

TALLINN UNIVERSITY OF TECHNOLOGY
DOCTORAL THESIS
9/2019

Simulation and Prototyping of Sociotechnical Systems Using Agent-Oriented Modelling

MSURY ROGASIAN MAHUNNAH

TALLINN UNIVERSITY OF TECHNOLOGY

School of Information Technologies

Department of Software Science

This dissertation was accepted for the defence of the degree 20/12/2018

Supervisor:

Kuldar Taveter, Ph.D., Senior Researcher
Department of Software Science
Tallinn University of Technology
Tallinn, Estonia

Co-supervisor:

Alexander Horst Norta, Ph.D., Associate Professor
Department of Software Science
Tallinn University of Technology
Tallinn, Estonia

Opponents:

Olegas Vasilecas, Ph.D., Professor
Department of Information Systems
Vilnius Gediminas Technical University
Vilnius, Lithuania

Ghassan Beydoun, Ph.D., Professor
School of Information, Systems and Modelling
University of Technology Sydney
Australia

Defence of the thesis: 22/02/2019, Tallinn

Declaration:

Hereby I declare that this doctoral thesis, my original investigation and achievement, submitted for the doctoral degree at Tallinn University of Technology has not been submitted for doctoral or equivalent academic degree.

MSURY ROGASIAN MAHUNNAH

signature



European Union
European Social Fund



Investing in your future

Copyright: Msury Rogasian Mahunnah, 2019

ISSN 2585-6898 (publication)

ISBN 978-9949-83-394-8 (publication)

ISSN 2585-6901 (PDF)

ISBN 978-9949-83-395-5 (PDF)

TALLINNA TEHNIKAÜLIKOOL
DOKTORITÖÖ
9/2019

**Sotsiotehniliste süsteemide simulatsioon
ja prototüüpimine kasutades
agentorienteeritud modelleerimist**

MSURY ROGASIAN MAHUNNAH

Contents

| | |
|--|----|
| List of Publications..... | 8 |
| Other Related Publications..... | 9 |
| Authors Contribution to the Publications | 10 |
| Abbreviations | 11 |
| 1 Introduction..... | 12 |
| 1.1 Sociotechnical Systems..... | 12 |
| 1.2 Research Motivation | 12 |
| 1.3 Research Objectives | 14 |
| 1.4 Research Questions | 14 |
| 1.5 Research Methodology..... | 15 |
| 1.6 Structure of the Thesis | 17 |
| 1.7 Contributions of the Thesis | 18 |
| 2 Background and Related Work..... | 20 |
| 2.1 Agent-Oriented Modelling..... | 20 |
| 2.1.1 Evolution of Agent Concept..... | 20 |
| 2.1.2 Overview of AOSE Methodologies..... | 21 |
| 2.1.3 Requirements Modelling for Sociotechnical Systems | 22 |
| 2.1.4 Design Modelling of Sociotechnical Systems..... | 23 |
| 2.1.5 Applications of AOM4STS methodology..... | 24 |
| 2.2 Coloured Petri Nets Formalism | 26 |
| 2.2.1 Basic Concepts of Petri Nets..... | 26 |
| 2.2.2 Coloured Petri Nets: The Language | 27 |
| 2.2.3 Analysing Coloured Petri Net Models..... | 28 |
| 2.3 Java Agent Development Environments Framework | 29 |
| 2.3.1 FIPA Specifications..... | 29 |
| 2.3.2 JADE Run-Time System Architecture | 30 |
| 2.4 Agent-Oriented Tools for Modelling Sociotechnical Systems | 31 |
| 2.4.1 Tools for Agent-Oriented Modelling..... | 31 |
| 2.4.2 Tools for Sociotechnical Systems..... | 31 |
| 2.4.3 Tools for Model Driven Engineering..... | 33 |
| 2.5 Summary..... | 34 |
| 3 Simulation by Coloured Petri Nets Tools..... | 35 |
| 3.1 Introduction..... | 35 |
| 3.2 Guidelines for Representing Agent-Oriented Models in CPN Tools | 35 |
| 3.2.1 Guidelines for transformation of knowledge models..... | 36 |
| 3.2.2 Guidelines for transformation of interaction models..... | 38 |
| 3.2.3 Guidelines for transformation of agent behaviour models..... | 42 |
| 3.3 Intruder Handling Case Study | 52 |
| 3.3.1 Description of the intruder handling scenario..... | 52 |
| 3.3.2 Conceptual models of the intruder handling system | 53 |
| 3.3.3 Simulation of the intruder handling system | 57 |
| 3.4 Empirical Evaluation of CPN Modelling Guidelines | 69 |
| 3.4.1 Experiment Scoping..... | 69 |

| | |
|---|-----|
| 3.4.2 Experiment Planning..... | 70 |
| 3.4.3 Experiment Operation | 73 |
| 3.4.4 Experiment Results and Interpretation | 74 |
| 3.5 Summary..... | 80 |
| 4 Prototyping by JADE Framework | 82 |
| 4.1 Introduction..... | 82 |
| 4.2 Extended Intruder Handling Case Study..... | 83 |
| 4.3 Guidelines for Prototyping of Agent-Oriented Models on the JADE Framework..... | 85 |
| 4.3.1 Conceptual models of the extended intruder handling system | 85 |
| 4.3.2 Prototyping of the extended intruder handling system | 91 |
| 4.4 Empirical Evaluation of the Guidelines for Prototyping of Agent-Oriented Models on the JADE Framework | 97 |
| 4.4.1 Experiment Scoping..... | 97 |
| 4.4.2 Experiment Planning..... | 98 |
| 4.4.3 Experiment Operation | 100 |
| 4.4.4 Experiment Results and Interpretation | 101 |
| 4.5 Summary..... | 103 |
| 5 Tool Support for Requirements and Design Modelling | 105 |
| 5.1 Introduction..... | 105 |
| 5.2 The viewpoint framework | 105 |
| 5.3 Functionality of the AOM4STS Tool..... | 106 |
| 5.3.1 Saving and Loading | 106 |
| 5.3.2 Online Diagramming..... | 107 |
| 5.3.3 Graphical User Interface..... | 108 |
| 5.3.4 Information propagation | 108 |
| 5.3.5 Consistency checking..... | 109 |
| 5.4 Summary..... | 110 |
| 6 Empirical Evaluation of AOM4STS Tool | 111 |
| 6.1 Introduction..... | 111 |
| 6.2 Experiment Planning..... | 111 |
| 6.2.1 Goal of the study | 111 |
| 6.2.2 Context selection | 111 |
| 6.2.3 Objects of study..... | 112 |
| 6.2.4 Subjects | 112 |
| 6.2.5 Experiment design | 112 |
| 6.3 Modelling Experiment | 113 |
| 6.3.1 Aspects for Research Questions | 113 |
| 6.3.2 Variables and measures..... | 114 |
| 6.3.3 Experiment procedure and materials..... | 114 |
| 6.4 Data Analysis | 116 |
| 6.5 Results and Interpretation..... | 117 |
| 6.5.1 Adequateness of the experimental settings..... | 117 |
| 6.5.2 Main factor: results and interpretation..... | 120 |
| 6.5.3 Additional results..... | 123 |
| 6.5.4 Threats to validity | 126 |
| 6.6 Summary..... | 127 |

| | |
|--|-----|
| 7 Conclusions, Limitations and Future Work..... | 129 |
| 7.1 Research Summary and Conclusions..... | 129 |
| 7.1.1 Guidelines for Modelling of Agent-Oriented Models in CPN Tools..... | 129 |
| 7.1.2 Guidelines for Prototyping of Agent-Oriented Models on the JADE Framework..... | 130 |
| 7.1.3 Support by the AOM4STS Tool for Modelling Sociotechnical Systems..... | 130 |
| 7.2 Limitations of the Research..... | 131 |
| 7.3 Future Work..... | 132 |
| List of Figures..... | 134 |
| List of Tables..... | 136 |
| References..... | 137 |
| Acknowledgements..... | 146 |
| Abstract..... | 147 |
| Kokkuvõte..... | 148 |
| Appendix A..... | 149 |
| Appendix B..... | 157 |
| Appendix C..... | 179 |
| Appendix D..... | 189 |
| Curriculum Vitae..... | 197 |
| Elulookirjeldus..... | 198 |

List of Publications

The following are articles published by the author during the doctoral study:

- I **Msurry Mahunnah**, Alexander Norta, Lixin Ma, and Kuldar Taveter. "Heuristics for Designing and Evaluating Sociotechnical Agent-Oriented Behaviour Models with Coloured Petri Nets." In Computer Software and Applications Conference Workshops (COMPSACW), 2014 IEEE 38th International, pp. 438–443. IEEE, 2014.
- II Cheah Wai Shiang, Bong Tien Onn, Fu Swee Tee, Muhammad Asyraf bin Khairuddin, and **Msurry Mahunnah**. "Developing Agent-Oriented Video Surveillance System through Agent-Oriented Methodology (AOM)." CIT. Journal of Computing and Information Technology 24, no. 4: 349–368. SCOPUS, 2016.
- III **Msurry Mahunnah**, Kuldar Taveter, and Raimundas Matulevicius, "An Empirical Evaluation of the Requirements Engineering Tool for Sociotechnical Systems.," in 26th IEEE International Requirements Engineering Conference Workshops (REW). IEEE, 2018.
- IV **Msurry Mahunnah**, Kuldar Taveter, Cheah Wai Shiang, and Sim Wai Yee, "An Empirical Evaluation of Guidelines for Prototyping Sociotechnical Systems in JADE Framework.," in 3rd IEEE International Symposium on Agents, Multi-Agent Systems and Robotics. IEEE, 2018.

Other Related Publications

The following are additional articles published by the author during the doctoral study:

- V **Mahunnah, Msury**, Annike Koorts, and Kuldar Taveter. "Towards Distributed Sociotechnical System for Reporting Critical Laboratory Results." In HEALTHINF, pp. 269–276. SCOPUS, 2013. *3.1 category*.
- VI **Mahunnah, Msury**, and Kuldar Taveter. "A scalable multi-agent architecture in environments with limited connectivity: Case study on individualised care for healthy pregnancy." In Digital Ecosystems and Technologies (DEST), 2013 7th IEEE International Conference on, pp. 83–89. IEEE, 2013. *3.1 category*.
- VII Alexander Norta, **Msury Mahunnah**, Tanel Tenso, Kuldar Taveter, and Nanjangud C. Narendra. "An agent-oriented method for designing large sociotechnical service-ecosystems." In Services (SERVICES), 2014 IEEE World Congress on, pp. 242–249. IEEE, 2014. *3.1 category*.
- VIII Narendra, Nanjangud C., Alexander Norta, **Msury Mahunnah**, and Fabrizio Maggi. "Modelling sound conflict management for virtual-enterprise collaboration." In Services Computing (SCC), 2014 IEEE International Conference on, pp. 813–820. IEEE, 2014. *3.1 category*.
- IX Narendra, Nanjangud C., Alexander Norta, **Msury Mahunnah**, Lixin Ma, and Fabrizio Maria Maggi. "Sound conflict management and resolution for virtual-enterprise collaborations." Service Oriented Computing and Applications 10, no. 3: 233–251. Springer, 2016. *1.1 category*.

Authors Contribution to the Publications

Contribution by the author in the published articles:

- I The author's contribution is the entire paper, except the description of the related work. This paper proposes guidelines that support representing design models produced by the AOM4STS methodology by CPN Tools for visualisation, validation and verification.
- II The author's contribution are the guidelines that support prototyping sociotechnical systems on the JADE framework based on design models produced by the AOM4STS methodology. This paper adapts the proposed JADE prototyping guidelines in the development of agent-oriented video surveillance system in JADE framework.
- III The author's contribution is the entire paper. This paper analyses the results of an empirical study that investigates the effectiveness of adapting guidelines for prototyping conceptual models of sociotechnical systems in Java Agent Development (JADE) framework.
- IV The author's contribution is the entire paper. This paper provides the overview of a novel software modelling prototype (AOM4STS) and analyses the results of empirical study that compares the modelling effort and effectiveness of the novel software tool for modelling requirements of sociotechnical systems against modelling on paper.

Abbreviations

| | |
|---------|--|
| ABS | Agent-Based Simulation |
| AOM | Agent Oriented Modelling |
| AOSE | Agent Oriented Software Engineering |
| AUML | Agent Unified Modelling Language |
| CASE | Computer Aided Software Engineering |
| CCTV | Closed-Circuit Television |
| CIM | Computational Independent Models |
| CLR | Critical Laboratory Results |
| CPN | Coloured Petri-Nets |
| DES | Discrete-Event Simulation |
| DSR | Design Science Research |
| ER | Early Requirements |
| IS | Information Systems |
| IT | Information Technology |
| JADE | Java Agent DEvelopment |
| LT | Late Requirements |
| MDA | Model Driven Architecture |
| MoP | Modelling on Paper |
| MoPS | Modelling on Software |
| MPM | Multi-Paradigm Modelling |
| MSC | Message Sequence Chart |
| NEMC | North Estonian Medical Centre |
| O-MaSE | Organization-based Multi-agent System Engineering |
| PASSI | Process for Agent Societies Specification and Implementation |
| PIM | Platform-independent Models |
| PSM | Platform-specific Models |
| RAP/AOR | Radical Agent-Oriented Process/Agent-Object-Relationship |
| SDLC | Software Development Life Cycle |
| SSA | State Space Analysis |
| UCD | User-Centred Design |
| UML | Unified Modelling Language |
| XML | Extensible Mark-up Language |

1 Introduction

1.1 Sociotechnical Systems

The developments in the software engineering field have led to a renewed interest in system development methodologies that cover higher contexts and leads to the design of systems that are more acceptable to end users and deliver better value for stakeholders [8]. The meta-context comprises community and personal requirements that complement physical and informational requirements of systems. Maguire [84] identifies such systems as sociotechnical systems. A sociotechnical system is a software-intensive system that has defined operational processes followed by human operators that operates within an organization [81]. Such a system contains both a social aspect and a technical aspect. These aspects can be expanded into different levels and perspectives. For example, according to Whitworth [152], a sociotechnical system involves four levels: physical, information, personal, and group level. da Conceição, *et al.* [31] distinguish between even seven abstraction levels of sociotechnical systems in the maritime domain: natural environment, and reactive, automated reactive, proactive, planning scheduling, planning strategic, and political-economic levels.

In addition to the abstraction levels, Davis *et al.* [37] propose six perspectives of sociotechnical systems, which are orthogonal to the abstraction levels: goals, people, technologies, physical infrastructure, cultural assumptions, and processes and working practices. Marsilio *et al.* [92] put forward the same number, while somewhat different perspectives for sociotechnical systems in the healthcare domain: devices and tools, layout and organization of space, core process standardization, organizational structure, human resource management, and operations management.

This thesis follows [7,40] and identifies the following five characteristics of a sociotechnical system. First, common main objective – the success of the whole system relies on the ways individual parties forming the system fulfil their objectives. Second, interdependent parties – the individual parties of the system collaborate by exchanging knowledge items through interactions. Third, social and technical sub-systems – the quality of the whole system depends on the joint optimisation of the technical and social sub-systems, i.e., focusing on one of these systems to the exclusion of the other is likely to degrade the overall system's quality. Fourth, open system – the system needs to be aware of and adaptive to the changes in the environmental conditions, including the existence of new knowledge and requirements. And the last, equifinality – the goals of the system can be achieved by many different ways or sets of activities. This motivates the need to make design choices during system development process.

1.2 Research Motivation

If a computer game does not feel fun, we will not play it; if an ecommerce website does not feel trustworthy (irrespective of the actual security) we will not purchase from it; and if a social networking application does not feel engaging we will not use it [91]. Therefore, consideration of human factors becomes crucial during the development life cycle of sociotechnical systems. For sociotechnical systems to evolve and extend their scope, the source [37] suggests (i) to extend conceptualization of what constitutes a system; (ii) apply our thinking to a much wider range of complex problems and global challenges; and (iii) engage in more predictive work. The sources [21,33,135] consider

human, software and devices as necessary building blocks of sociotechnical systems. However, [21] focuses on improving workplace safety from a sociotechnical perspective, [33] focuses on engineering security requirements for sociotechnical systems and [135] suggests a holistic modelling approach that focuses on requirements elicitation [2] and design [137] of sociotechnical systems using agent-oriented modelling. This thesis refers to the modelling approach suggested by Sterling and Taveter [135] as the Agent-Oriented Modelling for a Sociotechnical System (AOM4STS) methodology. The major difference between the AOM4STS methodology and related methodologies depends on their application domains. According to [135], the AOM4STS methodology has a wide application domain (i.e., it is domain independent) while related methodologies [21,33] have narrow application domain (i.e., they are domain dependent).

Generally, the AOM4STS methodology [135] consists of two phases: requirements-engineering phase [2] and system-design phase [137]. The requirements engineering phase involves the modelling of goals, roles, relationships between them – organisation – and domain knowledge with the aim to capture the sociotechnical requirements of the system. This is followed by deciding the agent types, identifying the knowledge possessed by agents, formulating interactions between agents and determining the behaviours of agents during the design phase.

However, this thesis identifies the following gaps in the AOM4STS methodology that should be further researched. First, the AOM4STS methodology does not provide a mechanism to support visualisation, and validation and verification of design models of sociotechnical systems before the development phase. Second, the AOM4STS methodology does not provide a software support for prototyping sociotechnical systems. Third, the AOM4STS methodology does not have a software tool to support its modelling process.

According to [124], simulation models provide adequate capabilities for validation, verification and prediction of the behaviour of the system being designed. Specifically, the capabilities of agent-based simulation (ABS) provide a useful technique for simulating sociotechnical systems that are distributed and involve complex interactions [36]. These capabilities include structure-preserving modelling of the simulated reality, simulation of proactive behaviour, parallel computations, and dynamic simulation scenarios. The development of ABS addresses inadequacy of Discrete-Event Simulation (DES) tools to address modelling of populations of diverse individuals having a variety of behaviours and interactions [131]. However, the development of Coloured Petri-Nets (CPN) Tools advances DES by providing the support for modelling and simulation of populations of diverse individuals having a variety of behaviours, interactions and information exchange [69]. Therefore, this thesis employs the CPN Tools¹ to support visualisation, validation and verification of conceptual design models of sociotechnical systems through simulation.

Moreover, the current literature [79] provides a comprehensive review of all available agent platforms that are or can be used for simulating and implementing real-life case studies. This study analyses twenty four agent platforms developed by different academic or industry-oriented groups. The results of this study show that Java Agent Development (JADE) framework² is the most popular agent platform, which has

¹ <http://cpntools.org/>

² <http://jade.tilab.com/>

been purely designed in Java and supports development of different kinds of systems. Thus, this thesis employs JADE framework for prototyping sociotechnical systems based on design models that have been produced by the AOM4STS methodology.

1.3 Research Objectives

The following research objectives are addressed in this thesis to fill the research gap that has been identified in Section 1.2:

- To propose guidelines that support representing design models produced by the AOM4STS methodology in CPN Tools for visualisation, validation and verification.
- To evaluate the effectiveness of applying the proposed guidelines to support representing design models produced by the AOM4STS methodology in CPN Tools.
- To evaluate the utility of CPN Tools in visualisation, validation and verification of design models produced by the AOM4STS methodology.
- To propose guidelines that support the prototyping of sociotechnical systems on the JADE framework based on design models produced by the AOM4STS methodology.
- To evaluate the effectiveness of applying the proposed guidelines for prototyping sociotechnical systems on the JADE framework based on design models produced by the AOM4STS methodology.
- To develop a software modelling tool that supports information propagation and consistency checking during the requirements and design modelling of sociotechnical systems using the AOM4STS methodology.
- To evaluate modelling effort and effectiveness of the developed software modelling tool in supporting the requirements and design modelling of sociotechnical systems using the AOM4STS methodology.

1.4 Research Questions

This thesis fulfils the research objectives identified in Section 1.3 by answering the overall research question:

How to support simulation and prototyping of distributed sociotechnical systems employing AOM4STS methodology?

The research questions below are aligned with the overall research question. To establish complexity-reducing separation of concerns, sub-research questions are deduced from each research question as follows:

RQ1. How to provide the support for visualisation, validation and verification of design models of sociotechnical systems produced by the AOM4STS methodology?

- (a) What guidelines support the modelling of design models of sociotechnical systems produced by the AOM4STS methodology in CPN Tools?
- (b) What is the effectiveness of the guidelines proposed in (a) for representing design models of sociotechnical systems produced by the AOM4STS methodology in CPN Tools?
- (c) What is the utility of CPN Tools in visualisation, validation and verification of design models produced by the AOM4STS methodology?

