Introduction

Massimo Cossentino, Ambra Molesini, Vincent Hilaire, and Valeria Seidita

Nowadays, software engineers face a wide range of particular application domains such as electronic commerce, enterprise resource planning, mobile computing, self-organisation, pervasive and adaptive computing, etc. The resulting heterogeneity and required functionalities call for complex systems and open architectures that may evolve dynamically over time so as to accommodate new components and meet new requirements. This is probably one of the main reasons why the agent metaphor and agent-based computing are gaining momentum in these areas.

As far as software engineering is concerned, the key implication is that the design and development of software systems according to a (new) paradigm can by no means rely on conceptual tools and methodologies conceived for a totally different (old) paradigm [19]. In this book we do not want to enter the debate about the differences in the definition of *methodology* and *design process* or simply *process*. For the aim of this book we may consider them as synonymous.

Even if it is still possible to develop a complex distributed system in terms of objects and client–server interactions, for example, such a choice appears odd and ill-adapted when the system is a Multi-agent System (MAS) or can be assimilated to an MAS. Rather, a brand new set of conceptual and practical tools—

M. Cossentino
Istituto di Calcolo e Reti ad Alte Prestazioni, Consiglio Nazionale delle Ricerche, Palermo, Italy
e-mail: cossentino@pa.icar.cnr.it

A. Molesini
Alma Mater Studiorum – Università di Bologna Viale Risorgimento 2, 40136 Bologna, Italy
e-mail: ambra.molesini@unibo.it

V. Seidita (✉)
Dipartimento di Ingegneria Chimica, Gestionale, Informatica, Meccanica, University of Palermo, Palermo, Italy
e-mail: valeria.seidita@unipa.it

V. Hilaire
IRTES-SET, UTBM, UPR EA 7274 90 010 Belfort cedex, France
vincent.hilaire@utbm.fr

M. Cossentino et al. (eds.), *Handbook on Agent-Oriented Design Processes*,
DOI 10.1007/978-3-642-39975-6_1,

specifically suited to the abstractions of agent-based computing—is needed to facilitate, promote and support the analysis, design and development of MASs, and to fulfil the great general-purpose potential of agent-based computing.

The definition of agent-specific methodologies is definitely one of the most explored topics in Agent-Oriented Software Engineering (AOSE). A great number of special-purpose methodologies were developed in the last few years to tackle the challenges of specific application domains.

A methodology has several constituent parts, including a full life cycle process, a comprehensive set of concepts, a set of rules, heuristics and guidelines, a set of metrics, information on quality assurance, a set of coding and other organisational standards, techniques for reuse and project management procedures [5, 10, 12]. Among the different methodologies developed [2, 6, 11, 14–16, 18, 20] there is a great consent to organise the process life cycle according to two classical phases: Analysis and Design. However, apart from this consensus, the cited methodologies exhibit many differences. Indeed, and it is not the only difference, regarding the abstractions adopted for modelling the systems there is no agreement among the different methodologies: each methodology proposes an own set of abstractions for modelling the systems. Due to this missing agreement, AOSE methodologies typically define their abstractions through a metamodel [1, 7, 8] that details all the abstractions adopted by the methodology and their relationships. Even if a lot of abstractions are labelled in the same way in the different methodologies, these abstractions frequently present different meaning and may belong to different process phases. For example, the "agent" concept is always present in the AOSE methodologies, however in some methodologies such as [6,15,18] the agent appears since Analysis phase, while in other methodologies such as [2,16,20] "agent" is only a design concept.

A key concept both in traditional software engineering and in agent-oriented one is a common agreement that the "ideal methodology" [17] for any system does not exist. This means that the methodology must be adapted to the particular characteristics of the adopted usage scenarios. There are two major ways of adapting methodologies: tailoring (particularisation or customisation of a pre-existing methodology) or Situational Method Engineering (SME)[3,4]. SME is the discipline that studies the composition of new, ad hoc software engineering processes for each specific need. This is based on the assumption that the "situation" is the information leading to the identification of the right design approach. This paradigm provides tools for the construction of ad hoc processes by means of reuse of the existing portion of methodologies (called method fragments) stored in a repository.

In order to be effective, this approach requires the design process and its fragments to be suitably documented and modelled.

In this context, the work of the IEEE-FIPA Design Process Documentation and Fragmentation (FIPA-DPDF) Working Group [9] aims at providing the possibility of representing design processes and method fragments through the use of standardised templates, thus allowing the creation of easily sharable repositories and enabling an easier composition of new design processes. At the beginning of 2012 the Process Documentation Template became an IEEE FIPA standard specification with number SC00097B [13].

