexibility in the car manufacturing process [30].

Sensitivity DSM
The sDSM is similar conceptually to CPA. It looks at design variables that are most sensitive to changes in design and functional requirements as areas to insert flexibility [31]. sDSM provides a high-level view of the design representation, "zooming out" from details to focus on the important design elements where to insert flexibility. The approach was applied for offshore oil platform design [31].

User Guidance
One benefit of CPA and sDSM is to provide systematic algorithms to identify areas where to embed flexibility. They are also better for a detailed analysis of an engineering system. On the downside, these methods are not as good for a quick first-pass analysis. This is because they require detailed data collection and expert interviews to build a DSM first, which can take a long time. Also, they require defining a bound for the engineering system. This means that designers may miss "low-hanging fruit" opportunities for flexibility because their effort is focused on the established DSM boundary, rather than thinking "outside the design box".

4.4 Phase 4: Design Space Exploration

After phase 3, designers explore the design space for the most valuable design configurations and management decision rules to operate the system (i.e. decide when it is appropriate to exercise flexibility). In the first subsection, procedures are described to evaluate quantitatively the lifecycle performance of each design concept. The real options literature stresses the importance of doing this to decide whether flexibility is worth the additional cost and design effort. In the second subsection, procedures are described to find the most valuable flexible design alternatives. Given engineers often work with high-fidelity models taking a long time to run, there is a need for computationally efficient and systematic procedures to explore the design space for flexibility [3].

4.4.1 Quantitative Concept Evaluation

Evaluation procedures from the financial and real option literature [6,32] have been adapted for engineering design. This is because many techniques build upon economic assumptions not necessarily holding in an engineering context [7]. Procedures build upon practical uncertainty modeling techniques like decision trees and simulation. They may rely on economic (e.g. NPV) and non-financial metrics (e.g. utility). They are used with optimization and design of experiments (DOE) techniques to rank order design alternatives efficiently and systematically.

Decision Analysis
A decision tree must be created to capture uncertainty scenarios and associated decisions over time. A folding back process based on dynamic programming (DP) is used to determine the best design decisions at each stage. Creating the tree is done from left to right. Analyzing the tree is done from right to left following the DP-based folding back process. The best payoff is taken at each decision point at time t_1, and the expected payoffs for the flexible ($E[\text{Payoff}]_{\text{Flexible}}$) and

($E[Payoff]_{Inflexible}$) are calculated using the probability assignments. The expected value of flexibility consists of the difference between the two expected payoffs.

Simulations
Here the stochastic scenarios and decisions enabled by a particular design are modeled explicitly. For example, de Neufville and Scholtes [3] used Monte Carlo simulation to value the flexibility to expand the capacity of a *n*-level parking garage design, if and only when demand exceeds capacity for two consecutive years. The starting point is typically a standard performance analysis (e.g. using NPV) of the baseline design concept (e.g. a fixed five or six-level design) using deterministic projections (e.g. demand). Uncertainty is then incorporated explicitly via stochastic simulations over the project lifetime. After this, flexibility is incorporated in the model, and valued using simple logical statements (e.g. IF, ELSE, etc.) to represent different decision rules (e.g. expand by one level if capacity exceeds demand for two consecutive years). For each scenario, the model computes a NPV outcome under the flexible design and decision rules incorporated in the model. The distributions for the fixed and flexible designs can be compared explicitly. Central (e.g. mean) and dispersion (e.g. standard deviation) measures can be computed and compared between design alternatives to support decision-making for different risk profiles (e.g. risk-averse, risk-neutral, risk-seeking).

User Guidance
Decision analysis is useful as a quick first-pass analysis to go through explicitly the exercise of recognizing and representing different uncertainty scenarios, with possible performance outcomes. Given the relatively simple DP-based valuation mechanism, it is helpful to provide a quick assessment on the value of flexibility. The approach also provides much analytical freedom. Although better suited for discrete uncertainty sources, it can represent both discrete and continuous stationary and non-stationary stochastic processes. Different decisions can be used and evaluated at different stages, and many uncertainty sources can be considered at once. Decision analysis suffers, however, from the curse of dimensionality. The number of paths can explode quickly, making it difficult to go beyond two or three stages, even with a minimum of decision and chance outcomes.