RQ2. How to provide the support for prototyping sociotechnical systems based on design models produced by the AOM4STS methodology?

- (a) What guidelines support the prototyping of sociotechnical systems on the JADE framework based on design models produced by the AOM4STS methodology?
- (b) What is the effectiveness of the guidelines proposed in (a) for prototyping sociotechnical systems on the JADE framework based on design models produced by the AOM4STS methodology?

RQ3. How to support the requirements and design modelling of sociotechnical systems with a software tool supporting the AOM4STS methodology?

- (a) What key features of a software tool support requirements engineering and design of sociotechnical systems using the AOM4STS methodology?
- (b) What is the impact of the software tool on the effort needed for requirements and design modelling of sociotechnical systems by means of the AOM4STS methodology?
- (c) What is the effectiveness of the software tool for requirements and design modelling of sociotechnical systems employing AOM4STS methodology?

1.5 Research Methodology

This thesis follows the Design-Science Research (DSR) methodology that is commonly applied in the research domain of information systems and software engineering [4]. The DSR seeks to extend the boundaries of human and organizational capabilities by creating new and innovative artefacts that are broadly defined as constructs (vocabulary and symbols), models (abstractions and representations), methods (algorithms and practices), and instantiations (implemented and prototype systems).

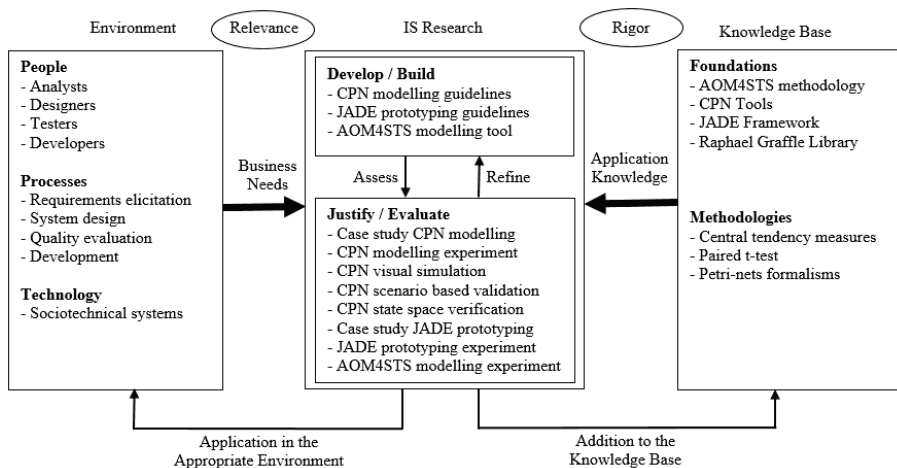


Figure 1-1: A framework for design-science research in information systems adapted from [4].

The depiction in Figure 1-1 gives an overview of the DSR framework that is associated with three pillars: the environment pillar on the left, the knowledge base pillar on the right, and the information systems (IS) research pillar in the middle. The environment pillar defines the problem domain in which the focus of interest consists of people, organizations, and technology. DSR achieves relevance by building artefacts that address the business needs evolving from the environment. The knowledge base pillar

determines the existing foundations and methodologies from and through which IS research is accomplished. The IS research pillar achieves rigor by applying foundations in the develop/build phase and applying measures during the justify/evaluate phase. The results of DSR are assessed by the accomplishments of business needs in the appropriate environment and by the contributions to the content of the knowledge base that establishes the foundations for applications and further research.

Figure 1-1 is populated by the notions of the environment, IS research, and knowledge base pillars that are relevant for applying DSR in this thesis. The major artefacts suggested by the thesis are the CPN guidelines proposed in Chapter 3, the JADE guidelines put forward in Chapter 4 and the AOM4STS tool support proposed in Chapter 5. The CPN guidelines provide the support for visualisation, validation and verification of design models of sociotechnical systems produced by the AOM4STS methodology. The JADE guidelines provide support for the prototyping sociotechnical systems based on design models produced by the AOM4STS methodology. Furthermore, the AOM4STS tool provides support for the requirements and design modelling of sociotechnical systems employing the AOM4STS methodology.

The rigor of these research artefacts is ensured by retrieving application knowledge from various sources. All three artefacts aim to extend the AOM4STS methodology [2,135,137] for the requirements engineering and design of sociotechnical systems. The rigor of the CPN guidelines is ensured by CPN Tools¹ and the CPN modelling language [69]. The rigor of the JADE guidelines is ensured by the JADE framework², which is an established agent platform, purely designed in Java, and supports developing different kinds of systems operating on the web [79]. Furthermore, the rigor of the AOM4STS tool support is ensured by the Raphael Graffle Library³, which is a JavaScript vector library to support the creation of web-based graphs.

Moreover, the utility, quality, and efficacy of design artefacts must be rigorously demonstrated via well-executed evaluation methods [146]. Therefore, the rigor of the research evaluation in this thesis is ensured by three major methodologies – measures of central tendency [6], paired t-test [60] and the Petri Nets formalism [116]. First, the measures of central tendency include mean, median and mode. Second, the paired t-test is a statistical analysis test that compares mean differences between treatments when the observations have been obtained in pairs [60]. Third, the Petri Nets formalism is a mathematical (formal) modelling language that describes distributed systems by employing methods of discrete event dynamics [116].

Furthermore, this thesis uses an intruder handling case study [134] to demonstrate the feasibility and applicability of the CPN guidelines in Chapter 3, of the JADE guidelines in Chapter 4 and of the AOM4STS tool support in Chapter 5. Moreover, this thesis reports three empirical studies. First, Chapter 3 presents an empirical study on modelling experiments that measures the effectiveness of using the CPN guidelines. Moreover, Chapter 3 analyses simulation results by CPN Tools that include visualisation, scenario-based validation and state space verification of design models of sociotechnical systems produced by the AOM4STS methodology. Second, Chapter 4 presents an empirical study on prototyping experiments that measures the effectiveness of using

¹ <http://cpntools.org/>

² <http://jade.tilab.com/>

³ <https://dmitrybaranovskiy.github.io/raphael/graffle.html>

the JADE guidelines. And third, Chapter 6 presents an empirical study on modelling experiments that measure the effort and effectiveness of using the AOM4STS tool support.

The results of this research work are relevant for analysts, designers, testers and developers from academic-, research and industrial organisations, who focus on requirements engineering, design, quality evaluation, and the development of sociotechnical systems. On the one hand, the results of this research work aim to support the simulation of design models of sociotechnical systems prior to the actual development phase. Therefore, we reduce the risk of failure by evaluating a number of possible scenario outcomes, providing performance measures, and stimulating creativity by allowing many different alternative design decisions to be tested quickly and cheaply [54]. On the other hand, the results of this research work aim to support prototyping sociotechnical systems based on their design models. Therefore, we enhance the understanding of the customer requirements at an early stage through the feedback received from the customer who assists the designers and developers to understand what exactly is expected from the software that is under development [15].

1.6 Structure of the Thesis

This Section presents the structure of the thesis that provides answers to the specific research questions and sub-research questions outlined in Section 1.4 in a step-by-step manner as follows.

Chapter 2 bridges the gap between the research questions and research objectives identified in Chapter 1 and the rest of the thesis by providing overviews of the AOM4STS methodology, CPN Tools, and JADE framework.

Chapter 3 proposes the CPN guidelines that support representing design models produced by the AOM4STS methodology in CPN Tools for visualisation, validation and verification. Furthermore, this Chapter demonstrates the feasibility and applicability of CPN Tools by applying the proposed CPN guidelines to modelling and simulating the intruder handling case study in CPN Tools using CPN guidelines and also reports simulation results by CPN Tools that evaluate visualisation, validation and verification of design models of sociotechnical systems produced by the AOM4STS methodology.

Chapter 4 proposes the JADE guidelines that support prototyping sociotechnical systems on the JADE framework based on design models produced by the AOM4STS methodology. Furthermore, this Chapter demonstrates the feasibility and applicability of the JADE guidelines by applying the proposed JADE guidelines to prototyping the intruder handling case study on the JADE framework. Lastly, this Chapter reports an empirical study about prototyping experiments on the JADE framework using the JADE guidelines.

Chapter 5 describes a novel online diagramming software tool – AOM4STS – that supports information propagation and consistency checking during requirements and design modelling of sociotechnical systems using the AOM4STS methodology.

Chapter 6 analysis and discusses the results of an empirical study that evaluates the modelling effort and modelling effectiveness of using the novel software tool for requirements and design modelling of sociotechnical systems by means of the AOM4STS methodology.

Lastly, Chapter 7 presents the conclusions of this thesis, explains research limitations, and outlines the directions for future work.

1.7 Contributions of the Thesis

The following are the major contributions of this thesis.

First, this thesis proposes novel guidelines that support the development of design models produced by the AOM4STS methodology in CPN Tools for visualisation, validation and verification. These guidelines are divided into the guidelines for CPN knowledge modelling, CPN interaction modelling and CPN behaviour modelling. This contribution is reflected by **Paper I** [86].

Second, this thesis analyses and interprets the results of an empirical study that evaluates the effectiveness of adapting the proposed guidelines to representing by CPN Tools design models produced by the AOM4STS methodology. The results of this empirical study conclude that the proposed CPN modelling guidelines more effectively support modelling of agents and their knowledge in CPN Tools compared to modelling agent behaviours and interactions.

Third, this thesis analyses and interprets the results of an empirical study that evaluates the utility of CPN Tools in visualisation, validation and verification of design models produced by the AOM4STS methodology. In this empirical study, the results of visual simulations demonstrate the capabilities of Message Sequence Charts (MSC) of CPN Tools in supporting visualisation of activities and interactions between various agents of different applications of sociotechnical systems. In addition, the scenario-based validation of the resulting CPN models of sociotechnical systems affirms that business rules are central building blocks in scenario-based validation of models of sociotechnical systems. Finally, the results of the state space verification demonstrate how the formalisms in CPN Tools support the quality improvement of conceptual design models by identifying unwanted states and activities of sociotechnical systems.

Fourth, this thesis proposes novel guidelines that support prototyping sociotechnical systems on the JADE framework based on design models produced by the AOM4STS methodology. These guidelines are divided into guidelines for JADE-specific interaction prototyping, behaviour prototyping, and knowledge prototyping. The latter involves the implementations of conceptual objects and ontologies for sociotechnical systems on the JADE framework. This contribution is reflected by **Paper II** [128].

Fifth, this thesis analyses and interprets the results of an empirical study that evaluates the effectiveness of applying the proposed guidelines to prototyping sociotechnical systems on the JADE framework based on design models produced by the AOM4STS methodology. The results of this empirical study conclude that the JADE guidelines together with the JADE website resources are more effective for the development and implementation of knowledge by agents belonging to sociotechnical systems on the JADE framework compared to using only the JADE website resources for the same purpose. Additionally, these empirical results affirm the research findings by [41] that conceptual objects are necessary building blocks for the development of ontologies. This contribution is reflected by **Paper IV** [89].

Sixth, this thesis describes the key features of a novel software modelling tool that supports information propagation and consistency checking during the requirements and design modelling of sociotechnical systems using the AOM4STS methodology. This tool aims to reduce the modelling effort and to improve the modelling effectiveness during the requirements elicitation and design of sociotechnical systems. This contribution is reflected by **Paper III** [88].

Lastly, this thesis analyses and interprets the results of an empirical study that evaluates modelling effort and effectiveness of the developed software modelling tool (AOM4STS tool) in supporting the requirements and design modelling of sociotechnical systems using the AOM4STS methodology. The study involved experimental tasks of modelling requirements for sociotechnical systems. The subjects created requirements models of two case studies, one of which was modelled on paper and another one – with the AOM4STS tool. The results show that the modelling effort on paper is nearly the same as the modelling effort with the AOM4STS software tool. Moreover, the effectiveness of modelling requirements with the modelling tool was higher than the effectiveness of modelling requirements on paper by considering information propagation, consistency checking, and visual cognition. However, the results show that goal decomposition activity is slightly more effective when modelling on paper compared to modelling with the tool. This contribution is reflected by **Paper III** [88].

2 Background and Related Work

2.1 Agent-Oriented Modelling

This Section aims to provide a deeper understanding of the notion of Agent-Oriented Software Engineering (AOSE) and its methodologies. Subsection 2.2.1 presents an overview of well-known AOSE methodologies that guide the development lifecycle of agent-oriented systems. Some of these methodologies form the foundation of the AOM4STS methodology [2,135,137]. An important difference between AOM4STS and other AOSE methodologies is that AOM4STS explicitly focuses on the development lifecycle of sociotechnical system, while other AOSE methodologies address the development lifecycle of agent-oriented systems consisting of software agents and robots, both of which can be subsumed under the term “man-made agents” [139].

2.1.1 Evolution of Agent Concept

The term agent is derived from the Latin word “agere” which means, “to do”, the agreement to act on one’s behalf. The same situation happens in computer science when a system user enters into agreement and delegates some of his/her own authority or responsibilities to another user, software or hardware component [103,135]. For example, a person may authorise another person, who is a dietician, to analyse his or her lifestyle and recommend him/her a healthy diet. Also, a person or a given software component may delegate his/her/its own responsibilities to another software or hardware component in order to automate complex and repetitive activities [125]. For example, to analyse various conditions of a patient, such as gender, age, height, weight, blood pressure, blood glucose, and so on, and to identify the most similar other patients from a database.

The use of the agent concept can be traced back to the research work published in 1977 about computations that involves communicating parallel processes [59]. The project used software components termed as actors created during computational processes. These actors interacted through message exchange and ran continuously to accomplish specific tasks assigned to them [3]. In the beginning of 1990s, researchers widely started to explore the potential use of agent technology in diverse research fields such as industrial applications, commercial applications, medical applications and entertainment [68]. Industrial applications included projects on process control [64], manufacturing [109], and air traffic control [76], while commercial applications included projects on information management [83], electronic processes [25], and business process management [67]. Some of the projects that focused on medical applications included patient monitoring [58], and healthcare systems for decision support [61], while the projects from entertainment industry included computer games [149] and interactive theatre and movies [57].

As briefly explained in the previous paragraph, the existence of literature on agent technology and agent-oriented systems from various research communities resulted to a range of definitions of the term “agent”. Widely known definitions define agent as anything that can perceive its environment through sensors and act upon that environment through actuators, and reason between perceiving and acting [118], [82], [56]. According to this definition, if the notations of environment, sensors and actuators aren’t restricted, a software application running in a mobile phone can be an agent [122,142].

The definition put forward in [103], defines an agent as a software or hardware component that can act on behalf of its user to accomplish tasks. This definition restricts an agent to a computational entity embedded in software or hardware that executes the responsibilities delegated to it by the user. This kind of agent is commonly referred as Personal Assistant (PA) [72,99]. For example in the development of Intelligent PA for task and time management [99], the PA executes routine tasks such as suggesting convenient meeting time, thus allowing the user to focus on tasks that critically require human problem solving skills such as confirming the best choice of meeting time.

The definition in [45] complements the definitions of agent overviewed in the previous paragraph by the notion of autonomy. This definition states that, an autonomous agent is a system situated within and a part of an environment that senses that environment and acts on it, over time, in pursuit of its own agenda. Consequently, becomes capable to effect what it senses in the future. The notion of autonomy of an agent refers to the agent's capability to make its own decisions about the actions to be performed by it in order to fulfil its intended set of goals [82], [56].

In the process of trying to unify the definitions of agent, [155] suggests four main characteristics of any agent: autonomy, social ability, reactivity and proactivity. The characteristic of autonomy has similar meaning as suggested by the definition in [45], which was reviewed in the previous paragraph. The characteristic of social ability suggests the need for an agent to interact with other agents to fulfil common objectives. The characteristic of reactivity has been included by most definitions of agent [45,103,118]. This characteristic requires agent to perceive its environment and respond in a timely fashion to changes that occur in the environment. Lastly, [155] suggest proactivity as crucial feature in agents to demonstrate their goal-oriented behaviour by taking initiative. The explained above four characteristics of agent can also be ascribed to all humans. With this rationale, the monograph [135] considers humans as agents in addition to software agents and robots.

2.1.2 Overview of AOSE Methodologies

Agent-Oriented Software Engineering (AOSE) is a software engineering field that aims to apply predefined processes and best practises to the development of complex agent-oriented systems [66]. AOSE methodologies focus on the application of software and hardware agents and organisations (communities) of agents [65]. Similarly, some other research contributions [94,135] categorise agents into human agents and man-made agents. The latter are split further into software agents and hardware agents – robots or devices. Therefore, this thesis considers humans, as well as software and hardware components complying with the definition of agent in Section 2.1.1 as agents that interact to achieve the common goal – purpose – of a sociotechnical system. For example, a biometric authentication process where a device accepts human input and compares the human input with information in the database can be modelled as an interaction between a human agent and an agent implemented in hardware.

Various AOSE methodologies in agent-oriented research community use the same notion of agent that was defined in Section 2.1.1. However, each of these methodologies has its own motivation and objective(s). Table 2-1 presents a list of nine AOSE methodologies and outlines their objectives.

Table 2-1: A list of AOSE methodologies.

| Name | Main Objective |
|------------------|---|
| O-MaSE [50] | To allow designers to create customized agent-oriented software development processes. O-MaSE is built on the MaSE methodology. |
| Prometheus [106] | To support a range of activities from requirements specification through to detailed design of agents that are based on goals and plans. |
| Tropos [18] | To cover the very early phases of requirements analysis, thus allowing for a deeper understanding of the environment where the software must operate, and of the kinds of interactions that should occur between software agents and humans. |
| Gaia [159] | To exploit the organizational abstractions, such as environment, roles, interactions, rules and structures, that provide clear guidelines for the analysis and design of complex and open software systems. |
| PASSI [32] | To design and develop multi-agent societies by integrating design models and concepts from both object-oriented (OO) software engineering and artificial intelligence approaches using the UML notation. |
| INGENIAS [110] | To combine agent concepts and methods established in MESSAGE/UML [19] in order to define contributions and default activities that help in planning an effort required for a given project. |
| ADELFE [13] | To design multi-agent systems that are complex, open, and not well specified, i.e., adaptive and self-organising systems. |
| ROADMAP [73] | To extend the Gaia methodology [159] by providing support for modelling complex open systems through requirements elicitation, a formal model for the environment and knowledge, a role hierarchy, explicit modelling of social aspects between agents, and modelling of dynamical changes. |
| RAP/AOR [139] | To employ a certain form of agent-based discrete event simulation for achieving more agility in the development process through the support of a foundational ontology. |

Sterling and Taveter [135] combined the ROADMAP [73] and AOR [139] methodologies to produce a set of systematic methods, vocabularies and notations for conceptualising sociotechnical systems, which are compliant with the model-driven development approach [98]. The AOM4STS methodology is further elaborated for requirements modelling [2] and design modelling [137].

2.1.3 Requirements Modelling for Sociotechnical Systems

Agent-oriented models for conceptual domain analysis act as a bridge between information technology (IT) and non-IT experts during the requirements elicitation and modelling phase in the development of a sociotechnical system [2,135]. These models provide a high level description of a sociotechnical system and use visual notations to enable all project stakeholders to reach a common understanding of the system requirements. Table 2-2 briefly describes agent-oriented models for conceptual domain analysis of sociotechnical systems.

Table 2-2: Summary of models for conceptual domain analysis.

| ID | Model Name | Objective |
|----|------------------------|--|
| 1 | Goal Modelling | To represent functional and non-functional requirements of the system as goals and quality goals, respectively, roles required for achieving the goals, and relationships among all of them. |
| 2 | Role Modelling | To identify responsibilities and constraints of each role in the system. |
| 3 | Organisation Modelling | To identify and represent the types of relationships between the roles of the system. |
| 4 | Domain Modelling | To represent the environments, the types of domain entities belonging to the environments, and the relationships between the roles, environments, and domain entities. |

2.1.4 Design Modelling of Sociotechnical Systems

Agent-oriented models of design aim to refine artefacts of the conceptual domain analysis layer by describing design features of the sociotechnical system independently of the technology [135,137]. Table 2-3 briefly describes agent-oriented models for platform-independent design of sociotechnical systems.