Following this standardisation work, this book gathers the documentations of some of the most known AOSE methodologies. Chapters describing the methodologies have been produced, in several cases, by the original authors of the methodology itself. In all cases, authors are FIPA-DPDF WG Members who with their effort also aimed at validating the effectiveness of the Process Documentation Template specification. In particular, the methodologies documented in this book are as follows:

- ADELFE—it is dedicated to applications characterised by openness and the need of system adaptation to an environment. Its main goal is to help and guide any designer during the development of an Adaptive Multi-agent System.
- ASPECS—it is a step-by-step requirement-to-code iterative development process for engineering Complex Systems using Multi-agent Systems and Holonic Multi-agent Systems.
- ELDAMeth—it is an agent-oriented methodology specifically designed for the simulation-based prototyping of distributed agent systems. It is centred on the ELDA agent model and on an iterative development process covering distributed agent system modelling, simulation and implementation for a target agent platform.
- Gaia—it focuses on the use of organisational abstractions to drive the analysis and design of MAS. Gaia models both the macro (social) aspects and the micro (agent internals) aspects of MAS, and it devotes a specific effort to model the organisational structure and the organisational rules that govern the global behaviour of the agents in the organisation.
- GORMAS—it is focused on designing large scale, open, service-oriented Organisation-Centred Multi-agent Systems, where organisations are able to accept external agents into them. GORMAS is based on a specific method for designing human organisations.
- INGENIAS—it covers the full development cycle of Multi-agent Systems. It includes support for requirements elicitation (basic), analysis, design, coding and testing. It is intended for general use, that is, with no restrictions on application domains.
- INGENIAS-Agile—it introduces the definition of an agile process for the INGENIAS methodology according to a well-known development process: Scrum. The process adopts the iterative and fast plan presented originally by the methodology and uses some of the activities and most of the work products of the INGENIAS proposal with the Unified Development Process.
- O-MaSE—it is a new approach in the analysis and design of agent-based systems, being designed from the start as a set of method fragments to be used in a method engineering framework. The goal of O-MaSE is to allow designers to create customised agent-oriented software development processes.
- OpenUp—it is an open source process framework developed within the Eclipse Foundation. It aims at enabling an easy adoption of the core of the RUP/Unified Process. It proposes an iterative and incremental approach within a structured life cycle. OpenUp embraces a pragmatic, agile philosophy that focuses on the collaborative nature of software development; this is the unique not agent-oriented

methodology reported in this book and it demonstrates the feasibility of the adopted Process Documentation Template even in the OO context.

- PASSI—it is a step-by-step requirement-to-code methodology for designing and developing multi-agent societies integrating design models and concepts from both OO software engineering and artificial intelligence approaches using the UML notation.
- ROMAS—it defines a set of activities for the analysis and design of regulated open multi-agent systems. The defining characteristics of systems of this kind are that they are social, open and regulated.
- SODA—it deals with MAS analysis and design, and focuses on critical issues such as agent coordination and MAS–environment interaction. SODA adopts the Agents & Artifacts meta-model, and it introduces a *layering principle* as an effective tool for scaling with system complexity.
- Tropos—it is a comprehensive, agent-oriented methodology for developing socio-technical systems. Such systems explicitly recognise the existence of and interplay between technical systems (software) and social actors (humans and organisations).

The book is organised as follows:

- Chapter "The IEEE-FIPA Standard on Design Process Document Template" presents the motivations and some basic concepts of the Process Documentation Template.
- Chapters "ADELFE 2.0" to "The OpenUp Process" present (with the exception of report) the documentation of the AOSE methodologies.
- Chapter "ROMAS Methodology" presents the documentation of a non-AOSE methodology (OpenUP).

Finally, to conclude this short introduction, we would like to highlight that this book represents a collective effort of the IEEE-FIPA DPDF Working Group, which would not have been possible to realise without the contributions of a number of individuals, to whom we are deeply grateful. In particular, we thank all the authors who accepted to contribute to this book and who took effort to contribute original chapters presenting their methodologies and design processes according to the Process Documentation Template. We would also like to thank all the other WG members for their contribution to the definition of the IEEE FIPA SC00097B specification. Finally, we would like to thank Prof. J. Ferber for having kindly written his interesting preface to this book.

We all hope you will enjoy reading this book.

References

1. Bernon, C., Cossentino, M., Gleizes, M.P., Turci, P., Zambonelli, F.: A study of some multi-agent meta-models. In: Odell, J., Giorgini, P., Müller, J.P. (eds.) Agent Oriented Software Engineering V. Lecture Notes in Computer Science, vol. 3382, pp. 62–77. Springer, New York (2004)
2. Bresciani, P., Giorgini, P., Giunchiglia, F., Mylopoulos, J., Perini, A.: Tropos: an agent-oriented software development methodology. Auton. Agent Multi Agent Syst. **8**(3), 203–236 (2004). http://www.springerlink.com/content/g757056736223u65/