Monte Carlo simulations provide more freedom in terms of uncertainty sources, decision rules, design variables, and parameters that can be modeled than decision and lattice analyses. It is most useful for a deeper valuation of flexible design concepts. On the other hand, it can be more demanding computationally, especially when a high fidelity model of the system is used. The procedures described in the next section are designed to help alleviate this computational issue.

4.4.2 Computationally Efficient Search

These procedures provide efficient search algorithms to explore systematically the design space for the most valuable flexible design configurations. This space can be prohibitively large, and even be computationally intractable because so many alternatives and uncertainty sources exist [2].

Screening Models

Screening models rely on optimization algorithms and DOE techniques to search the design space efficiently for interesting candidate designs. There are three types of screening models: bottom-up, simulators, and top-down [3]. Bottom-up screening models use a simplified version of a complex, detailed design model. Simulators incorporate statistical techniques (e.g. response surface methodology) and/or fundamental principles to mimic the response of the detailed model. Top-down screening models use representations of major relationships between the parts of the system to understand possible system responses. Example applications in the context of flexibility include hydroelectric dam design in China [33], offshore oil platform design [34], maritime systems [35,36], and car manufacturing [37].

Catalogue of Flexible Operating Plans Procedure

In some cases, it may not be possible or desirable to reduce model fidelity to explore the design space. Cardin [38] proposed an approach based on selecting a small set of representative scenarios of uncertainty. Each scenario is then associated with the best flexible design configuration for this scenario – called an *operating plan* – found using the adaptive One-Factor-at-A-Time method [39], thus creating a *catalogue of flexible operating plans*. This approach limits the number of optimizations and simulations to run using a high-fidelity model. It was applied to the analysis of flexible mining operations [15].

User Guidance

All procedures above are useful to designers when it is not clear how to structure the search for the best flexible design configurations, and when computational efficiency is an issue. Screening models and the operating plans procedures are most useful when quantitative performance metrics are involved, whether financial (e.g. NPV) or non-financial (e.g. CO_2 emission levels, service rate). Screening models are useful when the goal is to reduce model complexity. The modeling resolution loss is an obvious drawback, not guaranteeing an absolute optimal solution to be found. This may be an acceptable trade-off, however, when it is infeasible to screen the entire design space. The catalogue of flexible operating plan procedure is useful when it is not possible or desirable to reduce model complexity. One difficulty, however, is to select a representative set of uncertainty scenarios.

4.5 Phase 5: Process Management and Representation

Managing the design process for flexibility can be difficult because the process involves many stakeholders at different hierarchical levels. A typical decision requires inputs from decision-makers to determine what the system should accomplish, and for the ultimate go-no-go decisions. Inputs are needed from marketing/economists who are aware of market and other socio-economic considerations involving policy and regulation. Engineering designers need this information to craft better designs, and enable relevant flexibility strategies. They are also the ones most aware of technological capacity and uncertainty. Managers ultimately decide when it is appropriate to exercise the flexibilities, given operating conditions and budgetary constraints.

Procedures in phase 5 focus on these concerns, and are useful during all phases of the design process. They support collaborative interactions between designers, and represent an area widely opened for novel research contributions. Not much has been done to develop procedures to understand agency problems and asymmetries, support the design process from conceptual design to implementation, operations, and disposal, and to represent the process and/or engineering system. This section provides an overview of candidate procedures and research methodologies, suggesting they could be adapted specifically for this purpose.

Collaboration Engineering

Collaboration engineering "studies ways of designing recurring collaboration processes that can be transferred to groups that can be self-sustaining in these processes using collaboration techniques and technology" [40]. It is motivated by the fact that collaboration may sometimes put barriers to creativity, resulting in productivity loss [41]. Evaluation apprehension (fear of being judged), free riding (letting others do the work), and production blocking (losing an idea because someone else is talking) are potential causes of productivity loss [42]. Group Support System (GSS) technology helps minimize productivity loss, and stimulates creativity [27,43]. GSS is defined as "socio-technical systems consisting of software, hardware, meeting procedures, facilitation support, and a group of meeting participants engaged in intellectual collaborative work" [44]. GSS is useful to record discussion data, structure the collaborative process, and structure moderation.