Table 2-3: Summary of models for platform-independent design.

| ID | Model Name | Objective |
|----|-------------------------------|---|
| 1 | Agent Modelling | To transform the abstract constructs, i.e., roles, to design constructs, i.e., agent types. |
| 2 | Acquaintance Modelling | To design interaction pathways between the types of agents in the system. |
| 3 | Interaction Modelling | To describe interaction patterns between the types of agents in the system. |
| 4 | Knowledge Modelling | To represent private and shared knowledge by the agents of the system about themselves and about the agents and objects in their environment. |
| 5 | Behaviour Interface Modelling | To identify behavioural units of the system and define an interface for each behavioural unit. |
| 6 | Behaviour Modelling | To describe the behaviours of agents of the given types. |

According to the available research literature [2,135,137], the AOM4STS methodology is a domain-independent methodology that facilitates conceptual modelling of sociotechnical systems. The following Section 2.1.4 outlines some of the applications of the AOM4STS methodology in different domains.

2.1.5 Applications of AOM4STS methodology

Since the introduction of the AOM4STS methodology [135], many researchers have actively applied this methodology in various research studies from different domains. Table 2-4 presents a list of case studies that have applied the AOM4STS methodology in various domains.

Table 2-4: A list of case studies that apply AOM4STS methodology.

| Author(s) | Focus Area |
|---------------------------------------|--|
| Miller, Tim, <i>et al.</i> | The use of goal modelling in the case study of emergency systems to address emotional needs of users that increases technology adoption and usage [94] |
| Shvartsman, Inna, <i>et al.</i> | Modelling of conflict resolution that has an impact on winning “hearts and minds” of the local population in military conflicts [129]. |
| Narendra, Nanjangud C., <i>et al.</i> | Modelling of sound conflict management for virtual enterprise collaboration in the case study of automobile production [100] |
| Mahunnah, Msury, <i>et al.</i> | Modelling the design of a sociotechnical system for reporting Critical Laboratory Results at the North Estonia Regional Hospital [85] |
| Cheah, WaiShiang, <i>et al.</i> | Modelling of factors that influence the sustainability of e-commerce for the rural community [26]. |
| Du, Hongying, <i>et al.</i> | Modelling of societal healthcare information system for simulation purposes [39]. |
| Zupancic, Eva, <i>et al.</i> | Modelling of a trust management sub-system for a sociotechnical system [161] |
| Norta, Alex, <i>et al.</i> | Modelling of a large sociotechnical service-ecosystem for the provision of emergency healthcare services [101] |
| Pedell, Sonja, <i>et al.</i> | Modelling of a domestic scenario that encourages engagement between grandparents and grandchildren separated by distance [111]. |

Despite the fact that researchers apply the AOM4STS methodology across different domains, the gap exists in the methods to be used for evaluating the artefacts of the AOM4STS methodology. With the objective to evaluate artefacts of the AOM4STS methodology, the following Section 2.2 reviews the CPN formalism [69], which is a formalism proposed in this thesis to support visualisation, validation and verification of design models produced by the AOM4STS methodology.

2.2 Coloured Petri Nets Formalism

There are various software tools that support formalisation of agent-oriented models. Rodin [1] supports system formalisation with the Event-B formalism that uses set theory and theorem provers to represent systems at different abstraction levels. This way, the Rodin software tool integrates system modelling, set theory and theorem provers that yield formalised systems. The PiVizTool [108] supports system design by pi-calculus. The initial purpose of the PiVizTool was to model and analyse choreographies of web services. Pi-calculus is a promising candidate for formalising system design models of the AOM4STS methodology. However, as [143] discusses, differently from Petri nets, pi-calculus is not a graphical modelling language. Consequently, pi-calculus makes system modelling more challenging for laymen. In comparison with pi-calculus, using mature CPN Tools [69,71] is easier and just understanding the usage of this toolset is sufficient for simulation, performance testing, and verification to generate quickly visualisation, validation and verification of design models produced by the AOM4STS methodology.

The use of Petri Nets formalisms [69] in problem domain analysis and system design is gaining prominence. For example, the conceptual framework AgOS [23] allows for a high level representation of an agent-oriented environment using classical Petri nets. The disadvantage of using classical Petri Nets in AgOS is the decrease of expressiveness for large systems in contrast to CPN that allows for modelling hierarchies. Because of this reason, CPN Tools is more suitable than AgOS for modelling scalable sociotechnical systems. Furthermore, Iqbal and Yousaf in [63] apply a formalism based on Petri nets to formalise the design of agents for the management of computing resources in clouds. Moreover, [102] automate the domain engineering process for developing product lines for agent-oriented systems, which include supporting agent variability and providing agent feature traceability, resulting in reduced time-to-market and lower development costs. These examples have a technical focus in using Petri nets for the design of agent-oriented systems and include automatic translation to Java code that a programmer should implement. However, the AOM4STS methodology [2,135,137] focuses on sociotechnical issues and this way recognizes the interactions between people and technology in workplaces and at homes.

2.2.1 Basic Concepts of Petri Nets

A Petri Net consists of places, transitions and arcs (also called arrows, or edges). An arc connects a place to a transition and vice versa. An arc cannot connect two places or two transitions. There are two types of places – input places and output places. An input place connects to a transition while an output place connects from a transition [14]. Mathematically, a Petri net is a triple (S, T, F) consisting of a countable set S of places and a countable set T of transitions with $S \cap T = \emptyset$, and a mapping $F: (S \times T) \cup (T \times S) \rightarrow \mathbb{N}$ which defines arcs (also called arrows or edges) between places and transitions. $F(s, t)$ defines the number of arcs from s to t . Analogously, $F(t, s)$ defines the number of arcs from t to s [97].

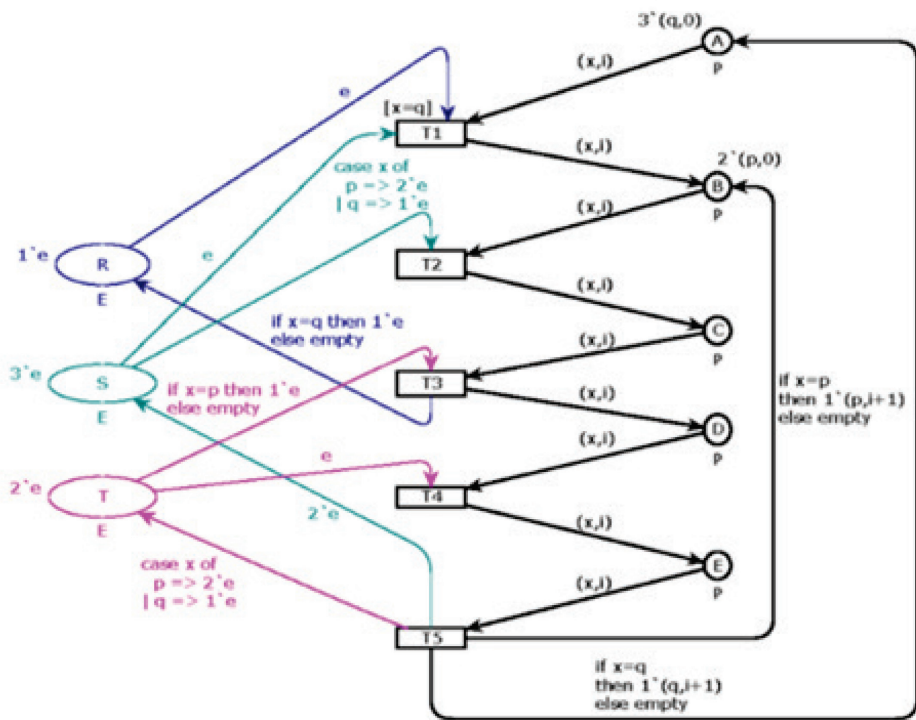
Graphically, a Petri net represents every place as a circle and every transition as a box (normally square and in general, rectangular). The places in a Petri net may contain a discrete number of marks called tokens. The distribution of these tokens in different places of the net presents a configuration of the net called a marking. A transition of a Petri net may fire if it is enabled, i.e. if there are sufficient tokens in all of its input places;

when the transition fires, it consumes the required input tokens, and creates tokens in its output places [116].

The execution of a Petri net is nondeterministic – if multiple transitions are enabled at the same time, they will fire in any order. Considering non-deterministic behaviour and the possibility of having multiple tokens in different places of the net (even in the same net), Petri nets are appropriate for modelling concurrent behaviours of a distributed system [69].

2.2.2 Coloured Petri Nets: The Language

Coloured Petri Nets (CPN) is a graphical language that combines the capabilities of Petri nets to provide a high-level language for constructing models of concurrent systems and analysing their properties [71]. CPN language is a general-purpose language that provides modelling capabilities for a very broad class of systems where concurrency and communication are key characteristics [69].



```

Declarations
colset U = with p | q;
colset I = int;
colset P = product U * I;
colset E = with e;
var x: U;
var i: I;

```

Figure 2-1: CPN model of a resource allocation system and variable declarations adapted from [69].

Figure 2-1 depicts a CPN model adapted from [69] that represents resource allocation system drawn by CPN Tools. The right side of this CPN model shows declarations of variables and colsets (data types) of the resource allocation system. As described in **Error! Reference source not found.**, the colset can either be simple colset, like colset I or a compound colset, like colset P, i.e., product of colset U and colset I. When simulating the CPN model, the initial state of the CPN model is referred to as an initial marking. After each step during the simulation process, a transition occurs and results in a new state of the given CPN model. The following Section 2.2.3 describes the analysis of CPN models through simulations.

2.2.3 Analysing Coloured Petri Net Models

When analysing CPN models, scenario-based validation considers a finite number of executions that helps to detect errors and increases the confidence in the correctness of the model, and thereby of the system [71]. However, scenario-based validation does not ensure 100% correctness of the model since it does not guarantee to cover all possible executions of the system. Therefore, full state space verification complements scenario-based validation by analysing all possible executions of the system [69]. The state space of a CPN model can be computed fully automatically by CPN Tools, which makes it possible to automatically verify, i.e., mathematically prove that the model possesses certain formally specified properties [70]. These properties include boundedness property, home property, liveness property and fairness property. However, this thesis focuses on home property and liveness property. The latter is further divided into dead markings, dead transitions and live transitions. The following paragraphs briefly describes home property and liveness property based on [69–71].

A *home marking* M_{home} is a marking that can be reached from any reachable marking. This means that it is impossible to have an occurrence of a sequence that cannot be extended to reach M_{home} . Moreover, a *dead marking* is a marking in which no binding elements are enabled. In some cases, the State Space Analysis (SSA) report shows only one dead marking. This means that the protocol as specified by the CPN model is partially correct – if the execution terminates then the protocol result is correct. In some other cases, the SSA report shows only one dead marking, which is also a home marking. This means that it is always possible to terminate the protocol with the correct result.

A transition is *dead* if there are no reachable markings in which it is enabled. When a SSA report does not show any dead transitions, each transition in the protocol can occur at least once. If a SSA report shows dead transitions then they correspond to parts of the CPN model that can never be activated. Hence, we can remove dead transitions from that CPN model without changing its behaviour. Furthermore, a transition is *live* if from any reachable marking, we can always find an occurrence of a sequence containing the transition. In other words, we cannot do things that will make it impossible for the transition to occur afterwards. We have already seen that our protocol has a dead marking, and this is the reason why it cannot have any live transitions – no transitions can be enabled from a dead marking.

In some cases, the analysis of models by CPN Tools generates a partial SSA. A partial SSA can either be caused by too much processing time during analysis or by a too big generated state space to be stored in the available computer memory. Consequently, this makes a partial SSA report not suitable for verification of design properties [69]. Moreover, dead markings do not imply that the design of the system is wrong but rather mean that in the resulting CPN models there are nodes without outgoing arcs.

Therefore, a prompt further analysis is necessary to cross check if the resulting dead markings are correct.

With the objective of prototyping validated and verified design models produced by the AOM4STS methodology, the following Section 2.3 reviews the JADE framework, which is a platform for developing agent-oriented systems. The JADE framework can be used for developing prototypical implementations based on design models produced by the AOM4STS methodology.

2.3 Java Agent Development Environments Framework

Although there are existing many agent platforms, the available research literature [79] shows that the JADE framework [10] is the most popular agent platform. JADE has been purely designed in Java and supports different kinds of web-based application systems and is compliant with the specifications by the Foundation for Intelligent Physical Agents (FIPA) [9].

2.3.1 FIPA Specifications

The Foundation for Intelligent Physical Agents (FIPA)¹ is an international non-profit association of companies and organisations that share the effort to produce specifications for generic agent technologies. These generic agent technologies support different application areas that developers can integrate to make complex systems with a high degree of interoperability. The generic agent technologies include an agent platform, agent management content language and ontology, which are described by [9,10] as follows. The agent platform includes three mandatory functionalities. The first functionality is the Agent Management System (AMS), which is the agent that supervises access to and usage of the platform; it is responsible for maintaining a directory of resident agents and for handling their life cycle. The second functionality is the Agent Communication Channel (ACC), which provides the path for basic contact between agents inside and outside of the platform. The ACC is a default communication method for agents, which offers a reliable, orderly and accurate message routing service. The third functionality is the Directory Facilitator (DF), which is the agent that provides a yellow pages' services to the agent platform.

Furthermore, the FIPA specifications define the Agent Communication Language (ACL) [44], which is used by agents to exchange messages. FIPA ACL is a language that describes message encoding and semantics, but it does not mandate specific mechanisms for message transportation.

An application-specific ontology [160] describes the elements that can be used as components of agent messages. An ontology is composed of two parts: a vocabulary that describes the terminology of concepts used by agents in their space of communication and the nomenclature of the relationships between these concepts, and their semantic and structure [104]. The ontology is implemented by extending the class Ontology predefined in JADE and adding a set of element schemas describing the structure of concepts, actions, and predicates that can be used to compose the content of your messages. The ontology may also extend directly the basic ontology classes BasicOntology or ACL Ontology according to the application need. Extending either of

¹ <http://www.fipa.org/>

these two classes also indirectly extends the Ontology class because the two classes are subclasses of the Ontology class.

2.3.2 JADE Run-Time System Architecture

JADE framework is a distributed system that can split over various hosts with one of them acting as a front end for inter-platform Internet Inter-Orb Protocol (IIOP) communication. The IIOP provides the protocol to connect different agent platforms [10]. Figure 2-2 depicts the software architecture of one agent platform.

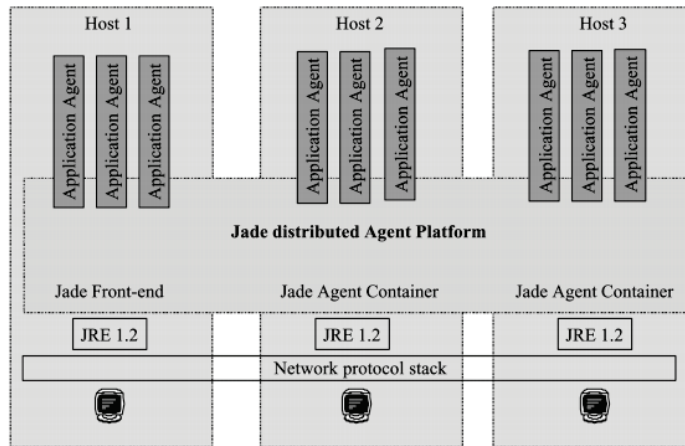


Figure 2-2: Software architecture of a JADE framework [10].

A JADE system comprises one or more Agent Containers, each of which has been deployed in a separate Java Virtual Machine (JVM) and delivers a run-time environment support to several JADE agents. Java RMI is used to communicate among the containers, each of which can also act as an IIOP client to forward outgoing messages to foreign agent platforms. A special Front End container is also an IIOP server that listens to the official agent platform ACC address for incoming messages from other platforms. The two mandatory system agents – AMS and DF – run within the front-end container. New agent containers can be added to a running JADE platform; as soon as a non-front-end container is started, it follows a simple registration protocol with the front-end container and adds itself to a container table maintained by the front-end container [10].

Complex tasks in agent-oriented systems are usually tackled using collaboration among many agents, whereby a single agent is typically a strongly cohesive piece of software. On the other hand, asynchronous message passing with the message consumption model of the “pull” type leads to a very loose coupling between different agents. Furthermore, no implementation inheritance and hence no code reuse is considered when dealing with software agents. This way, software agents bear a strong resemblance to actors, and the JADE execution model indeed has its roots in actor languages. In JADE, the abstraction used to model agent tasks is called Behaviour: each JADE agent holds a collection of behaviours which are scheduled and executed to perform agent duties. Behaviours represent logical threads of the implementation of a software agent [10].

2.4 Agent-Oriented Tools for Modelling Sociotechnical Systems

This Section reviews the existing tools for agent-oriented modelling. This is followed by a review of the tools for modelling sociotechnical systems. Lastly, an empirical evidence is provided on the effectiveness of such tools.

2.4.1 Tools for Agent-Oriented Modelling

The presence of software tools promotes the usage of agent-oriented software engineering for complex system development [113]. According to [47], only a few tools support the full development process of agent-oriented systems. The following paragraph provides an overview of some of these tools.

In the software tool described by [47], designers define the phases and activities of the development of agent-oriented systems and identify the relationships between modelling activities. Moreover, the tool described by Fuentes-Fernández *et al.* [47] allows designers to complete an agent specification phase and generate running code. Another relevant tool – AgentTool Process Editor (APE) – is a software tool implemented as an Eclipse plug-in that supports the design, validation and management of agent-oriented systems according to the O-MaSE AOSE methodology [49]. In particular, the APE tool allows developers to create customised agent-based processes based on O-MaSE; maintain and update the O-MaSE method library; verify custom processes to ensure compliance with O-MaSE; and fully integrate the custom processes into the Eclipse development environment. Freitas *et al.* [46] introduce a tool that enables the transformation of conceptual models into the implementations of agent-oriented system. Also, Yu *et al.* [157] describe an Integrated Development Environment (IDE) for modelling a system behaviour based on the Goal Net model. Moreover, this software tool supports software engineers in simplifying design processes of agent-oriented systems. Furthermore, this software tool facilitates automatic generation of design data of intelligent agents with the help of the Multi-Agent Development Environment (MADE). Finally, Manzoor and Zafar [90] describe a Multi-Agent Modelling Toolkit (MAMT) that uses Agent Unified Modelling Language (AUML) models to support designers during rapid development of complex systems. Although the literature reports on the existence of various agent-oriented tools, more research work is required on agent-oriented tools for sociotechnical systems.

2.4.2 Tools for Sociotechnical Systems

This Section reviews the previous research studies on the usage of software tools that apply agent-oriented approach for requirements engineering and design of systems by considering social and technical features.

On the one hand, TAOM4e¹ is a graphical modelling framework realised as a plug-in for the Eclipse² project to support modelling in all phases of the Tropos methodology [96], including Early Requirements (ER) and Late Requirements (LR) engineering. The Tropos methodology [18] uses the notions of actor, goal, plan, resource, dependency and capability to produce domain documentation. During the ER engineering phase, ER actor models and ER goal models describe “as-is-system”. The deliverables of the ER engineering phase are then elaborated into LR actor diagram,

¹ <http://selab.fbk.eu/taom/>

² <https://www.eclipse.org/eclipse/>

LR goal diagram and LR capability model that describe “to-be-system” during the LR engineering phase.

Another tool for engineering sociotechnical systems is the sociotechnical system-Tool¹, which is a Computer Aided Software Engineering (CASE) tool that has primarily been developed to support Sociotechnical Security modelling language (STS-ml) [33]. The sociotechnical system-Tool allows stakeholders to apply the concepts of actor, goal and delegation for expressing the security requirements of sociotechnical systems. The design of the sociotechnical system-Tool entails three different views – social, information and authorization views – where each view displays specific elements of the sociotechnical system to be designed and hides others [107]. The social view captures goals, interactions between actors and information exchange. The information view hides interactions and goals and shows structured information and documents. Lastly, the authorisation view captures the authority granted by actors to other actors over given information or document.

The TAOM4e, sociotechnical system-Tool, and AOM4STS tool make use of the modelling concepts employed by the respective methodologies and modelling languages Tropos, sociotechnical system-ml, and AOM4STS. The concepts that are used for requirements engineering in Tropos [96] and sociotechnical system-ml [33] are also available in the AOM4STS methodology [135] but are either represented by different names and notations or carry different meanings. For example, actors in Tropos and sociotechnical system-ml are captured as roles in the AOM4STS methodology and have the same meaning but use different notations. Similarly, dependencies in Tropos are presented as delegations in sociotechnical system-ml and are captured as types of relationships between roles in the AOM4STS methodology. However, dependencies, delegations, and relationship types carry different meanings. For example, a delegation in sociotechnical system-ml captures the delegation of a goal achievement from one actor to another, while a dependency in Tropos captures either a delegated goal, requested plan or shared resource. Differently, the relationship type in the AOM4STS methodology captures the type of delegation, which can be peer, control or benevolence. Lastly, goals in Tropos, sociotechnical system-ml and AOM4STS have different names and notations and carry different meanings. Tropos has hard-goals and soft-goals. Hard-goals represent functional requirements while soft-goals represent non-functional requirements from the system perspective, such as scalability, performance, maintainability, and so on. On the other hand, sociotechnical system-ml has goals, security needs and threats, where goals capture functional requirements, while security needs and threats capture non-functional security requirements from the system perspective and from the human perspective, such as trustworthiness, safety, and so on. Lastly, the AOM4STS methodology has functional goals, quality goals and emotional goals. Functional goals capture functional requirements, quality goals capture non-functional requirements from the system perspective and emotional goals capture non-functional requirements from the human perspective (feelings) [94], such as safe, cared, independent, in control, in touch, and so on.