3. Brinkkemper, S.: Method engineering: engineering of information systems development methods and tools. Inf. Softw. Technol. **38**(4), 275–280 (1996)
4. Brinkkemper, S., Saeki, M., Harmsen, F.: Meta-modelling based assembly techniques for situational method engineering. Inf. Syst. **24**(3), 209–228 (1999)
5. Cernuzzi, L., Cossentino, M., Zambonelli, F.: Process models for agent-based development. Eng. Appl. Artif. Intell. **18**(2), 205–222 (2005)
6. Cossentino, M.: From requirements to code with the PASSI methodology. In: Henderson-Sellers, B., Giorgini, P. (eds.): Agent Oriented Methodologies, Chap. IV, pp. 79–106. Idea Group Publishing, Hershey (2005). http://www.idea-group.com/books/details.asp?id=4931
7. Cossentino, M., Gaglio, S., Sabatucci, L., Seidita, V.: The passi and agile passi mas meta-models compared with a unifying proposal. In: Proceedings of the CEEMAS'05 Conference, Budapest, Hungary, pp. 183–192 (2005)
8. Cossentino, M., Gaglio, S., Galland, S., Gaud, N., Hilaire, V., Koukam, A., Seidita, V.: A MAS metamodel-driven approach to process fragments selection. In: Luck, M., Gómez-Sanz, J.J. (eds.) Agent-Oriented Software Engineering IX. Lecture Notes in Computer Science, vol. 5386, pp. 86–100. Springer, Berlin (2009)
9. Cossentino, M., Hilaire, V., Molesini, A.: Fipa design process documentation and fragmentation working group. http://www.pa.icar.cnr.it/cossentino/fipa-dpdf-wg/ (2010)
10. Fuggetta, A.: Software process: a roadmap. In: ICSE '00: Proceedings of the Conference on The Future of Software Engineering, pp. 25–34. ACM Press, New York (2000). http://doi.acm.org/10.1145/336512.336521
11. Garijo, F.J., Gòmez-Sanz, J.J., Massonet, P.: The MESSAGE methodology for agent-oriented analysis and design. In: Henderson-Sellers, B., Giorgini, P. (eds.): Agent Oriented Methodologies, Chap. 8, pp. 203–235. Idea Group Publishing, Hershey (2005). http://www.idea-group.com/books/details.asp?id=4931
12. Ghezzi, C., Jazayeri, M., Mandrioli, D.: Fundamental of Software Engineering, 2nd edn. Prentice Hall, Englewood Cliffs (2002)
13. IEEE-FIPA: Design Process Documentation Template. `http://fipa.org/specs/fipa00097/SC00097B.pdf` (2012)
14. Pavòn, J., Gòmez-Sanz, J.J., Fuentes, R.: The INGENIAS methodology and tools. In: Henderson-Sellers, B., Giorgini, P. (eds.): Agent Oriented Methodologies, Chap. IX, pp. 236–276. Idea Group Publishing, Hershey (2005). http://www.idea-group.com/books/details.asp?id=4931
15. Picard, G., Bernon, C., Gleizes, M.P.: Cooperative agent model within ADELFE framework: an application to a timetabling problem. In: Jennings, N.R., Sierra, C., Sonenberg, L., Tambe, M. (eds.) Autonomous Agents and Multi Agent Systems, vol. 3, pp. 1506–1507. ACM Press, New York (2004). http://doi.ieeecomputersociety.org/10.1109/AAMAS.2004.10200
16. SODA: Home page. `http://soda.apice.unibo.it`. http://soda.alice.unibo.it (2012)
17. Sommerville, I.: Software Engineering, 8th edn. Addison-Wesley, Reading (2007)
18. Wood, M.F., DeLoach, S.A.: An overview of the multiagent systems engineering methodology. In: Ciancarini, P., Wooldridge, M.J. (eds.) Agent-Oriented Software Engineering. Lecture Notes in Computer Science, vol. 1957, pp. 207–221. Springer, Berlin (2001). http://www.springerlink.com/link.asp?id=0pw8rqj5kpbdnflx. 1st International Workshop (AOSE 2000), Limerick, Ireland, 10 June 2000. Revised Papers
19. Zambonelli, F., Omicini, A.: Challenges and research directions in agent-oriented software engineering. Auton. Agent Multi Agent Syst. **9**(3), 253–283 (2004). doi:10.1023/B:AGNT.0000038028.66672.1e. http://journals.kluweronline.com/article.asp?PIPS=5276775. Special Issue: Challenges for Agent-Based Computing
20. Zambonelli, F., Jennings, N., Wooldridge, M.: Multiagent systems as computational organizations: the Gaia methodology. In: Henderson-Sellers, B., Giorgini, P. (eds.): Agent Oriented Methodologies, Chap. 6, pp. 136–171. Idea Group Publishing, Hershey (2005). http://www.idea-group.com/books/details.asp?id=4931