Serious Gaming

Serious gaming provides an interesting research platform to study the process and management of designing for uncertainty and flexibility. As explained by Ligtvoet and Herder [45], serious games are "experience-focused, experimental, rule-based, interactive environments where participants learn by taking actions and by experiencing their effects through feedback mechanisms that are deliberately built into and around the game" [46]. Mostly used in business schools [47], these techniques can be useful in engineering to understand the cognitive dynamics of design decision-making, as well as agency and information asymmetries arising during the system lifecycle. Sterman's beer game [48] was used to highlight the bullwhip effect in supply chain management. Lessons and issues in design for flexibility could be modeled and studied more systematically using such techniques.

5 Discussion

The proposed taxonomy provides benefits to industry practitioners and engineering educators along three dimensions. This section discusses first the benefits derived from using the taxonomy as a systematic process to extract value from uncertainty and improve lifecycle performance. Second, it provides an example application in a case study, showing how other authors have followed this process to assess lifecycle performance improvement in a real case. Third, the section discusses how the taxonomy can be used to organize ongoing research contributions.

5.1 Extracting Value from Uncertainty

The main assumption in this paper is that flexibility can improves expected lifecycle performance of a complex system by extracting value from uncertainty. This is done by physically enabling a system to reduce the effects from downside conditions – like buying insurance – and positioning the system to capture upside opportunities – like buying a call option on a stock. Improving both worst and best possible outcomes shifts the distribution of outcomes towards better overall value outcomes, with the net effect of improving the expected lifecycle performance. Nevertheless, it is not sufficient to desire flexibility in a system. One needs to enable it in the design concretely and manage it in operations. The procedures elicited in phases 1-4 help designers do this systematically in early conceptual analysis. Phase 5 provides procedures and techniques to help teams of designers interact together to select design concepts or recognize the different asymmetries between stakeholders that may reduce the value of flexibility in operations.

5.2 Example Application

The taxonomy in **Fig.** 1 can be used as a systematic process to design complex systems for uncertainty and flexibility. The following example illustrates the design thinking associated with each phase of the process. It also shows how a particular procedure can be selected in each phase based on the guidance provided above. The example is based upon the paper by de Weck et al. [49], who revisited the Iridium case study to evaluate whether flexibility could have improved the system lifecycle cost (LCC). Each analytical decision is re-casted in the logical analytical decisions made by the authors.

In phase 1, the authors determined the baseline design concept for the LEO satellite constellation based on standard industry practice, trying to match Iridium's publicly available target communication capacity and LCC. Their analysis led to a baseline concept of fifty satellites along five circular polar orbits, altitude of 800 km, and elevation angle of 5°. Communication capacity would be for 80,713 duplex channels. Assuming a 10-years lifecycle, 10% discount rate, 3 million users, and average monthly activity of 125 minute/month, the expected lifecycle cost of such architecture would be $2.01 billion, very close to the quoted development cost. In phase 2, the authors recognized market demand as the main uncertainty driver of LCC performance. Because demand is a continuous diffusion process, they used binomial lattice as modeling tool, assuming GBM. In phase 3, they relied on canonical real option strategies to identify quickly a flexible staged deployment strategy to deal with demand uncertainty. They recommended deploying additional capacity only when economic conditions are favorable, instead of rapidly deploying the constellation as done by Iridium. To identify quickly the relevant design variables to enable this flexible strategy, they screened qualitatively each design variable. They determined that altitude and elevation angle were the design variables that could be adjusted to accommodate different staged deployment strategies. In essence, their strategy required the ability to add more satellites, and move them on-orbit to increase coverage capacity. This gave rise to a design radically different from the baseline concept optimized for fixed capacity in phase 1. This required a satellite constellation capable of changing orbital configuration. In phase 4, the authors explored the design tradespace using optimization for the best ways of staging and deploying the flexible engineering system under demand uncertainty. Since a quantitative performance LCC metric was in-use, standard optimizations techniques were favored over a utility-based framework. Each flexible design concept was evaluated using lattice analysis and optimizations enabled finding the best flexible design concept. It was found that the optimal design path would start with twenty-eight satellites over four orbital planes at 1,600 km altitude and 5° elevation, converging towards a full three hundred and sixty-four satellite constellation over 14 orbital planes, 800 km altitude, and 35° elevation. In essence, the analysis led to a 1) a different design than originally planned, and 2) significant expected LCC improvement from $2.01 billion down to $1.46 billion. This showed that, on average, the new design would require less investment while providing the same level of communication capability.