From this review, it is evident that all the goal types in Tropos and sociotechnical system-ml can be captured by the AOM4STS methodology in a more generalised form but not vice versa, i.e., Tropos does not capture emotional goals of any type while

¹ <http://www.sts-tool.eu/>

sociotechnical system-ml captures only security-based emotional goals. Therefore, the differences in notations and meanings between the modelling concepts of the AOM4STS methodology and other agent-oriented methodologies need to be considered when developing the AOM4STS software tool.

2.4.3 Tools for Model Driven Engineering

The AOM4STS methodology [2,135,137] stems from the paradigm of Model-Driven Engineering (MDE) [16] that focuses on the systematic use of models as primary engineering artefacts throughout the system engineering lifecycle. Among the key benefits of the MDE paradigm are effective expression of domain concepts [150], decreasing system development time (effort), and improving system quality [95].

Despite the benefits of the MDE paradigm, various studies show that a domain-specific MDE language is not enough for industry-wide adoption and a tool supporting such language increases the complexity of the development process instead of diminishing it [52]. Elsewhere, Whittle *et al.* [151] interviewed 39 practitioners on tool-related issues affecting the adoption of MDE. The results of this study indicate that the complexity of the modelling tools is among the major issues hindering practical application of MDE. Moreover, the study [151] suggests the need for developing new software modelling tools that focus on early design stages, support creativity in modelling, and match the way people think rather than the other way round. Another study involved 15 MDE experts in a thought experiment to identify the biggest problems with current MDE technologies [98]. The results of this study found that steep learning curves and arduous user interfaces are among significant usability challenges to industry-wide adoption of MDE tools.

Considering the benefits of MDE languages and the challenges of using MDE tools, Gorschek *et al.* [53] conducted a survey with 3785 developers to find out the extent to which design models are used before actual coding. The results of this study found that design models are not used very extensively in industry. Moreover, in companies where they are used, the notation is often not UML and the use of design models is informal and without tool support. Instead of relying on tools, the models are usually drawn on a whiteboard or paper.

The findings from this review of related work point to the need of conducting research studies on MDE software tools to empirically compare claimed benefits of a modelling tool against modelling on a whiteboard or paper. Furthermore, the summary of papers presented in the workshop on the experiences and empirical studies in software modelling [24] suggests the need to conduct more empirical studies on the evaluation of modelling techniques, languages and tools in order to assess their advantages and disadvantages, to ensure their applicability in different contexts, their ease of use, and other issues such as required skills and costs. The papers overviewed by [24] include a study that assessed the frequency of empirical evaluation in software modelling research [22] by reviewing 266 papers. The study found that 195 (73%) of the publications did not report about any empirical evaluation. This finding clearly indicates the need for more empirical studies in software modelling research.

2.5 Summary

This Chapter provided the background of the AOM4STS methodology in Section 2.1, which has been extended in this thesis by proposing the guidelines for simulating and prototyping sociotechnical systems. Furthermore, this Chapter gave a comprehensive review of the Coloured Petri Nets (CPN) formalisms and the JADE framework in the respective Sections 2.2 and 2.3. The former laid down the foundation for the proposed CPN modelling guidelines in Chapter 3, which support visualisation, validation and verification of design models produced by the AOM4STS methodology. The latter established the foundation for the proposed in the thesis JADE prototyping guidelines in Chapter 4, which support prototyping sociotechnical systems based on design models produced by the AOM4STS methodology. Lastly, Section 2.4 reviews the existing software tools that employ model-driven engineering approach to support agent-oriented modelling of sociotechnical systems. Subsequently, established the foundation for the development of AOMSTS tool described in Chapter 5 and empirically evaluated in Chapter 6.

3 Simulation by Coloured Petri Nets Tools

3.1 Introduction

Our society is becoming increasingly dependent on complex information systems for carrying out daily activities. The complexity of information systems mainly stems from the need for integration and orchestration of independently managed software systems that are distributed in dynamic environments [132] for problem domains, such as healthcare, aviation, air traffic control, and telecommunications. In addition, the behaviour of people who work across organizational, geographical, cultural and temporal boundaries [20] increases the complexity of the resulting sociotechnical systems and poses a great engineering challenge.

Researchers have undertaken various studies in designing sociotechnical systems from interacting technical, societal, and organisational aspects. These studies have focused on domains such as healthcare [87], military training [130], and domestic applications [111] using an agent-oriented modelling approach [135]. The latter is a top-down holistic approach for designing sociotechnical systems by engaging all stakeholders during the problem domain analysis and system design phases of a system's development life cycle. However, a gap exists in formalising and evaluating agent-oriented behaviour, knowledge and interaction models before the actual implementation of sociotechnical systems based on these models.

In this chapter, we fill the identified knowledge gap by answering the following research question: How to provide the support for visualisation, validation and verification of design models of sociotechnical systems produced by the AOM4STS methodology? To establish complexity-reducing separation of concerns, we deduce the following sub-questions: What guidelines support the modelling of design models of sociotechnical systems produced by the AOM4STS methodology in CPN Tools? What is the effectiveness of the proposed guidelines for modelling design models of sociotechnical systems produced by the AOM4STS methodology in CPN Tools? What is the utility of CPN Tools in visualisation, validation and verification of design models produced by the AOM4STS methodology?

This set of sub-questions assumes that the design of syntactically correct agent-oriented behaviour models precedes the mapping to formalizations that carry equivalent model properties.

The structure of this Chapter is as follows. Section 3.2 proposes mapping guidelines towards a formalisation and evaluation of agent-oriented models for distributed sociotechnical systems. Sections 3.3 uses a real-life intruder handling case study to demonstrate the application of the mapping guidelines to transforming agent-oriented models to the corresponding formal representation. Section 3.4 reports the results of empirical study that evaluates the effectiveness of the proposed CPN modelling guidelines. Section 3.5 presents the conclusion and provides future work.

3.2 Guidelines for Representing Agent-Oriented Models in CPN Tools

This Section presents a comprehensive description of the guidelines for transformation of system design models created by the AOM4STS methodology to CPN models for validation and visualisation through simulation in CPN Tools. For clarity, the

transformation guidelines are categorised according to the knowledge, interaction and behaviour models, which are respectively described in Sections 3.2.1, 3.2.2, and 3.2.3.

3.2.1 Guidelines for transformation of knowledge models

Agents need knowledge to perform their activities and make decisions. In the AOM4STS methodology, an agent uses knowledge attributes and conceptual objects to describe information about itself and other agents and objects in its environment. Knowledge attributes and conceptual objects are represented by knowledge model described in Section 2.1. The transformation guidelines introduced in this Section are the means to transform knowledge attributes and conceptual objects to CPN to validate by CPN Tools the knowledge models created by the AOM4STS methodology and visualise knowledge sharing between agents.

3.2.1.1 Knowledge Attributes

Intent: To represent qualities of an agent or conceptual object.

Problem Description: Agent uses knowledge attributes to represent qualities of itself and qualities of conceptual objects in its environment. For example, an agent of the type Person can be characterised by the date of birth, height, weight, hair colour, eye colour, and so forth. This way, an agent can easily differentiate itself from other agents of the same type. Consequently, it becomes easier to identify an agent in a sociotechnical system. The most frequent types of knowledge attributes include *String*, *Integer*, *Real*, *Boolean*, *Date* and *Enumeration*. The challenge here is finding the best way to represent knowledge attributes in CPN. Figure 3-1 represents key components of a knowledge model in the AOM4STS methodology: agent type *AgentType1* and conceptual object type *ResourceType1*. Each of them contains a set of knowledge attributes of various types.

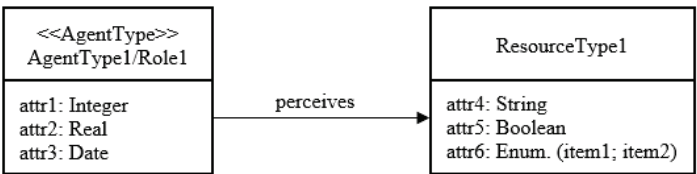


Figure 3-1: Sample representation of components knowledge model components.

Solution: This transformation guideline uses simple *built-in data types* in CPN Tools such as *integer* and *string* to represent knowledge attributes from conceptual knowledge model in CPN. Listing 3-1 represents the result of transforming knowledge attributes that describe the agent type *AgentType1* and conceptual object type *ResourceType1* in Figure 3-1 into the format of CPN used in CPN Tools:

Listing 3-1: Declaration of knowledge attributes in CPN Tools.

```
Colset
  colset INT = int;
  colset STRING = string;
  colset BOOL = bool;
  colset TIME = time;
  colset REAL = real;
  colset ITEMS = with item1 | item2;
Variables
  var attr1: INT;
  var attr2: REAL;
  var attr3: TIME;
  var attr4: STRING;
  var attr5: BOOL;
  var attr6: ITEMS;
```

3.2.1.2 Conceptual Objects

Intent: To represent knowledge of an agent about other agents and objects in its environment.

Problem Description: Agent uses conceptual objects to represent knowledge about other agents and objects in its environment. Conceptual objects may be viewed as resources consumed by agents. Conceptual objects normally are described in terms of knowledge attributes. For example, a medical prescription can be characterised by patient name, patient address, prescriber name, prescriber address, prescriber registration number, drug(s) prescribed, and so on. This way, an agent can possess or share a large amount of information using only one conceptual object. The challenge here is to find a suitable way to represent conceptual objects in CPN. Figure 3-2 represents conceptual object type *ResourceType2* that contains a set of knowledge attributes belonging to various data types.

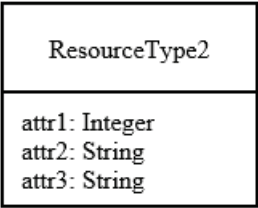


Figure 3-2: Sample representation of a conceptual object type in a knowledge model.

Solution: This transformation guideline helps the designer to create a *user-defined data type* in CPN Tools according to the knowledge contained by the corresponding conceptual object type. Normally, a *user-defined data type* combines two simple *built-in data types*, such as *integer* and *string*, to form INTxSTRINGxSTRING data type. Listing 3-2 shows the result of mapping the conceptual object type represented in Figure 3-2 to the format of CPN used in CPN Tools:

Listing 3-2: Declaration of conceptual object types in CPN Tools.

```
Colset
  colset INT = int;
  colset STRING = string;
  colset INTxSTRINGxSTRING = product INT*STRING*STRING;
Variables
  var Resource2: INTxSTRINGxSTRING;
```

3.2.2 Guidelines for transformation of interaction models

The key properties of sociotechnical systems described in Section 1.1 include full distribution of knowledge and collaborating parties (agents). Consequently, each agent of a sociotechnical system can only possess incomplete knowledge about the problem domain. Therefore, interactions between agents become the key aspect of sociotechnical system for knowledge sharing. The following constructs provide the guidelines of how to represent in CPN agent interaction constructs of the AOM4STS methodology:

3.2.2.1 Sending Message

Intent: To describe an agent asynchronously sending a message to another agent.

Problem Description: Agents are naturally distributed and work in collaboration to perform assigned to them tasks in order to achieve the overall goal of the system [112]. Consequently, agents often send messages to other agents. One of the key concepts of sociotechnical systems is asynchronous communication, which means that agents exchange messages without the use of an external clock signal [30]. Sending a message asynchronously means that the sender can send a message while the receiver is offline or is engaged in other activities. In such a case the receiver stores an incoming message in its incoming messages' buffer.

For each message, the sending agent needs to specify an appropriate receiving agent and the content of the message. When designing an interaction between agents, it is important to specify necessary condition(s) that trigger an activity of sending a message. Figure 3-3 describes an agent sending a message. The transformation guideline consists of the activity type Main Activity Type1 performed by the agent sending a message. This activity type contains the activity type SubActivity Type1 for sending an instance of Message Type1. An activity of the type SubActivity Type1 executes when an agent fulfils the condition stated for rule R1.

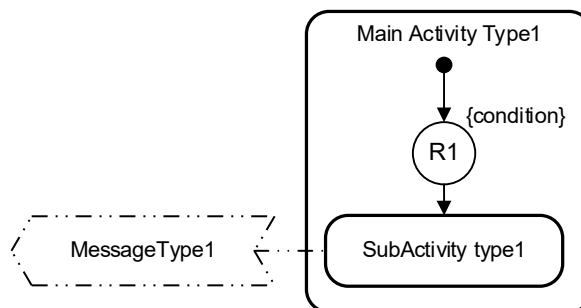


Figure 3-3: An agent sending a message.

Solution: Figure 3-4 presents the results of transforming into CPN the agent interaction model of an agent sending a message depicted in Figure 3-3. The transition SubActivity Type 1 in Figure 3-4 contains two incoming arcs from the places *Precondition1* and *Trigger1*. The place *Precondition1* stores the identity of an agent (*sid*) that waits to send the message while the trigger contains the identity of the expected receiver of the message (*rid*) and the message to be sent (*messageType*). Additionally, the transition SubActivity Type1 has the label *[messageType = value]*, which defines the rule R1 from Figure 3-3. When the rule R1 is triggered, the CPN Tools performs the transition SubActivity Type1 and transfers to the place *Postcondition1*. Otherwise, the CPN Tools will never perform the transition SubActivity Type1. Listing 3-3 shows the result of transforming the agent interaction model of sending a message shown in Figure 3-3 to the format of CPN used by CPN Tools, preceded by the corresponding behavioural interface model:

| AID | Activity Name | Trigger | Precondition(s) | Postcondition(s) |
|-----|-------------------|----------|-----------------|------------------|
| 1 | SubActivity Type1 | Trigger1 | Precondition1 | Postcondition1 |

Listing 3-3: Declaration of the construct for sending a message in CPN Tools.

```

MSC Setup
  val msc = MSC.createMSC("Sequence Diagram");
  val sender = "Sender";
  val receiver = "Receiver";
  val _ = MSC.addProcess(msc, sender);
  val _ = MSC.addProcess(msc, receiver);
Colset
  colset INT = int;
  colset STRING = string;
  colset INTxSTRING = product INT*STRING;
  colset INTxINTxSTRING = product INT*INT*STRING;
Variables
  var messageType: STRING;
  var sid, rid: INT;
Values
  val value = "messageType1";
Functions
  Fun send_message(senderID, receiverID, message)=
    MSC.addEvent(msc, sender, receiver, "SUBACTIVITY 1:"^message);

```

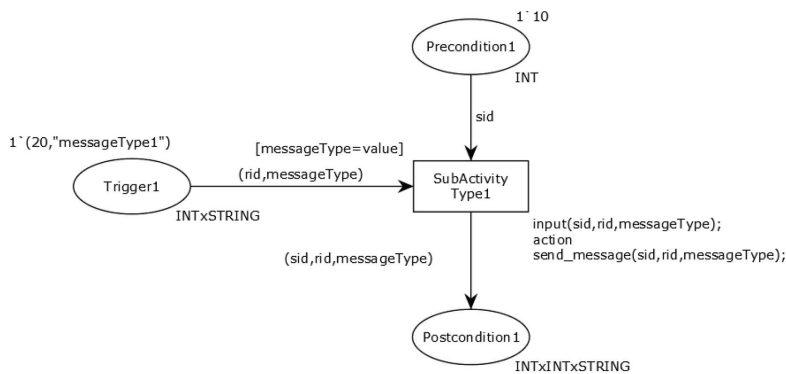


Figure 3-4: CPN representation of an agent sending a message.

3.2.2.2 Receiving Message

Intent: To describe an agent asynchronously receiving a message sent by another agent.

Problem Description: One of the key concepts of sociotechnical systems is asynchronous communication, which means that agents exchange messages without the use of an external clock signal [30]. Receiving a message asynchronously means that the receiving agent does not have to stop its other activities for dealing with an incoming message but the receiver stores the incoming message in its incoming messages' buffer and fetches it from there for processing at the time deemed appropriate by the agent. Therefore, it is important to identify necessary condition(s) that may trigger an agent to receive asynchronously a message sent by another agent. Figure 3-5 describes an agent asynchronously receiving a message. This agent receives an instance of MessageType1 and performs an activity of the type SubActivity Type1 after fulfilling the necessary condition contained by the rule R1.

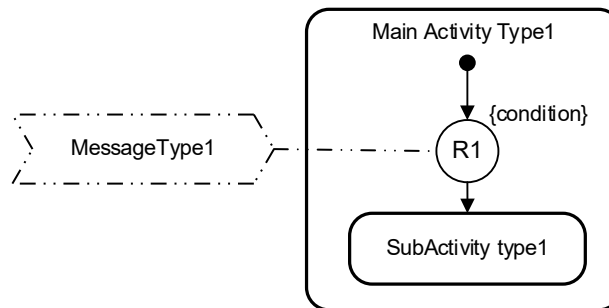


Figure 3-5: An agent receiving a message.

Solution: Figure 3-6 presents the result of transforming into CPN the agent interaction model of an agent receiving a message represented in Figure 3-5. The transition SubActivity Type1 shown in Figure 3-6 contains two incoming arcs from the places *Precondition1* and *Trigger1*. The place *Precondition1* contains the identity of an agent (*rid*) that waits to receive a message while the latter contains an asynchronously sent message (*messageType*) that waits to be received by the given agent.

Additionally, the transition SubActivity Type1 holds the label *[messageType = value]*, which defines the rule R1 from Figure 3-5. When the rule R1 is triggered, the CPN Tools performs the transition SubActivity Type1 and transfers to the place *Postcondition1*. Otherwise, the CPN Tools will never perform the transition SubActivity Type1. Listing 3-4 represents the result of transforming the agent interaction model of receiving a message shown in Figure 3-5 into the format of CPN used by CPN Tools, preceded by the corresponding behavioural interface model:

| AID | Activity Name | Trigger | Precondition(s) | Postcondition(s) |
|-----|-------------------|----------|-----------------|------------------|
| 1 | SubActivity Type1 | Trigger1 | Precondition1 | Postcondition1 |

Listing 3-4: Declaration of the construct for receiving a message in CPN Tools.

```

MSC Setup
  val msc = MSC.createMSC("Sequence Diagram");
  val sender = "Sender";
  val receiver = "Receiver";
  val _ = MSC.addProcess(msc, sender);
  val _ = MSC.addProcess(msc, receiver);
Colset
  colset INT = int;
  colset STRING = string;
  colset INTxINTxSTRING = product INT*INT*STRING;
Variables
  var messageType: STRING;
  var sid, rid: INT;
Values
  val value = "messageType1";
Functions
  fun receive_message(senderID,receiverID, message)=
  MSC.addEvent(msc, sender, receiver, "SUBACTIVITY
1:"^message);

```

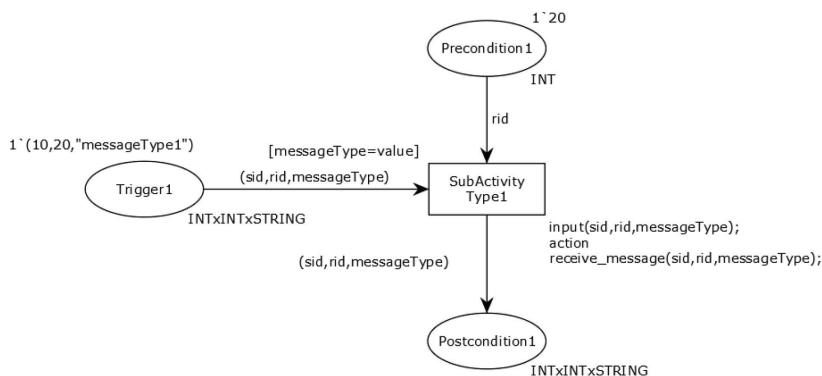


Figure 3-6: CPN representation of an agent receiving a message.

3.2.3 Guidelines for transformation of agent behaviour models

According to the properties of sociotechnical systems described in Section 1.1, each agent in sociotechnical system has its own set of activities that are selected by the agent itself according to its behavioural rules. These rules are triggered by the knowledge perceived by the agent from its environment or by some changes occurring in its internal state. The resulting execution of an agent's behaviour may affect the agent's environment. In the following, we will describe how agent behaviour models of the AOM4STS methodology can be represented in CPN:

3.2.3.1 Agent Initialization

Intent: To identify and show the availability of an agent instance in an open and distributed system.

Problem Description: In an open and distributed system, each collaborating agent type may have one or more agent instances that can enter and leave the system at any time. Therefore, during initialisation process of an agent, it is important to register an instance of the given agent type.

During agent initialisation, an agent instance acquires a unique identifier and makes itself ready for collaboration, i.e., for perceiving events and performing actions. For example, in the Java Agent Development (JADE) framework, Agent Management System (AMS) service is responsible for registering agents that enter the system and deregistering agents that leave the system [11]. Figure 3-7 models the conceptual initialisation process of an instance of AgentType1 according to the AOM4STS methodology. In Figure 3-7, the conceptual initialisation process is performed within an activity of the type ActivityType1.

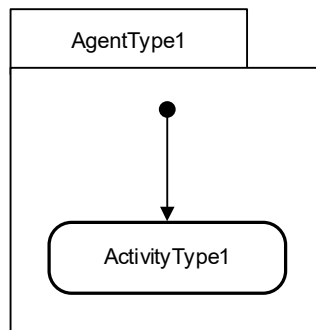


Figure 3-7: Agent initialisation.

Solution: During the initialisation process, an agent needs to provide its identity (*aid*) that enables the agent instance to communicate with other agent instances in the sociotechnical system by either sending or receiving messages. Figure 3-8 represents in CPN the initialisation process of an agent instance with the example value of *aid* 10. Listing 3-5 represents the result of transforming the conceptual model of agent initialisation shown in Figure 3-7 to the format of CPN used in CPN Tools, preceded by the corresponding behavioural interface model:

| AID | Activity Name | Trigger | Precondition(s) | Postcondition(s) |
|-----|---------------|-----------|-----------------|------------------|
| 1 | ActivityType1 | Trigger 1 | | Postcondition1 |

Listing 3-5: Declaration of the construct for initialising an agent in CPN Tools.

```

MSC Setup
    val msc = MSC.createMSC("Sequence Diagram");
    val agent = "Agent";
    val _ = MSC.addProcess(msc, agent);
Colset
    colset INT = int;
Variables
    var aid: INT;
Functions
    fun initialisation(agentID)=
        MSC.addInternalEvent(msc, agent, "INITIALIZE:" ^ INT.mkstr
            (agentID));

```

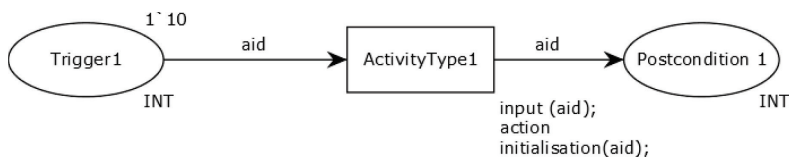


Figure 3-8: CPN representation of an agent initialisation.

3.2.3.2 Composite Activity

Intent: To describe the behaviour of an agent that executes an activity consisting of a set of sub-activities.

Problem Description: It is common to find an activity composed of a set of sub-activities. The execution of a set of sub-activities gives the outcome of the main activity. It is therefore important to correctly describe the connection between the main activity and its sub-activities.

Figure 3-9 illustrates an instance of the main activity of the type Main Activity Type1 consisting of two instances of sub-activities that are executed sequentially. For the main activity and both of its sub-activities of the respective types SubActivity Type1 and SubActivity Type2, there are the arcs showing both the control and data flow. Since sub-activities are contained by the main activity, the input of the first sub-activity of the type SubActivity Type1 comes from the input of the main activity. Similarly, the output of the main activity comes from the output of the last sub-activity of the type SubActivity Type2. Generalizing, the output of a sub-activity of the type SubActivity Type(n) becomes the input of the following sequential sub-activity of the type SubActivity Type(n+1).

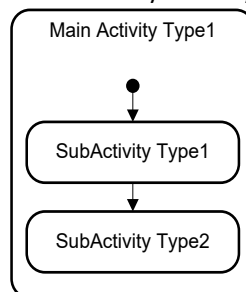


Figure 3-9: A composite main activity consisting of two sub-activities.

Solution: Figure 3-10 represents the transformation guideline for composite activity modelled in Figure 3-9 as two CPN models. The upper CPN model represents the CPN transition corresponding to the activity type Main Activity Type1 while the bottom CPN model represents the CPN transitions corresponding to the sub-activity types – SubActivity Type1 and SubActivity Type2. These two CPN models clearly show that the trigger of SubActivity Type1 is the same as the trigger of the Main-Activity Type1 and the post-condition of SubActivity Type2 is the same as the post-condition of Main-Activity Type1. Furthermore, the post-condition of SubActivity Type1 is the same as the trigger for SubActivity Type2. In Figure 3-10, the triggers, preconditions and postconditions are represented as the corresponding CPN places. Listing 3-6 represents the result of transforming the conceptual model of composite activity shown in Figure 3-9 to the format of CPN used in CPN Tools, preceded by the corresponding behavioural interface models:

| AID | Activity Name | Trigger | Precondition(s) | Postcondition(s) |
|-----|---------------------|-----------|-----------------|------------------|
| 1 | Main-Activity Type1 | Trigger 1 | | Postcondition1 |

| AID | Activity Name | Trigger | Precondition(s) | Postcondition(s) |
|-----|-------------------|----------------|-----------------|------------------|
| 1 | SubActivity Type1 | Trigger 1 | | Postcondition2 |
| 2 | SubActivity Type2 | Postcondition2 | | Postcondition1 |

Listing 3-6: Declaration of the construct for composite activity in CPN Tools.

```

MSC Setup
    val msc = MSC.createMSC("Sequence Diagram");
    val agent = "Agent";
    val _ = MSC.addProcess(msc, agent);
Colset
    colset STRING = string;
Variables
    var messageType: STRING;
Functions
    fun SubActivity_Type1(message)=
    MSC.addInternalEvent(msc, agent, "CONTENT:"^message);
    fun SubActivity_Type2(message)=
    MSC.addInternalEvent(msc, agent, "CONTENT:"^message);

```

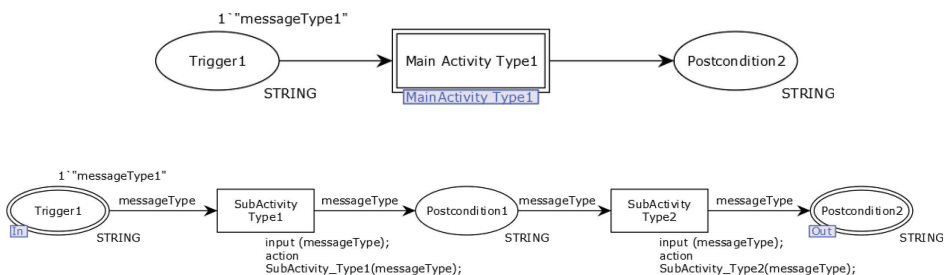


Figure 3-10: CPN representation of an activity containing two sub-activities.

3.2.3.3 Reactive Behaviour

Intent: To describe the behaviour of an agent after the agent has perceived changes in its internal knowledge and/or external environment.

Problem Description: Reactivity is among important characteristics of an agent in sociotechnical systems. Reactivity is a system behaviour in which every single agent in the sociotechnical system copes with the environmental changes by providing a specific solution to reorganize its own task in order to fulfil the accomplishment of its originally assigned goal [62]. A reactive agent continuously observes the environment and detects changes that trigger certain behaviours after satisfying given conditions. Figure 3-11 describes a reactive behaviour of an agent that is triggered when a precondition of the rule R1 is satisfied by changes in its internal knowledge and/or the environment. The agent represented in Figure 3-11 executes a sub-activity of the type SubActivity Type1 in response to the detected change that fulfils the precondition. Otherwise, the agent executes a sub-activity of the type SubActivity Type2.

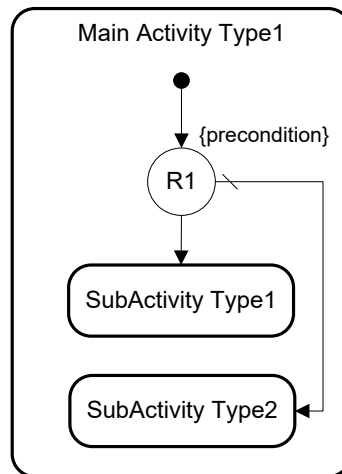


Figure 3-11: Reactive behaviour of an agent.

Solution: Figure 3-12 shows the results of applying this transformation guideline in CPN Tools. The resulting CPN model depicted in Figure 3-12 represents two CPN transitions corresponding to sub-activities of the respective types SubActivity Type1 and SubActivity Type2 that are connected to the same CPN place *precondition1*. Moreover, the CPN transition SubActivity Type1 in Figure 3-12 contains two labels, namely, *[messageType = value]* and *P-HIGH*. The label *[messageType = value]* represents the rule R1 from Figure 3-11. The label *P-HIGH* enables to evaluate the place *precondition1*, which corresponds in Figure 3-11 to the evaluation of the precondition of the rule R1 before the execution of a sub-activity of the type SubActivity Type2. If the evaluation of the place *precondition1* in Figure 3-12 results in *TRUE*, CPN Tools will execute the transition SubActivity Type 1 and will transfer to the place *postcondition1*. Otherwise, CPN Tools will execute the transition SubActivity Type2 and will transfer to the place *postcondition2*. Listing 3-7 represents the result of transforming the agent behaviour model of reactive behaviour shown in Figure 3-11 to the format of CPN used by CPN Tools, preceded by the corresponding behavioural interface model:

| AID | Activity Name | Trigger | Precondition(s) | Postcondition(s) |
|-----|-------------------|---------|-----------------|------------------|
| 1 | SubActivity Type1 | | Precondition1 | Postcondition1 |
| 2 | SubActivity Type2 | | Precondition1 | Postcondition2 |

Listing 3-7: Declaration of a construct for a reactive behaviour in CPN Tools.

```

MSC Setup
  val msc = MSC.createMSC("Sequence Diagram");
  val agent = "Agent";
  val _ = MSC.addProcess(msc, agent);
Colset
  colset INT = int;
  colset STRING = string;
  colset INTxINTxSTRING = product INT*INT*STRING;
Variables
  var messageType: STRING;
Values
  val value = "messageType1";

Functions
  fun SubActivity_Type1(senderID, receiverID, message)=
  MSC.addInternalEvent(msc, agent, "CONTENT: "^message);
  fun SubActivity_Type2(message)=
  MSC.addInternalEvent(msc, agent, "CONTENT: "^message);

```

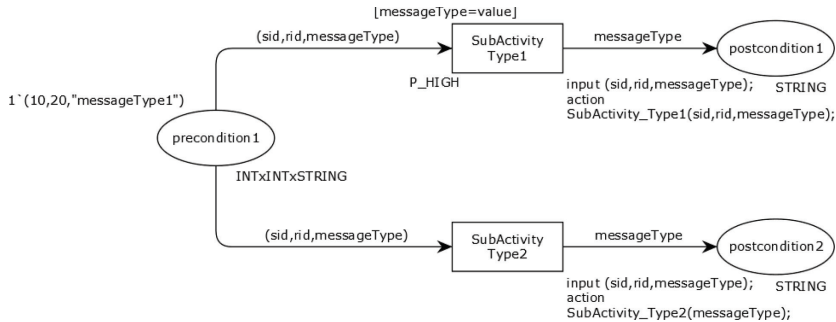


Figure 3-12: CPN representation of a reactive behaviour by an agent.

3.2.3.4 Looping Condition

Intent: To allow an agent to execute the same activity repeatedly while (a) given precondition(s) or post-condition(s) hold(s).

Problem Description: An agent behaviour consists of a set of activities, where each activity contains at least one precondition and one post-condition. Sometimes an agent needs to execute the same activity repeatedly while a given precondition or post-condition holds.

Figure 3-13 (a) describes the pre-conditional looping behaviour that occurs when an agent executes a sub-activity of the type SubActivity Type 1 repeatedly while the precondition of the rule R1 holds, and otherwise executes a sub-activity of the type

SubActivity Type 2 once without repetition. On the other hand, Figure 3-13(b) describes the post-conditional looping behaviour that occurs when an agent executes a sub-activity of the type SubActivity Type 1 repeatedly while the post-condition of the rule R1 holds, and otherwise executes a sub-activity of the type SubActivity Type 2 once without repetition. The main difference between these two conditional looping constructs is that in pre-conditional looping, the minimum times of executing a sub-activity of the type SubActivity Type 1 is zero, while in post-conditional looping, the minimum times of executing a sub-activity of the type SubActivity Type 1 is one.

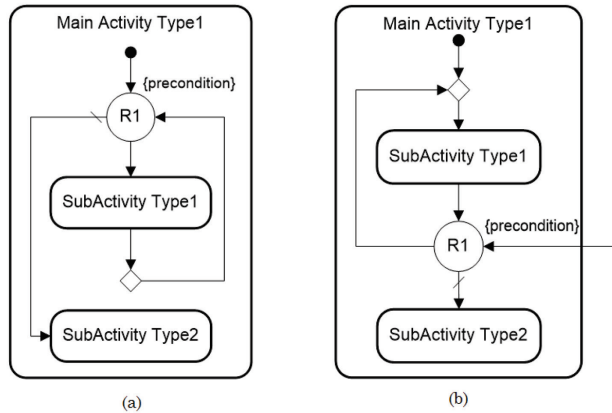


Figure 3-13: (a) Pre-conditional looping, (b) Post-conditional looping.

Solution (a): Figure 3-14 shows the results of applying this transformation guideline in CPN Tools. The resulting CPN model depicted in Figure 3-14 represents two CPN transitions corresponding to sub-activities of the respective types SubActivity Type1 and SubActivity Type2 that are connected to the same CPN place *precondition1*. Moreover, the CPN transition SubActivity Type1 in Figure 3-14 has two labels, namely, *[messageType = value]* and *P-HIGH*. The label *[messageType = value]* represents the rule R1 from Figure 3-13 (a). The label *P-HIGH* enables to evaluate the place *precondition1*, which corresponds in Figure 3-13 (a) to the evaluation of the precondition of the rule R1 before the execution of a sub-activity of the type SubActivity Type2. If the evaluation of the place *precondition1* in Figure 3-14 results in *TRUE*, CPN Tools will execute the transition SubActivity Type 1, and will return to the place *precondition1* to re-evaluate the place and if it yields the value *TRUE*, CPN Tools will re-execute the transition SubActivity Type 1. Otherwise, CPN Tools will execute the transition SubActivity Type2 and will transfer to the place *postcondition2*. Listing 3-8 represents the result of mapping the construct shown in Figure 3-13 (a) to the format of CPN used by CPN Tools, preceded by the corresponding behavioural interface model:

| AID | Activity Name | Trigger | Precondition(s) | Postcondition(s) |
|-----|-------------------|---------|-----------------|------------------|
| 1 | SubActivity Type1 | | Precondition1 | Postcondition1 |
| 2 | SubActivity Type2 | | Precondition1 | Postcondition2 |

Listing 3-8: Declaration of construct for pre-conditional looping in CPN Tools.

```

MSC Setup
  val msc = MSC.createMSC("Sequence Diagram");
  val sender = "Sender";
  val _ = MSC.addProcess(msc, sender);
Colset
  colset STRING = string;
Variables
  var messageType: STRING;
Values
  val value = "messageType";
Functions
  fun SubActivity_Type1(message)=
  MSC.addInternalEvent(msc, sender, "SUBACTIVITY 1:"^message);
  fun SubActivity_Type2(message)=
  MSC.addInternalEvent(msc, sender, "SUBACTIVITY 2:"^message);

```

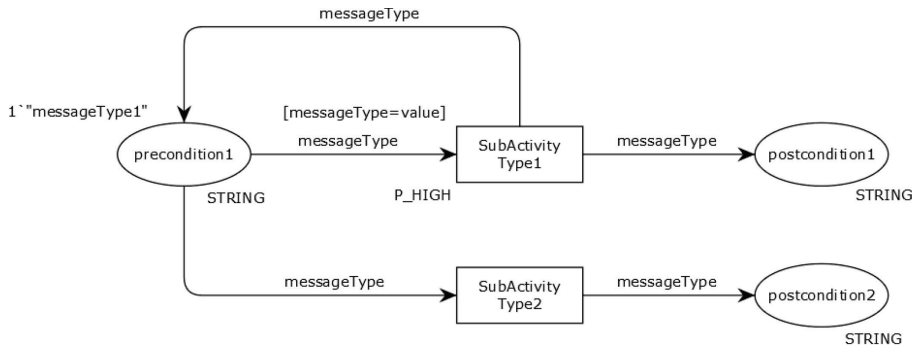


Figure 3-14: CPN representation of pre-conditional looping.

Solution (b): Figure 3-15 shows the results of applying this transformation guideline in CPN Tools. The resulting CPN model depicted in Figure 3-15 represents two CPN transitions corresponding to sub-activities of the respective types – SubActivity Type1 and SubActivity Type2 and the CPN transition Evaluate Rule. The CPN transition Evaluate Rule in Figure 3-15 has two labels, namely, *[messageType = value]* and *P-HIGH*. The label *[messageType = value]* represents the rule R1 from Figure 3-13 (b). The Evaluate Rule transition allows to execute always first the transition SubActivity Type1 because the label *[messageType = value]* representing the rule R1 from Figure 3-13 (b) is not attached to the transition SubActivity Type1. The label *P-HIGH* enables to evaluate the place *precondition1*, which corresponds in Figure 3-13 (b) to the evaluation of the precondition of the rule R1 before the execution of a sub-activity of the type SubActivity Type2. If the evaluation of the place *precondition1* in Figure 3-15 results in *TRUE*, CPN Tools will execute the transition SubActivity Type 1, and will return to the place *precondition1* to re-evaluate the place and if it yields the value *TRUE*, CPN Tools will re-execute the transition SubActivity Type 1. Otherwise, CPN Tools will execute the transition SubActivity Type2 and will transfer to the place *postcondition2*. Listing 3-9 represents the result of conceptual model shown in Figure 3-13 (b) to the format of CPN used by CPN Tools, preceded by the corresponding behavioural interface model:

| AID | Activity Name | Trigger | Precondition(s) | Postcondition(s) |
|-----|-------------------|---------|-----------------|------------------|
| 1 | SubActivity Type1 | | Precondition1 | Postcondition1 |
| 2 | SubActivity Type2 | | Postcondition1 | Postcondition2 |

Listing 3-9: Declaration of the construct for post-conditional looping in CPN Tools.

```

MSC Setup
  val msc = MSC.createMSC("Sequence Diagram");
  val sender = "Sender";
  val _ = MSC.addProcess(msc, sender);
Colset
  colset STRING = string;
Variables
  var messageType: STRING;
Values
  val value = "messageType";
Functions
  fun SubActivity_Type1(message)=
  MSC.addInternalEvent(msc, sender, "SUBACTIVITY 1:"^message);
  fun SubActivity_Type2(message)=
  MSC.addInternalEvent(msc, sender, "SUBACTIVITY 2:"^message);

```

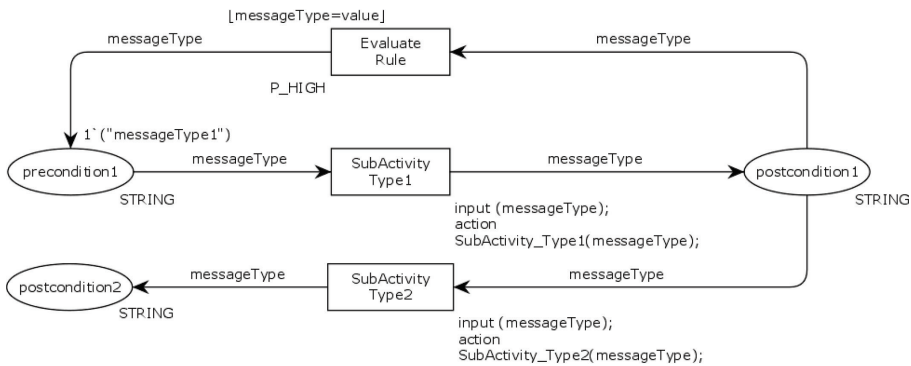


Figure 3-15: CPN representation of post-conditional looping.

3.2.3.5 Rule-Based Activity

Intent: To allow an agent to reason to execute the most appropriate activity.