5.3 Guidance for Future Research

The benefits and drawbacks identified in each phase provide guidance for future research. Phases 3 and 5 in particular provide a rich environment for novel research contributions. In phase 3, rigorous case-base or experimental comparisons could be made between existing procedures for concept generation and enabler identification. More research is needed to develop new procedures or adapt existing ones – see review in Shah et al. [50] – for flexible design concept generation. More work is required in phase 5 to understand the cognitive dynamics, agency problems, and issues of information asymmetries that render difficult the application of flexibility principles in practice. An obvious drawback is that the proposed taxonomy may not be the only one possible. There are many ways to organize existing and future research contributions on uncertainty and flexibility.

6 Conclusion

This paper presented an organizing taxonomy of design procedures to support the design of complex systems for uncertainty and flexibility. Explicit considerations of uncertainty and flexibility can lead to radically different designs offering on average better lifecycle performance [3,14]. The taxonomy involves five phases: 1) initial/standard design, 2) uncertainty recognition, 3) concept generation and identification, 4) design space exploration, and 5) process management and representation. It is geared specifically for complex engineering systems, for example in the defense, energy, housing, telecommunications, and transportation industries. It gathers design procedures from the literature on engineering design, manufacturing, product development, and real options analysis. The organizing principles are based on state-of-the-art processes to design for uncertainty and flexibility [3]. The suggested taxonomy addresses the need to organize recent developments in this emerging research area into a unified framework. It also provides an organizing framework to identify current and future research opportunities.

Acknowledgments. This project was supported by the Faculty of Engineering and the Department of Industrial and Systems Engineering (ISE) at the National University of Singapore (NUS) via grant WBS R-266-000-061-133, and the Singapore-MIT Alliance for Research and Technology via grant WBS R-266-000-069-592. The author thanks MIT Profs. Richard de Neufville and Olivier de Weck for their intellectual support, and the help of Mehdi Ranjbar Bourani.

References

1. ESD, Engineering Systems Division Strategic Report. Massachusetts Institute of Technology, Cambridge, MA, United States (2011)
2. Braha, D., Minai, A.A., Bar-Yam, Y.: Complex Engineered Systems: Science Meets Technology. Springer, Netherlands (2006)
3. de Neufville, R., Scholtes, S.: Flexibility in Engineering Design. Engineering Systems. MIT Press, Cambridge (2011)
4. Tomiyama, T., Gu, P., Jin, Y., Lutters, D., Kind, C., Kimura, F.: Design Methodologies: Industrial and Educational Applications. CIRP Annals – Manufacturing Technology 58, 543–565 (2009)
5. Fricke, E., Schulz, A.P.: Design for Changeability (DfC): Principles to Enable Changes in Systems Throughout Their Entire Lifecycle. Systems Engineering 8(4), 342–359 (2005)
6. Trigeorgis, L.: Real Options. MIT Press, Cambridge (1996)
7. Wang, T., de Neufville, R.: Real Options 'In' Projects. Paper presented at the Real Options Conference, Paris, France (2005)
8. Mikaelian, T., Nightingale, D.J., Rhodes, D.H., Hastings, D.E.: Real Options in Enterprise Architecture: A Holistic Mapping of Mechanisms and Types for Uncertainty Management. IEEE Transactions on Engineering Management 54(3), 457–470 (2011)
9. Guma, A., Pearson, J., Wittels, K., de Neufville, R., Geltner, D.: Vertical Phasing as a Corporate Real Estate Strategy and Development Option. Journal of Corporate Real Estate 11(3), 144–157 (2009)
10. Eckert, C.M., de Weck, O.L., Keller, R., Clarkson, P.J.: Engineering Change: Drivers, Sources and Approaches in Industry. Paper presented at the 17th International Conference on Engineering Design, Stanford, CA, United States (2009)
11. Cardin, M.-A., Nuttall, W.J., de Neufville, R., Dahlgren, J.: Extracting Value from Uncertainty: A Methodology for Engineering Systems Design. Paper presented at the 17th Symposium of the International Council on Systems Engineering, San Diego, CA, United States (2007)
12. Sethi, A.K., Sethi, S.P.: Flexibility in Manufacturing: A Survey. The International Journal of Flexible Manufacturing Systems 2, 289–328 (1990)
13. Saleh, J.H., Mark, G., Jordan, N.C.: Flexibility: a Multi-Disciplinary Literature Review and a Research Agenda for Designing Flexible Engineering Systems. Journal of Engineering Design 1, 1–17 (2008)
14. Nembhard, H.B., Aktan, M.: Real Options in Engineering Design, Operations, and Management. Taylor & Francis, Boca Raton (2010)