Problem Description: All agents in a sociotechnical system aim to effectively and efficiently achieve the main purpose (goal) of the system [115]. Since the environment keeps changing, often each agent needs to reason and decide an appropriate set of activities to be selected for execution in order to achieve the intended goal [147]. Reasoning is amongst the main characteristics of an agent that is achieved through execution of rules stored in the agent's rule engine. Figure 3-16 depicts the agent behaviour model where an agent executes a sub-activity of the type SubActivity Type 1 when the precondition of the rule R1 is fulfilled.

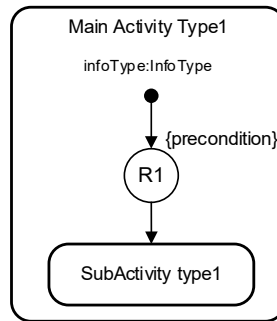


Figure 3-16: Rule-based activity.

Solution: Figure 3-17 shows the results of applying the guideline of rule-based activity in CPN Tools. The resulting CPN model consists of the transition SubActivity Type1 that accepts information through the input variable infoType coming from the place precondition1. Additionally, the transition SubActivity Type1 has the label [infoType = value], which represents the rule R1 modelled in Figure 3-16. When the precondition of the rule R1 is satisfied, the CPN Tools performs the transition SubActivity Type1 and thereafter transfers to the place postcondition1. Otherwise, the CPN Tools will never perform the transition SubActivity Type1.

Listing 3-10 represents the result of transforming the agent behaviour model shown in Figure 3-16 to the format of CPN used by CPN Tools, preceded by the corresponding behavioural interface model:

| AID | Activity Name | Trigger | Precondition(s) | Postcondition(s) |
|-----|-------------------|---------|-----------------|------------------|
| 1 | SubActivity Type1 | | Precondition1 | Postcondition1 |

Listing 3-10: Declaration of the construct for rule-based activity in CPN Tools.

```

MSC Setup
val msc = MSC.createMSC("Sequence Diagram");
val sender = "Sender";
val _ = MSC.addProcess(msc,sender);
Colset
colset STRING = string;
Variables
var infoType: STRING;
Values
val value = "infoType1";
Functions
fun SubActivity_Type1(info)=
  MSC.addInternalEvent(msc,sender,"SUBACTIVITY 1:"^info);

```

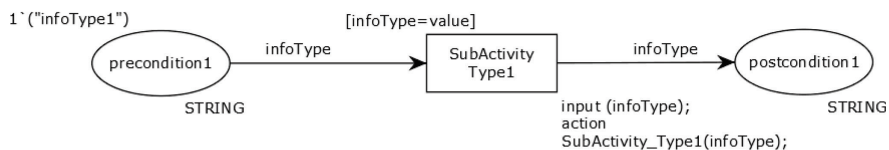


Figure 3-17: CPN representation of a rule-based activity.

3.2.3.6 Parameter Passing Between Activities

Intent: To allow an agent to pass information from one activity to another activity.

Problem Description: Normally an agent performs a set of activities when executing a certain goal. In many cases, these activities depend on each other, i.e., the output of a given activity becomes the input for the following activity. Therefore, it is important to enable an agent to seamlessly pass the information between two successive activities. Figure 3-18 describes an agent passing information from an activity of the type Activity Type1 to an activity of the type Activity Type2.

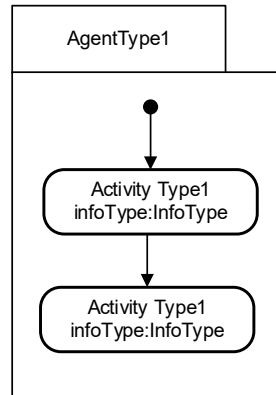


Figure 3-18: An agent passing knowledge between activities.

Solution: Figure 3-19 shows the results of applying the transformation guideline of parameter passing between activities in CPN Tools. The resulting CPN model consist of the transition Activity Type1 that accepts the input variable *infoType* that comes from the place *precondition1*. Consequently, the transition Activity Type1 results in CPN Tools transferring to the place *postcondition1* that also becomes a precondition (input) for the transition Activity Type2. This way the variable *infoType* is passed from the transition Activity Type1 to the transition Activity Type2. Listing 3-11 represents the result of transforming the agent behaviour model shown in Figure 3-19 to the format of CPN used by CPN Tools, preceded by the corresponding behavioural interface model:

| AID | Activity Name | Trigger | Precondition(s) | Postcondition(s) |
|-----|----------------|----------------|-----------------|------------------|
| 1 | Activity Type1 | Precondition1 | | Postcondition1 |
| 2 | Activity Type2 | Postcondition1 | | Postcondition2 |

Listing 3-11: Declaration of the construct for knowledge passing between activities in CPN Tools.

```

MSC Setup
  val msc = MSC.createMSC("Sequence Diagram");
  val sender = "Sender";
  val _ = MSC.addProcess(msc, sender);
Colset
  colset STRING = string;
Variables
  var infoType: STRING;
Functions
  fun Activity_Type1(info)=
  MSC.addInternalEvent(msc, sender, "ACTIVITY 1:"^info);
  fun Activity_Type2(info)=
  MSC.addInternalEvent(msc, sender, "ACTIVITY 2:"^info);

```

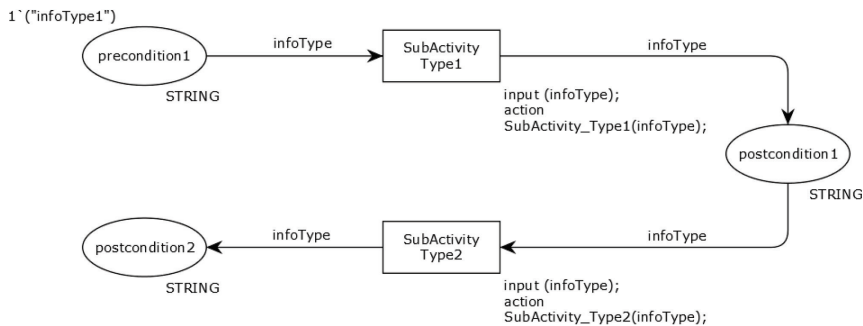


Figure 3-19: CPN representation of knowledge passing between activities.

The following Section demonstrates the utility of the transformation guidelines suggested in this Section by applying them to transform conceptual agent-oriented models of the intruder handling case study [134] to CPN models in order to visualise the behaviour of the system and validate the correctness of the design through simulation by CPN Tools.

3.3 Intruder Handling Case Study

This Section aims to demonstrate the utility of mapping guidelines put forward in Section 3.2 for transforming agent-oriented models to CPN Tools for evaluating syntactical correctness and soundness of agent-oriented models for sociotechnical systems. Furthermore, it depicts and explains the visualisation of sociotechnical interaction models through simulation in CPN Tools.

3.3.1 Description of the intruder handling scenario.

Residential home burglary is a serious social problem. Thieves like to break into houses when houses are unattended, especially during daytime. To prevent home break-ins, Closed-Circuit Television (CCTV) is installed to monitor any intruders who enter the house. If the CCTV detects any unfamiliar faces, it will send an intrusion alert to the house owner as well as to the police station. After receiving the alert, the police will be waiting for the confirmation of the detected intruder from the house owner and will

then further notify the police patrol officer on duty if needed. In addition, the police officer will contact the house owner for further actions.

3.3.2 Conceptual models of the intruder handling system

This Section uses the goal model from the abstraction layer of conceptual domain modelling, and knowledge model, agent behaviour model, behavioural interface model, and interaction frame diagram from the abstraction layer of platform-independent computational design for transforming the intruder handling system into CPN for simulation by CPN Tools. Interaction frame diagrams are needed because interactions between stakeholders play an important role in the intruder handling system.

The goal model represented in Figure 3-20 represents the goals, quality goals and roles of the intruder handling system and relationship between them. The overall goal of this system is to handle an intruder. The key role required for achieving this goal is Security Manager, which is therefore attached to the highest-level goal of the system. However, the presence of an intruder is a necessary condition for achieving this objective. Therefore, also the role Intruder is attached to the uppermost goal of the system. The key qualities for achieving the main goal of the system are *Appropriate* and *Timely*, which are attached to the highest-level goal as quality goals. The quality goal *Appropriate* means that the system needs to follow an established procedure for intruder handling such as informing police, house owner and expected guest(s), if any. The quality goal *Timely* means that the intruder handling system is a time-critical system. For example, the system needs to promptly alert the police – otherwise the system will fail.

Sub-goals represent different aspects of achieving their parent goal. In the goal model of the intruder handling system depicted in Figure 3-20, the sub-goals represent that to handle an intruder, the system needs to notice and identify the intruder and respond to detecting the intruder. Although the system needs to notice any movement, including the movement by the owner and guests, the system needs to accurately identify an intruder. As was described in the previous paragraph, the system needs to contact various stakeholders of the system after identifying an intruder. To reflect that, the sub-goals *Inform police*, *Inform visitors* and *Inform owner* elaborate the goal *Respond*. Furthermore, the corresponding role is attached to each sub-goal. For example, the role Police is attached to the sub-goal *Inform police*.

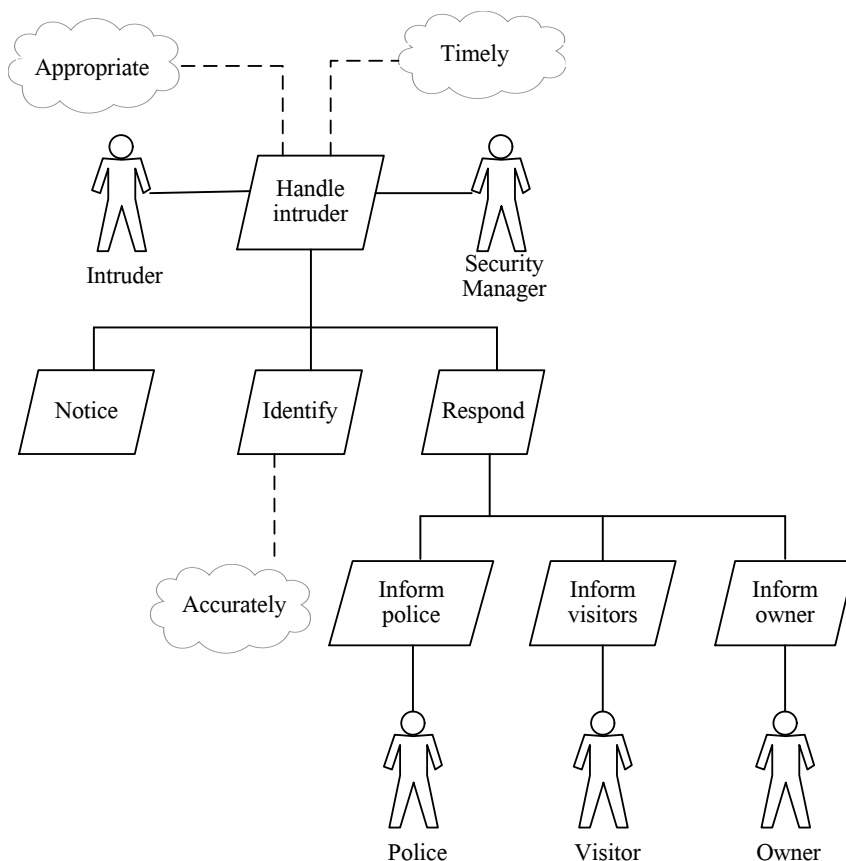


Figure 3-20: Goal model of the intruder handling system [135].

The identified roles in the goal model are then mapped to different agent types during the design phase. In this case study, we map five identified roles to five different agent types, namely, intruder agent, security manager agent, police agent, visitor agent and owner agent that represent intruder, security manager, police, visitor and owner roles respectively.

At the level of conceptual domain modelling, domain entities are used for representing knowledge handled by the sociotechnical system. Domain entities and relationships between them, as well as relationships between domain entities and roles, are captured by domain models. However, for mapping the models of AOM to CPN for fast prototyping by CPN Tools, we represent the knowledge to be handled by the intruder-handling system right away in a more detailed fashion at the level of platform-independent computational design as knowledge models. The knowledge can be either private or shared. Private knowledge is only known by an agent of one type, while shared knowledge is shared between agents of different types. Figure 3-21 represents a knowledge model of the intruder handling system, which contains three conceptual object types: PersonDescription, Suspect, and HouseSchedule. Instances of each identified conceptual object type are shared by agents of two or more types of the intruder handling system. For example, HouseSchedule is shared by the house owner

and security manager agents. Furthermore, the designer of the intruder handling system needs to identify the appropriate sets of attributes and the corresponding data types for each conceptual object type and include them in the knowledge model. For example, the conceptual object type HouseSchedule is characterized by the scheduleID, houseID, and Date attributes.

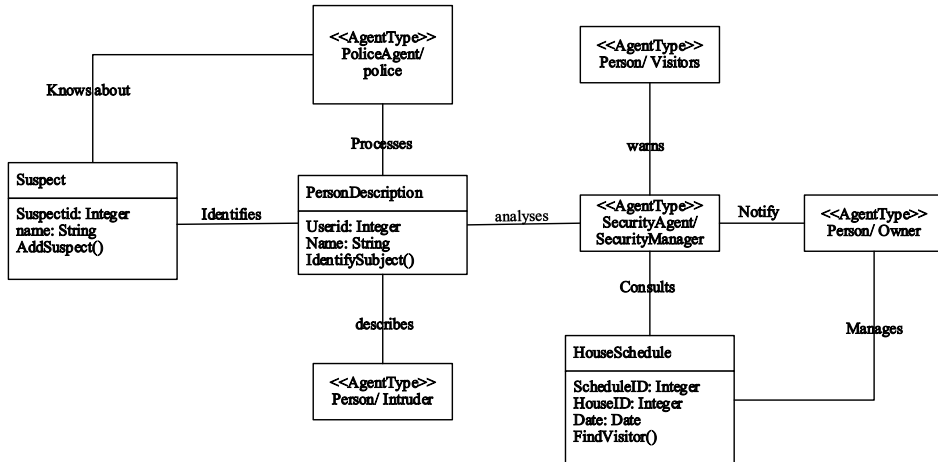


Figure 3-21: Knowledge model of the intruder handling system.

For an agent to autonomously and intelligently respond to events originating in its environment or in other agents, a set of rules is normally created and included in the agent behaviour model. Figure 3-22 represents the combined agent interaction model – interaction frame diagram – and agent behaviour models for each agent type of the intruder handling system. In addition to the interactions that occur between agents of the intruder handling system, the combined model represents activities, actions, and rules for each agent of the sociotechnical system. For example, the rule R2 in Figure 3-22 is triggered when a security agent notices a subject and then checks if the subject exists in its knowledge base. If checking for the condition *Subject exists* within the rule R2 returns true, the security agent updates its knowledge base with the knowledge that the detected subject is known. Otherwise, the security agent identifies the detected subject as an intruder and starts to alert the relevant agents for handling the intruder.

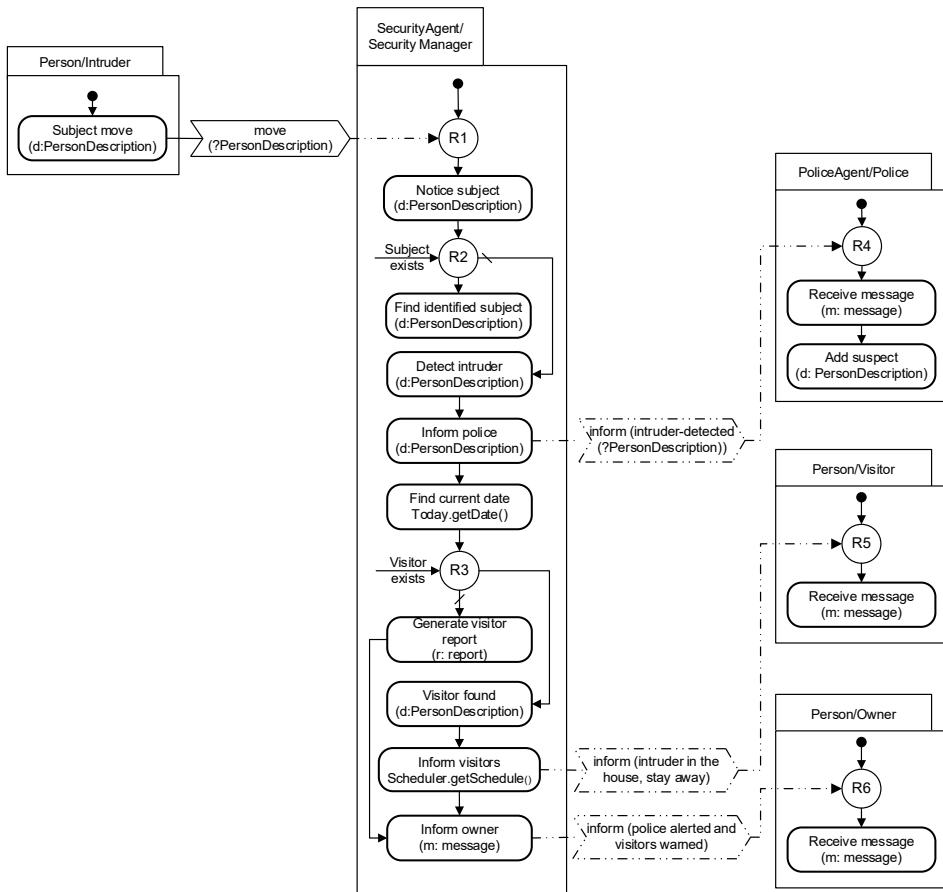


Figure 3-22: Combined interaction model and agent behaviour models of the intruder handling system.

To appropriately transform the agent-oriented conceptual models of the intruder handling system into CPN for simulation by CPN Tools, the behavioural interface model is created based on the agent behaviour models depicted in Figure 3-22. The behavioural interface model represented in Table 3-1 lists the set of activities and atomic actions – *move* and *inform* – for the intruder handling system. Each identified activity or atomic action has at least one precondition and at least one post-condition. When an activity or atomic action contains only one precondition, that precondition automatically becomes the trigger. However, when an activity has more than one precondition, any of the preconditions can become a trigger for the execution of that activity.

Table 3-1: Behavioural interface model of the intruder handling system.

| SNo | Pre-Condition(s) | Activity or Action | Post-Condition(s) |
|-----|---------------------------------------|-------------------------|--------------------------------------|
| 1 | Subject exist | Move | Subject moved |
| 2 | Subject moved Security agent exist | Notice | Subject noticed |
| 3 | Personal details DB | Find identified subject | Personal details DB Known subject |
| 4 | Subject noticed | Detect intruder | Intruder detected |
| 5 | Intruder detected | Inform police | Police msg sent Police informed |
| 6 | Police msg sent | Police receive msg | Received police msg |
| 7 | Police msg received | Add suspect | Suspect added |
| 8 | Police informed Dates | Find current date | Dates Current date |
| 9 | Current date | Generate visitor report | Visitor handled Generated report |
| 10 | Current date | Find visitor | Found visitor |
| 11 | Found visitor | Warn visitor | Visitor handled Visitor msg sent |
| 12 | Visitor msg sent | Visitor receive msg | Received visitor msg |
| 13 | Visitor informed House owner exist | Inform owner | Owner informed Owner msg sent |
| 14 | Owner msg sent | Owner receive msg | Received owner msg |

The following Section 3.3.3 describes a CPN simulation that provides visualisation and scenario-based validation of the agent-oriented conceptual models created for the intruder handling system.

3.3.3 Simulation of the intruder handling system

This Section describes how agent-oriented models for the intruder handling system were turned into for the corresponding representation in CPN based on the guidelines that have been put forward in Section 3.2. Secondly, this Section describes and discusses the results of simulating the intruder handling system by CPN Tools.

In Section 3.2 the CPN modelling guidelines have been categorized into three groups – knowledge, interaction and behaviour models – that guide the transformation of the corresponding types of conceptual agent-oriented models into a syntactically correct CPN model for simulation by CPN Tools.

```

▼MSC Setup
  ▼val msc = MSC.createMSC("Sequence Diagram");
  ▼val intruder = "Person/Intruder";
  ▼val security_manager = "SecurityAgent/SecurityManager";
  ▼val police = "PoliceAgent/Police";
  ▼val visitor = "Person/Visitor";
  ▼val owner = "Person/Owner";
  ▼val _ = MSC.addProcess(msc, intruder);
  ▼val _ = MSC.addProcess(msc, security_manager);
  ▼val _ = MSC.addProcess(msc, police);
  ▼val _ = MSC.addProcess(msc, visitor);
  ▼val _ = MSC.addProcess(msc, owner);
▼Built-in Colset
  ▶colset UNIT
  ▶colset INT
  ▶colset BOOL
  ▶colset INTINF
  ▶colset REAL
  ▼colset STRING = string;
  ▶colset TIME
▼User-Defined Colset
  ▼colset INTxINT=product INT*INT;
  ▼colset INTxSTRING= product INT*STRING;
  ▼colset INTxSTRINGxSTRING= product INT*STRING*STRING;
  ▼colset INTxINTxSTRINGxSTRING = product INT*INT*STRING*STRING;
▼Variables
  ▼var sid, pid,vid,hid,oid: INT;
  ▼var pn,d,vn,vd: STRING;

```

Figure 3-23: Representation of the agent knowledge model in CPN Tools.

Figure 3-23 shows the results of transforming the knowledge model of the intruder handling system represented in Figure 3-21 to the format of CPN used by CPN Tools by using two knowledge transformation guidelines – the transformation guidelines for transforming knowledge attributes and conceptual objects introduced in Section 3.2.1. The transformation guideline for transforming knowledge attributes maps simple attributes from a conceptual agent knowledge model to the format of CPN used by CPN Tools by using *Built-in Colset*, such as INT and STRING. The transformation guideline for transforming types of conceptual objects uses *User-Defined Colset*, such as INTxINT or INTxSTRING, to map conceptual object types from a conceptual agent knowledge model to the format of CPN used by CPN Tools. For example, the variable *sid* represents a simple attribute that identifies the subject, while INTxSTRING represents the conceptual object type *personal details* that consists of *pid* to store a person’s identity code and *pn* to store a person’s name. Table 3-2 describes knowledge for the intruder handling system represented in the form of the resulting CPN model, after application of the knowledge transformation guidelines.

Table 3-2: Description of the CPN variables of the intruder handling system.

| Sno. | Variable Name | Variable description |
|------|---------------|--|
| 1 | Sid | Identification code of the subject |
| 2 | pid | Identification code of known person |
| 3 | vid | Identification code of expected visitor |
| 4 | hid | Identification code of house (house address) |
| 5 | oid | Identification code of house owner |
| 6 | pn | Name of known person |
| 7 | d | Current date, randomly generated |
| 8 | vn | Name of the visitor |
| 9 | vd | Date of the visit |

Table 3-2 describes knowledge of the intruder-handling case study represented in the CPN Tools after application of knowledge transformation guidelines from Section 3.2.1.

In addition to applying the knowledge transformation guidelines from Section 3.2.1, it is also necessary to appropriately apply agent-oriented interaction and behaviour transformation guidelines from the respective Sections 3.2.2 and 3.2.3. Figure 3-28 shows agent-oriented interaction and behaviour modelling constructs identified in a combined agent interaction and behaviour model of the intruder handling system. Coloured rectangles have been used in Figure 3-24 to distinguish between the modelling constructs of the AOM4STS methodology as follows:

- *blue* represents the interaction modelling construct of sending a message identified in Section 3.2.2. For example, when a security agent sends the identity of an intruder and the address of the house that has been intruded to the police agent;
- *green* represents the interaction modelling construct of receiving a message identified in Section 3.2.2. For example, when the police agent receives from the security agent a message containing the identity of an intruder and the address of the house that has been intruded;
- *purple* represents the behaviour modelling construct of reactive behaviour identified in Section 3.2.3. For example, when a subject is detected, the security agent reacts to this event by searching for the identity of the detected subject in its knowledge base;
- *red* represents the behaviour modelling construct of parameter passing between activities identified in Section 3.2.3. For example, after the police agent has received information about the intruder, the police agent passes that information to another activity that adds the information about the suspect into the police database.



Figure 3-24: Transformation guidelines identified in the combined agent interaction and behaviour models of the intruder handling system.


```

▼ Functions
▼ fun move_subject(subjectID) =
  MSC.addEvent (msc, intruder, security_manager, "MOVE SUBJECT ("^INT.mkstr(subjectID)^")");
▼ fun inform_police(subjectID,houseID) =
  MSC.addEvent (msc, security_manager, police, " INFORM POLICE ("^INT.mkstr(subjectID)^" , "^INT.mkstr(houseID)^")");
▼ fun notice_subject(subjectID,houseID) =
  MSC.addInternalEvent (msc, security_manager, "NOTICE SUBJECT ("^INT.mkstr(subjectID)^" , "^INT.mkstr(houseID)^")");
▼ fun identify_subject(subjectID,subjectName) =
  MSC.addInternalEvent (msc, security_manager, "IDENTIFY SUBJECT ("^INT.mkstr(subjectID)^" , "^subjectName^")");
▼ fun detect_intruder(subjectID,houseID) =
  MSC.addInternalEvent (msc, security_manager, "DETECT INTRUDER ("^INT.mkstr(subjectID)^" , "^INT.mkstr(houseID)^")");
▼ fun get_current_date(currentDate) =
  MSC.addInternalEvent (msc, security_manager, "GET CURRENT DATE ("^currentDate^")");
▼ fun inform_visitor(houseID,visitorID,visitorName,visitingDate) =
  MSC.addEvent (msc, security_manager, visitor, "INFORM VISITOR ("^INT.mkstr(houseID)^" , "^INT.mkstr(visitorID)^" , "^visitorName^" , "^visitingDate^")");
▼ fun visitor_receive_msg(houseID,visitorID,visitorName,visitingDate) =
  MSC.addInternalEvent (msc, visitor, "RECEIVE MESSAGE ("^INT.mkstr(houseID)^" , "^INT.mkstr(visitorID)^" , "^visitorName^" , "^visitingDate^")");
▼ fun owner_receive_msg(ownerID,houseID) =
  MSC.addInternalEvent (msc, owner, "RECEIVE MESSAGE ("^INT.mkstr(ownerID)^" , "^INT.mkstr(houseID)^")");
▼ fun police_receive_msg(subjectID,houseID) =
  MSC.addInternalEvent (msc, police, "RECEIVE MESSAGE ("^INT.mkstr(subjectID)^" , "^INT.mkstr(houseID)^")");
▼ fun update_suspects_db(subjectID,houseID) =
  MSC.addInternalEvent (msc, police, "UPDATE SUSPECTS DB ("^INT.mkstr(subjectID)^" , "^INT.mkstr(houseID)^")");
▼ fun visitor_found(visitorID,visitorName,visitingDate) =
  MSC.addInternalEvent (msc, security_manager, "FOUND VISITOR ("^INT.mkstr(visitorID)^" , "^visitorName^" , "^visitingDate^")");
▼ fun no_visitor_found(houseID, currentDate) =
  MSC.addInternalEvent (msc, security_manager, "NO FOUND VISITOR ("^INT.mkstr(houseID)^" , "^currentDate^")");
▼ fun inform_owner(ownerID,houseID) =
  MSC.addEvent (msc, security_manager, owner, "INFORM OWNER ("^INT.mkstr(ownerID)^" , "^INT.mkstr(houseID)^")");

```

Figure 3-26: Functions that support to visualise the behaviour of intruder handling system.

The results of transforming the AOM modelling constructs identified in Figure 3-24 by different colours into the CPN model of the intruder handling system are represented in Figure 3-25 by using the same selection of colours. For example, Figure 3-24 contains four transformation guidelines of receiving a message represented by green colour. Accordingly, Figure 3-25 contains four green segments of the CPN model resulting from the application of the transformation guideline of receiving a message.

Each transition in the resulting CPN model has a label representing a function that passes one or more parameters to visualise the behaviour of the system by CPN Tools. For example, the transition *inform police* has the function *inform_police(sid,hid)* that helps to visualise by a Message Sequence Chart (MSC) [55] the interactions between the security agent and police agent and the sharing of the knowledge – subject identity and the address of the house that has been intruded – between the two agents. This way, MSC provides an attractive visual formalism that supports visualisation of system interactions and knowledge exchange before actual development of the system. Figure 3-30 provides the description of functions for each transition in the resulting CPN model.

We then use three different scenarios to validate the correctness of the CPN model of the intruder handling system in Figure 3-25 and discusses the validation results. Moreover, this Section presents visualisation of simulation results by CPN Tools capturing the knowledge, interactions and behaviours of agents by the resulting CPN model of the intruder handling system. The validation process entails three different scenarios. Each scenario uses the same CPN model and the same values for initial states represented as initial tokens in CPN. The only difference between these three scenarios emerges during the simulation process. Namely, the CPN Tools applies a non-deterministic algorithm to select values in the initial states and the suitable transitions for execution that enables CPN Tools to demonstrate different behaviours of the system during the simulation process [71]. Consequently, the three scenarios result in different values represented as final CPN tokens in the final states of CPN.

In each scenario, there are 10 CPN places that are represented in a table consisting of 10 rows, such as Table 3-3, Table 3-4 and Table 3-5. Row 1 represents the “*Subject*” place that stores the identity of a subject detected in the house. Row 2 represents the “*Security agent*” place that stores the address of the house where a subject has been detected. Row 3 represents the “*Personal details*” place that contains the identity numbers and names of the people known by the security agent, i.e., the internal knowledge by the security agent. This means that when the security agent captures the identity of the subject, it checks the subject’s identity against the database of personal details of family members and other people visiting the house for identifying the subject. Therefore, row 4 represents the “*Known subject*” place that stores the details of the subjects known by the security agent, while row 5 represents the “*Suspects*” place that stores the identity of the intruder and the address of the house where the intruder has been detected. The latter helps the police agent to identify the intruder. Row 6 represent the “*Dates*” place that helps to generate the current date during the simulation process, while row 7 represents the “*Visitors*” place that stores the schedule of the visitors expected to visit the house. After detecting an intruder, the security agent needs to find the current date and warn the visitors expected on that date about the detection of an intruder in the given house address. Therefore, row 8 represents the “*Visitors msg*” place that stores the messages sent to visitors. Moreover, the security agent needs to identify the appropriate owner of the house where the intrusion has

happened and inform him or her about the incident. Therefore, row 9 represents the “House owners” place that stores the identities of house owners and the addresses of their houses, while row 10 represents the “Owners msg” place that stores the messages sent to the owners.

Initially, each scenario contains two subjects with the identity numbers 10 and 11. In addition, each scenario involves security agents in two different houses at the addresses 100 and 200. The database of personal details includes information about two persons Albert and Beata with the respective identity numbers 11 and 12. For the simulation purpose, the current date can either be 10.01.2016 or 11.01.2016, while the house schedule contains two visitors – Andrew and Brenda – scheduled to visit the house on 11.01.2016 and 12.01.2016, respectively. Lastly, there are two house owners with the identity numbers 1000 and 2000 for the houses with the respective addresses 100 and 200. The following Sections describe the simulation processes for the three scenarios of the intruder handling system and discusses the simulation results.

3.3.3.1 Scenario 1

Scenario 1 of simulating the intruder handling system considers a situation where the subject with the identity number 11 passes around the house at the address 100. In this situation, the security agent of the house at the address 100 detects the subject with the identity number 11, then searches for the subject with the identity number 11 in its internal knowledge base of personal details that contains information about Albert and Beata with the identity numbers 11 and 12, respectively. Consequently, the security agent identifies the subject with the identity number 11 as Albert because the identity number 11 exists in its internal knowledge base. As a result, the security agent stores that knowledge in the database for detected known subjects, and the simulation for Scenario 1 ends.

Table 3-3: Summary of validation results for Scenario 1 of simulating the intruder handling system.

| Sno. | Place Name | Initial token(s) | Final token(s) |
|------|------------------|--|---|
| 1 | Subject | (10), (11) | (10) |
| 2 | Security agent | (100), (200) | (200) |
| 3 | Personal details | (11,"Albert"), (12,"Beata") | (11,"Albert"), (12,"Beata") |
| 4 | Known subject | — | (11,"Albert") |
| 5 | Suspects | — | — |
| 6 | Dates | ("10.01.2016"), ("11.01.2016") | ("10.01.2016"), ("11.01.2016") |
| 7 | Visitors | (1,"Andrew","11.01.2016"), (2,"Brenda","12.01.2016) | (1,"Andrew","11.01.2016), (2,"Brenda","12.01.2016) |
| 8 | Visitors msg | — | — |
| 9 | House owners | (1000,100), (2000,200) | (10,100), (20,200) |
| 10 | Owners msg | — | — |

In other words, this scenario considers a situation where the security agent detects a subject known by the intruder handling system. In such a situation, the security agent does not need to interact with the police, visitor or owner agents. Table 3-3 summarises

the results of simulating the CPN model for this scenario by CPN Tools. This table shows the initial values (before the simulation) and final values (after the simulation) for each place in Scenario 1. For example, initially the place “Subject” contains the identity numbers 10 and 11, while the security agent contains the house addresses 100 and 200. During the simulation process, the security agent of the house at the address 100 detects the subject with the identity 11. Therefore, the places “Subject” and “Security agent” remain with the subject identity number 10 and house address 200, respectively. Also, the place “Known subject” is empty before the simulation, while after the simulation it contains information about the detected subject, i.e., the identity number 11 and the name Albert. Figure 3-27 represents a screenshot from CPN Tools that complies with the simulation results presented in Table 3-3 and presents a visualisation of the knowledge sharing, interaction and behaviour of the agents participating in Scenario 1 of simulating the intruder handling system. Figure 3-27 clearly shows that the police, visitor and owner agents are not involved in Scenario 1. In summary, this is the case where the security agent detects a subject known by the system.

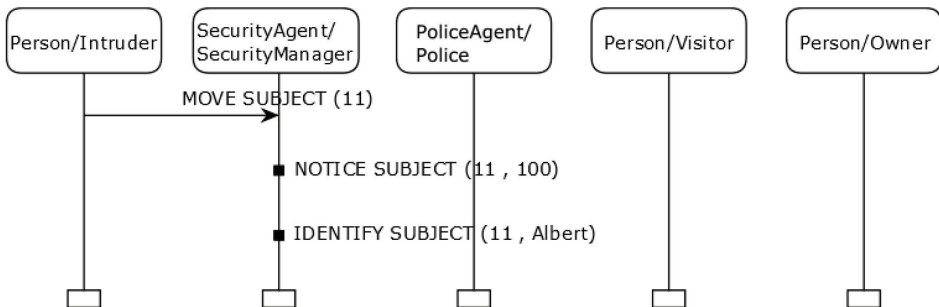


Figure 3-27: Visualisation of Scenario 1 of the intruder handling system by CPN Tools.

3.3.3.2 Scenario 2

Scenario 2 of simulating the intruder handling system considers a situation where the subject with the identity number 10 passes around the house at the address 100 on 10.01.2016. In this situation, the security agent of the house at the address 100 detects the subject with the identity number 10, and then searches for the subject with the identity number 10 in its internal knowledge base of personal details that contains information about Albert and Beata with the identity numbers 11 and 12, respectively. Consequently, the security agent categorises a detected subject as an intruder because the subject with the identity number 10 does not exist in its internal knowledge base. As a result, the security agent shares its new knowledge about the intruder with the police agent, who stores the identity of the intruder (10) and the address of the house that has been intruded (100) in its internal knowledge base of suspects for further processing. Following that, the security agent obtains the current date (10.01.2016) and starts searching for visitor(s) on that day in its internal knowledge base. However, in this scenario, the knowledge base does not contain any visitors scheduled to visit the house on 10.01.2016. Therefore, the security agent proceeds to identify the owner with the identity number 1000 for the house at the address 100 and inform him or her about the incident.

Table 3-4: Summary of validation results for the Scenario 2 of simulating the intruder handling system.

| Sno. | Place Name | Initial token(s) | Final token(s) |
|------|------------------|---|---|
| 1 | Subject | (10), (11) | (11) |
| 2 | Security agent | (100), (200) | (200) |
| 3 | Personal details | (11,"Albert"), (12,"Beata") | (11,"Albert"), (12,"beata") |
| 4 | Known subject | — | — |
| 5 | Suspects | — | (10,100) |
| 6 | Dates | ("10.01.2016"), ("11.01.2016") | ("11.01.2016") |
| 7 | Visitors | (1,"Andrew", "11.01.2016"), (2,"Brenda", "12.01.2016") | (1,"Andrew","11.01.2016"), (2,"Brenda","12.01.2016") |
| 8 | Visitors msg | — | — |
| 9 | House owners | (1000,100), (2000,200) | (1000,100), (2000,200) |
| 10 | Owners msg | — | (1000,100) |

Scenario 2 considers a situation where the security agent detects the subject who is not known by the system and where there are no visits to the house scheduled to take place on the day in question. In such a situation, the security agent does not need to interact with visitor agents. Table 3-4 summarises the simulation results of the CPN model for this scenario by CPN Tools. This table shows the initial values (before the simulation) and the final values (after the simulation) for each place in Scenario 2 of simulating the intruder handling system. For example, row 5 shows that the simulation categorises the subject with the identity number 10 as an intruder to the house at the address 100. Also, row 8 shows that visitors do not receive any warning messages and lastly row 10 shows that the security agent notifies the appropriate house owner about the incident. Figure 3-28 represents a screenshot from CPN Tools that complies with the simulation results presented in Table 3-4 and visualises the knowledge sharing, interactions, and behaviours of the agents in Scenario 2. Figure 3-28 clearly shows that visitors are not involved in the Scenario 2. In summary, this is the case where a security agent detects a subject who is not known by the system on the day when no visitor is scheduled to visit the house.

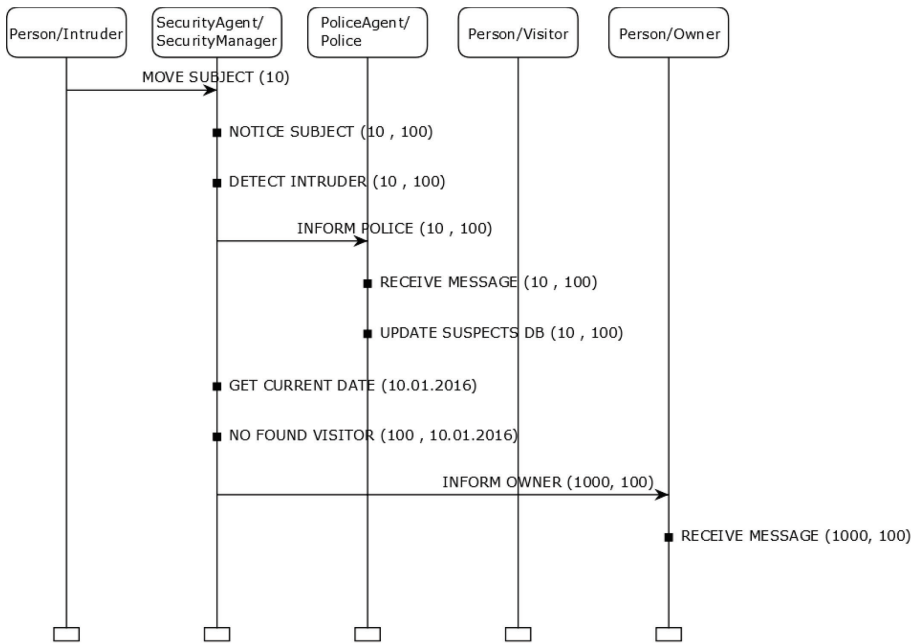


Figure 3-28: Visualisation of Scenario 2 of the intruder handling system by CPN Tools.

3.3.3.3 Scenario 3

Scenario 3 of simulating the intruder handling system considers a situation where the subject with the identity number 10 passes around the house at the address 100 on 11.01.2016. In this situation, the security agent of the house at the address 100 detects the subject with the identity number 10. Thereafter the security agent searches for the subject with the identity number 10 in its internal knowledge base of personal details that contains information about Albert and Beata with the identity numbers 11 and 12, respectively. Consequently, the security agent categorises a detected subject as an intruder because the subject with the identity number 10 does not exist in its internal knowledge base. As a result, the security agent shares the new knowledge about the intruder with the police agent, who stores the identity of the intruder (10) and the address of the house that has been intruded (100) in its internal knowledge base of suspects for further actions. Following that, the security agent obtains the current date (11.01.2016) and starts to search for visitor(s) scheduled to visit the house on that day in its internal knowledge base. The security agent finds Andrew. Hence, the security agent sends a message to Andrew and warns him about the incident. Lastly, the security agent proceeds to identify the owner with the identity number 1000 for the house at the address 100 and inform him or her about the incident.