15. Cardin, M.-A., de Neufville, R., Kazakidis, V.: A Process to Improve Expected Value in Mining Operations. Mining Technology: IMM Transactions, Section A 117(2), 65–70 (2008)
16. Pahl, G., Beitz, W.: Engineering Design. Springer Design Council, London (1984)
17. Altshuller, G.: The Innovation Algorithm. Technical Innovation Center, Worcester, MA, United States (1973)
18. de Weck, O.L., Eckert, C.: A Classification of Uncertainty for Early Product and System Design. ESD Working Paper Series. Massachusetts Institute of Technology, Cambridge (2007)
19. Halpern, J.Y.: Reasoning about Uncertainty. MIT Press, Cambridge (2003)
20. Shafer, G.: A Mathematical Theory of Evidence. Princeton University Press, Princeton (1976)
21. Dempster, A.P.: Upper and Lower Probabilities Induced by a Multivalued Mapping. The Annals of Mathematical Statistics 38(2), 325–339 (1967)
22. Zadeh, L.: Fuzzy Sets as the Basis for a Theory of Possibility. Fuzzy Sets and Systems 1, 3–28 (1978)
23. Helmer-Hirschberg, O.: Analysis of the Future: The Delphi Method. RAND Corporation (1967)
24. Morgan, M.G., Henrion, M.: Uncertainty: A Guide to Dealing with Uncertainty in Quantitative Risk and Policy Analysis. Cambridge University Press, United Kingdom (1990)
25. Cardin, M.-A.: Quantitative Performance-Based Evaluation of a Procedure for Flexible Design Concept Generation. Doctoral Dissertation in Engineering Systems. Massachusetts Institute of Technology, Cambridge (2011)
26. Santanen, E.L., Briggs, R.O., de Vreede, G.-J.: Causal Relationships in Creative Problem Solving: Comparing Facilitation Interventions for Ideation. Journal of Management Information Systems 20(4), 167–197 (2004)
27. Nunamaker, J.F., Briggs, R.O., Mittleman, D.D., Vogel, D.R., Balthazard, P.A.: Lessons from a Dozen Years of Group Support Systems Research: A Discussion of Lab and Field Findings. Journal of Management Information Systems 13(3), 163–207 (1997)
28. Steward, D.V.: The Design Structure System: A Method for Managing the Design of Complex Systems. IEEE Transactions on Engineering Management 28, 71–74 (1981)
29. Browning, T.R.: Applying the Design Structure Matrix to System Decomposition and Integration Problems: A Review and New Directions. IEEE Transactions on Engineering Management 48(3), 292–306 (2001)
30. Suh, E.S., de Weck, O.L., Chang, D.: Flexible Product Platforms: Framework and Case Study. Research in Engineering Design 18, 67–89 (2007)
31. Kalligeros, K.: Platforms and Real Options in Large-Scale Engineering Systems. Doctoral Dissertation in Engineering Systems. Massachusetts Institute of Technology, Cambridge (2006)
32. Cox, J.C., Ross, S.A., Rubinstein, M.: Options Pricing: A Simplified Approach. Journal of Financial Economics 7(3), 229–263 (1979)
33. Wang, T.: Real Options in Projects and Systems Design – Identification of Options and Solutions for Path Dependency. Doctoral Dissertation in Engineering Systems. Massachusetts Institute of Technology, Cambridge (2005)
34. Lin, J.: Exploring Flexible Strategies in Engineering Systems Using Screening Models – Applications to Offshore Petroleum Projects. Doctoral Dissertation in Engineering Systems. Massachusetts Institute of Technology, Cambridge (2009)

35. Buurman, J., Zhang, S.X., Babovic, V.: Reducing Risk Through Real Options in Systems Design: The Case of Architecting a Maritime Domain Protection System. Risk Analysis 29(3), 366–379 (2009)
36. Taneja, P., Aartsen, M.E., Annema, J.A., van Schuylenburg, M.: Real Options for Port Infrastructure Investments. Paper presented at the Third Conference on Infrastructure Systems and Services: Next Generation Infrastructure Systems for Eco-Cities, Shenzhen, China (2010)
37. Yang, Y.: A Screening Model to Explore Planning Decisions in Automotive Manufacturing Systems Under Demand Uncertainty. Doctoral Dissertation in Engineering Systems. Massachusetts Institute of Technology, Cambridge (2009)
38. Cardin, M.-A.: Facing Reality: Design and Management of Flexible Engineering Systems. Master of Science Thesis in Technology and Policy, Massachusetts Institute of Technology, Cambridge, MA, United States (2007)
39. Frey, D.D., Wang, H.: Adaptive One-Factor-at-a-Time Experimentation and Expected Value of Improvement. Technometrics 48(3), 418–431 (2006)
40. de Vreede, G.-J., Briggs, R.O.: Collaboration Engineering: Designing Repeatable Processes for High-Value Collaborative Tasks. Paper presented at the Proceedings of the 38th Annual Hawaii International Conference on System Sciences, Big Island, HI, United States (2005)
41. Mullen, B., Johnson, C., Salas, E.: Productivity Loss in Brainstorming Groups: A Meta-Analytic Integration. Basic and Applied Social Psychology 12(1), 3–23 (1991)
42. Nijstad, B.A., Stroebe, W., Lodewijkx, H.F.M.: Production Blocking and Idea Generation: Does Blocking Interfere with Cognitive Processes? Journal of Experimental Social Psychology 39(6), 531–548 (2003)
43. Bostrom, R.P., Nagasundaram, M.: Research in Creativity and GSS. Paper presented at the Proceedings of the 31st Annual Hawaii International Conference on System Sciences, Kohala Coast, HI, United States (1998)
44. de Vreede, G.-J., Vogel, D., Kolfschoten, G., Wien, J.: Fifteen Years of GSS in the Field: A Comparison Across Time and National Boundaries. Paper presented at the 36th Annual Hawaii International Conference on System Sciences, Big Island, HI, United States (2003)
45. Ligtvoet, A., Herder, P.M.: Simulation and Gaming for Understanding the Complexity of Cooperation in Industrial Networks. In: Hammami, O., Krob, D., Voirin, J.-L. (eds.) Complex Systems Design and Management. Springer, Heidelberg (2011)
46. Mayer, I.S.: The Gaming of Policy and the Politics of Gaming: A Review. Simulation and Gaming 40(6), 825–862 (2009)
47. Faria, A., Hutchinson, D., Wellington, W.J., Gold, S.: Developments in Business Gaming: A Review of the Past 40 Years. Simulation and Gaming 40(4), 464–487 (2009)
48. Sterman, J.D.: Modeling Managerial Behavior: Misperceptions of Feedback In a Dynamic Decision Making Experiment. Management Science 35(3), 321–339 (1989)
49. de Weck, O.L., de Neufville, R., Chaize, M.: Staged Deployment of Communications Satellite Constellations in Low Earth Orbit. Journal of Aerospace Computing, Information, and Communication 1, 119–136 (2004)
50. Shah, J.J., Kulkarni, S.V., Vargas-Hernandez, N.: Evaluation of Idea Generation Methods for Conceptual Design: Effectiveness Metrics and Design of Experiments. Journal of Mechanical Design 122, 377–384 (2000)

Author Index