Table 3-5: Summary of validation results for Scenario 3 of simulating the intruder handling system.

| Sno. | Place Name | Initial token(s) | Final token(s) |
|------|------------------|---|---|
| 1 | Subject | (10), (11) | (11) |
| 2 | Security agent | (100), (200) | (100) |
| 3 | Personal details | (11,"Albert"), (12,"Beata") | (11,"Albert"), (12,"Beata") |
| 4 | Known subject | — | — |
| 5 | Suspects | — | (10,200) |
| 6 | Dates | ("10.01.2016"), ("11.01.2016") | ("10,01.2016") |
| 7 | Visitors | (1,"Andrew", "11.01.2016"), (2,"Brenda", "12.01.2016") | (1,"Andrew","11.01.2016"), (2,"Brenda","12.01.2016") |
| 8 | Visitors msg | — | (200,1,"Andrew","11.01.2016") |
| 9 | House owners | (1000,100), (2000,200) | (1000,100), (2000,200) |
| 10 | Owners msg | — | (2000,200) |

This scenario considers a situation where the security agent detects the subject who is not known by the system on the day when there are visits scheduled to the house. In such a situation, the security agent needs to interact with all the other agents, i.e., the police, visitor and owner agents. Table 3-5 summarises the simulation results of the CPN model for this scenario by CPN Tools. Table 3-5 shows the initial values (before the simulation) and the final values (after the simulation) for each place in Scenario 3 of simulating the intruder handling system. For example, row 6 of Table 3-5 shows the value for the instantiated current date as 11.01.2016. Furthermore, the internal knowledge of the security agent represented in row 7 shows that Andrew with the identity number 1 is scheduled to visit the house on 11.01.2016. Therefore, row 8 represents a warning message sent to Andrew, who has the identity number 1 and is supposed to visit the house at the address 200 on 11.01.2016. Figure 3-29 represents a screenshot from CPN Tools that complies with the simulation results presented in Table 3-5 and visualises the knowledge sharing, interactions, and behaviours of the agents in Scenario 3. Figure 3-29 clearly shows that agents of all types participating in the intruder handling system are involved in Scenario 3. In summary, this is the case where the security agent detects a subject who is not known by the system on the day with scheduled visits to the house.

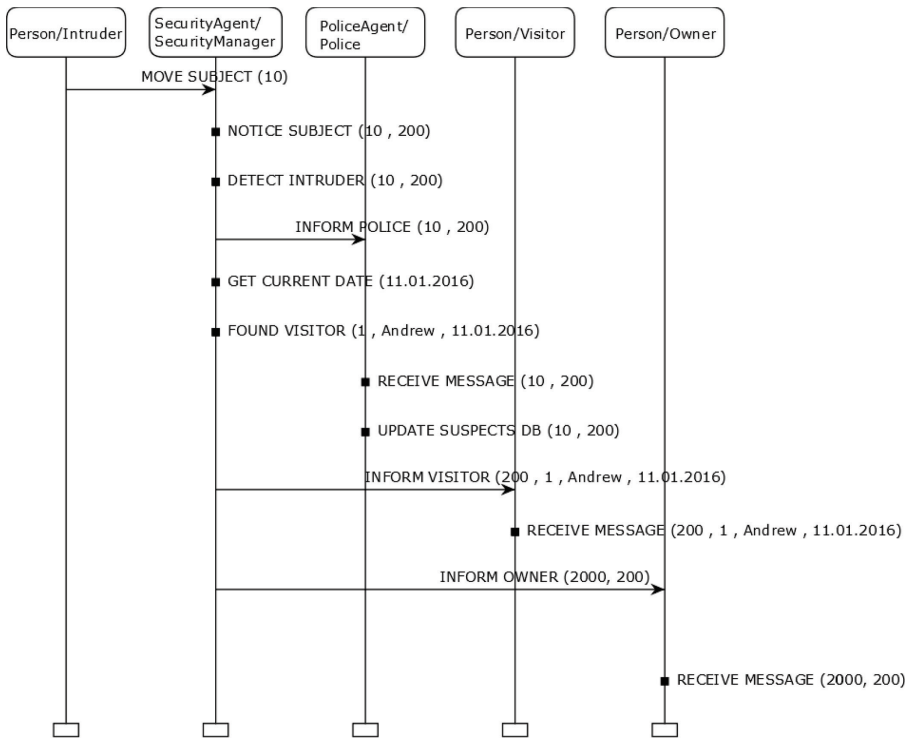


Figure 3-29: Visualisation of Scenario 3 of the intruder handling system by CPN Tools.

3.4 Empirical Evaluation of CPN Modelling Guidelines

This Section adapts the guidelines for experimentation in software engineering [154] to report the empirical study that evaluates the effectiveness of the proposed CPN modelling guidelines.

3.4.1 Experiment Scoping

This Section describes the scope of the empirical study.

3.4.1.1 Goal definition

The objective of this empirical study is to determine the effectiveness of CPN modelling guidelines put forward in Section 3.2 for representing design models of sociotechnical systems produced by the AOM4STS methodology for CPN Tools.

The CPN modelling guidelines are expected to represent design models of sociotechnical systems produced by AOM4STS methodology in CPN Tools without changing the scope of the system, i.e. the number of entities included in the system. Consequently, provide the support for visual simulation, validation and verification of sociotechnical systems through CPN Tools.

To determine the effectiveness of the modelling guidelines, it is important to compare the number of entities in the design models produced by the AOM4STS methodology and the number of entities in the CPN models produced through the proposed CPN modelling guidelines.

3.4.1.2 Object of study

The object of the study are the modelling guidelines that support the process of representing interaction, knowledge and behaviour design models of sociotechnical systems produced by the AOM4STS methodology for CPN Tools.

3.4.1.3 Purpose

The purpose of the experiment is to evaluate the effectiveness of the modelling guidelines for representing design models of sociotechnical systems produced by the AOM4STS methodology for CPN Tools. The experiment provides insight to what can be expected in terms of the number of entities of design models with and without using the proposed modelling constructs.

3.4.1.4 Perspective

The perspective taken is that by the researchers and designers willing to make use of the transformation guidelines for checking the soundness of the knowledge, interaction and behaviour design models of sociotechnical systems. This also includes people who would like to adopt the modelling guidelines in industry or conduct further research on the modelling guidelines.

3.4.1.5 Quality focus

The major effect studied in this experiment is the relation between the numbers of entities in the design models produced by the AOM4STS methodology and the corresponding number of entities in the resulting CPN models after the adoption of the modelling guidelines. This effect includes the number of agents, interactions, rules, and knowledge items. The latter are referred to as simply “knowledge” in this study.

3.4.1.6 Context

The experiment is run in the context of designing sociotechnical systems. The experiment was conducted within the course on agent oriented modelling for multi-agent systems given by the Department of Software Science at Tallinn University of Technology in Estonia. The study is from the course that was given in spring semester of the academic year 2015/2016.

3.4.1.7 Summary of Scoping

Evaluate the effectiveness of the modelling guidelines for representing design models of sociotechnical systems produced by the AOM4STS methodology for CPN Tools from the perspective of researchers and designers of sociotechnical systems in the context of the course on agent-oriented modelling for sociotechnical systems.

3.4.2 Experiment Planning

This Section describes the plan for conducting this experiment.

3.4.2.1 Context Selection

The context of the experiment is the course on agent-oriented modelling for sociotechnical systems given at the university, and hence the experiment is run off-line, i.e. not in an industrial software development environment. It is conducted by MSc and

PhD students. Moreover, this experiment is specific since it is focused on agent-oriented modelling for sociotechnical systems in an educational environment. Later, this Section discusses the threats to the validity of the experiment and elaborates the ability to generalise the research findings from this specific context. This experiment addresses a real problem – the effectiveness of modelling guidelines for representing design models of sociotechnical systems produced by the AOM4STS methodology for CPN Tools.

The usage of the course on agent-oriented modelling for sociotechnical systems as an experimental context provides other researchers with an opportunity to replicate the experiment. Furthermore, it means that there is no need to spend much effort in setting up the repeated experiment in terms of defining the experiment and creating the environment for running the experiment.

3.4.2.2 Hypothesis Formulation

An important aspect of any experiment is to know and state clearly the hypothesis to be evaluated by the experiment. The experiment involves the usage of modelling guidelines by the subjects to represent sociotechnical agents, interactions, rules and knowledge entities produced by the AOM approach as CPN models in CPN Tools. It is expected that the number of agents, interactions, rules and knowledge entities produced by the AOM approach is the same as the number of entities in the corresponding CPN models in CPN Tools. The following are the formal statements of the hypotheses followed by the definition of necessary measures to evaluate the hypotheses.

Null hypothesis, H_{10} : There is no difference between the number of sociotechnical agents produced by the AOM approach and the number agents in the corresponding CPN models in CPN Tools.

H_{10} : agents (AOM4STS) = agents (CPN)

The necessary metrics for this hypothesis is the number of agents in sociotechnical models produced by the AOM approach and the number of agents in the corresponding CPN models produced by CPN Tools.

Null hypothesis, H_{20} : There is no difference between the number of sociotechnical action events produced by the AOM approach and the number of interaction events in the corresponding CPN models in CPN Tools.

H_{20} : interactions (AOM4STS) = interactions (CPN)

The necessary metrics for this hypothesis is the number of communicative and non-communicative action events in sociotechnical interaction models produced by the AOM approach and the number of events in the corresponding CPN models produced by CPN Tools.

Null hypothesis, H_{30} : There is no difference between the number of rules in the sociotechnical behaviour models produced by the AOM approach and the number of rules in the corresponding CPN models in CPN Tools.

H_{30} : rules (AOM4STS) = rules (CPN)

The necessary metrics for this hypothesis is the number of rules in sociotechnical behaviour models produced by the AOM approach and the number of rules in the corresponding CPN models produced by CPN Tools.

Null hypothesis, H_{40} : There is no difference between the number knowledge entities in the sociotechnical knowledge models produced by the AOM approach and the number of knowledge items in the corresponding CPN models in CPN Tools.

H_{40} : knowledge (AOM4STS) = knowledge (CPN)

The necessary metrics for this hypothesis is the number of knowledge entities in sociotechnical knowledge models produced by the AOM approach and the number of knowledge items in the corresponding CPN models produced by CPN Tools.

3.4.2.3 Variable Selection

The independent variables are the modelling guidelines presented in Section 3.3. The dependent variables are the sociotechnical models¹ produced by the AOM4STS methodology and the resulting CPN models² in CPN Tools.

3.4.2.4 Selection of Subjects

The subjects were chosen based on convenience and interest, but not as a random sample in the sense that the subjects were students registered for the elective course on agent-oriented modelling for sociotechnical systems offered to MSc and PhD students of the School of Information Technology of Tallinn University of Technology.

3.4.2.5 Experiment Design

The case studies were not assigned randomly to the subjects but the subjects chose their own case studies. All the subjects were provided with the same knowledge on the AOM4STS methodology and CPN Tools. Also, the subjects were not selected randomly. They were the students, who had optionally registered for the elective course.

Furthermore, it would have been preferable to have a balanced dataset, but the experimental study was based on a course for which the subjects had registered. Therefore, it was impossible to influence the backgrounds of the subjects and this way balance the dataset.

The research design was two treatments per each factor. The factors are the case studies. The first treatment is the AOM4STS methodology without CPN modelling guidelines, which is in this study simply referred to as AOM4STS. The second treatment is the AOM4STS methodology with CPN modelling guidelines, which is in this study simply referred to as CPN. Each subject used both treatments on the same case study. Hence, a paired test is suitable. In this study, the paired t-test [75] was used.

The definitions, hypotheses and metrics of the second, third and fourth evaluations follow the same research design as for the first evaluation. Therefore, the paired t-test is used for all the three evaluations.

3.4.2.6 Instrumentation

Before conducting the experiment, each subject was provided with a set of CPN modelling guidelines for modelling interactions, knowledge and behaviour of the system. This data provided input for representing sociotechnical models in CPN by CPN Tools, and hence the modelling guidelines were independent variables in the experiment. Moreover, subjects made their own choices of case studies of designing sociotechnical systems of different sizes, but according to the provided instructions. These case studies were the objects of the experiment.

¹ <https://goo.gl/vxQcYU>

² <https://goo.gl/Z4jhbh>

3.4.2.7 Validity Evaluation

In this experiment, there are two major threats to the internal validity [154]. The first threat is that the modelling of case studies was conducted in groups of three to five students that were assigned based on convenience but not considering computing experience. However, all the subjects were novices in agent-oriented modelling of sociotechnical systems and CPN Tools. The second major threat to the internal validity is that while the duration of the modelling experiment was 16 weeks, from February to May of 2016, some of the subjects missed some weekly sessions. However, in each week there were at least two subjects from each student group present.

Concerning the external validity [154], it is highly probable that similar results will be obtained when running this experiment in a similar way with other subjects because the subjects of this experiment decided to register for the course of agent-oriented modelling of sociotechnical systems based on their interest in advanced software engineering and convenience. In addition, the results of the analysis can probably be generalised for representing design models of other agent-oriented software engineering methodologies in CPN by CPN Tools.

The major threat with respect to the conclusion validity [154] is the quality of the data collected during the course on agent-oriented modelling for sociotechnical systems. The students were expected to provide comprehensive reports that contain large amounts of models and description of the models as an essential part of the course. This involves a risk that the data is faked or simply not correct due to human mistakes. The data inconsistencies among the models of sociotechnical systems are, however, not believed to have a significant impact on the usage of the modelling constructs. Hence, the conclusion validity is not considered to be critical.

The construct validity [154] includes two major threats. The first threat to the construct validity is that the used metrics may not be appropriate ones for evaluating the effectiveness of the modelling guidelines. For example, is “the comparison between the number of entities produced by the AOM4STS methodology and the number of the resulting entities in CPN for CPN Tools” an appropriate metric for evaluating the effectiveness of the modelling guidelines? The second threat to the construct validity is that the experiment was conducted as a part of the course, where the students were graded. This implies that the students may bias their data, as they believe that it will give them better grades. However, in the beginning of the course it was emphasised that the grade did not depend on the actual data. The grade was instead based on the completeness of the requirements, proper delivery, and the understanding of the topics expressed in the reports that were handed in by students at the end of the course.

3.4.3 Experiment Operation

3.4.3.1 Preparation

The subjects of this experiment were not aware of what aspects were going to be evaluated. They were only told that the researchers wanted to study the outcome of the course on agent-oriented modelling of sociotechnical systems with respect to the usage of the CPN modelling guidelines. They were, however, not aware of the actual hypotheses to be evaluated. The subjects, from their point of view, did not primarily participate in an experiment but were just taking a course. All students were guaranteed anonymity. The materials of the experiment were prepared in advance. The course itself

was based on the textbook [135] about agent-oriented modelling for sociotechnical systems and the information provided on the course website¹.

3.4.3.2 Execution

The experiment was executed over a period of 15 weeks, during which subjects participated in the modelling workshops, giving presentations and writing the reports. The data for the experiment was primarily collected from the reports submitted by students at the end of the experiment and course.

As was stated before, the experiment was run within the course on agent-oriented modelling of sociotechnical systems and conducted in the university environment. The design of the experiment was in line with the course objectives and therefore did not affect the study plan.

3.4.3.3 Data Validation

Data was collected from 35 students, who formed 9 groups. Each group consisted of two to five students and focused on one case study. After the course, the reports were collected for analysis and interpretation. Data from one group consisting of two students was removed and regarded as invalid. This is because the subjects did not follow the experiment guidelines and decided to create two CPN models representing two major sub-goals of their case study to achieve full SSA for each CPN model. However, the need of full SSA was not mentioned as the requirement for this experiment.

This means that 1 group of the 9 groups was removed, hence leaving 8 groups for statistical analysis and interpretation of the results.

3.4.4 Experiment Results and Interpretation

This Section presents the results of analysing the collected empirical data and interprets the obtained results.

3.4.4.1 Descriptive Statistics

As the first step in analysing the data, descriptive statistics was used to visualize the data collected. We will now present the descriptive statistics about the numbers and types of agents and action events that were modelled in different case studies.

Numbers of agent types in case studies. In this experiment, subjects selected their own case studies of designing sociotechnical systems consisting of human agents and software agents. Figure 3-30 shows the frequency distribution of agent types in the selected case studies of designing sociotechnical systems.

¹ http://maurus.ttu.ee/sts/?page_id=2230

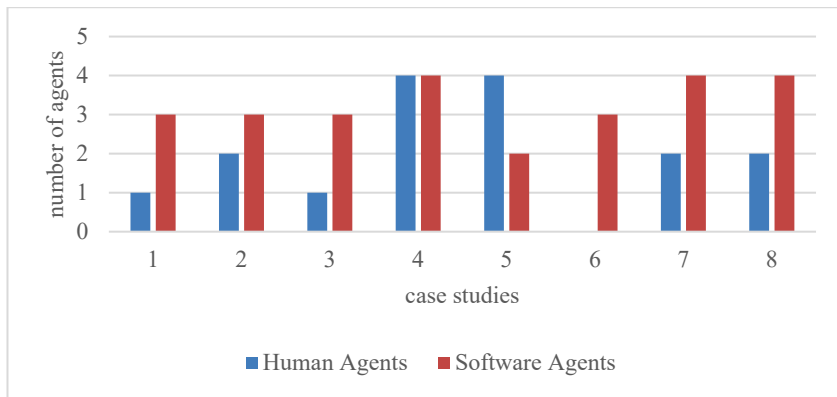


Figure 3-30: Frequency distribution of agent types in the selected case studies.

The results show that the number of software agent types in all case studies is higher than or close to the number of human agent types with an exception of case study 5 that deals with the collection and distribution of unused and leftover food to the people in need of nutritious food. It is also noticeable that case study 6 includes only software agent types without any human agent type. This is because case study 6 is about smart loans where banks give loans to their clients without any human involvement.

Numbers of action event types in case studies. Action events can be communicative (message) or non-communicative (physical). The former is performed by a software agent while the latter is performed by a human agent or robot or some device complying with the definition of agent. However, the case studies used in this experiment did not involve any robots or intelligent devices. Figure 3-31 shows the frequency distribution of action event types in the case studies.

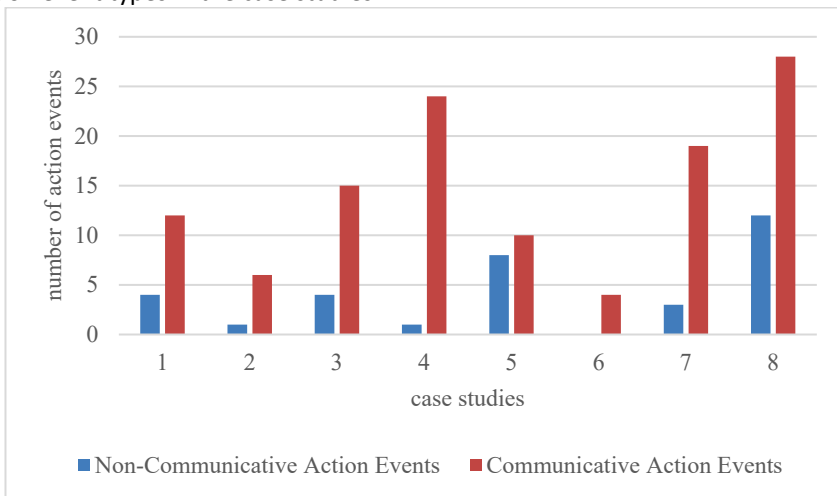


Figure 3-31: Frequency distribution of action event types in the selected case studies.

The results show that the number of communicative action event types (message types) is higher than the number of non-communicative action event types in each case study – including case study 5, which has more human agent types than software agent types. Therefore, the higher number of human agent types compared to the number of

software agent types does not necessarily result in a higher number of non-communicative action event types compared to the number of communicative action event types.

Figure 3-32 consists of the scatter plot charts (a) to (d) that show different types of relations that exists between the number of entities modelled by the AOM4STS methodology and the number of entities modelled in CPN. All the scatter plots show positive associations with an exception of scatter plot (b). This kind of positive association simply means that smaller values of the numbers of AOM4STS entities are generally associated with smaller values of the numbers of CPN entities for the same case study, and vice versa. All the scatter plots represented in Figure 3-32 show a linear pattern with an exception of the scatter plot (b). This linear pattern simply means that most of the markers in a scatter plot fall on or near a straight line. Therefore, the scatter plot (b) does not show any relationship pattern. Moreover, each scatter plot contains a straight line with y-intercept = 0 and slope = 1. Therefore, the marker in the scatter plot that falls on the straight line means that the number of AOM4STS entities is close to the number of CPN entities. The shorter the distance of the marker from the straight line is, the stronger is the association. Therefore, the scatter plot (c) shows a very strong association, while the scatter plots (a) and (d) show strong associations and the scatter plot (b) shows a very weak association.

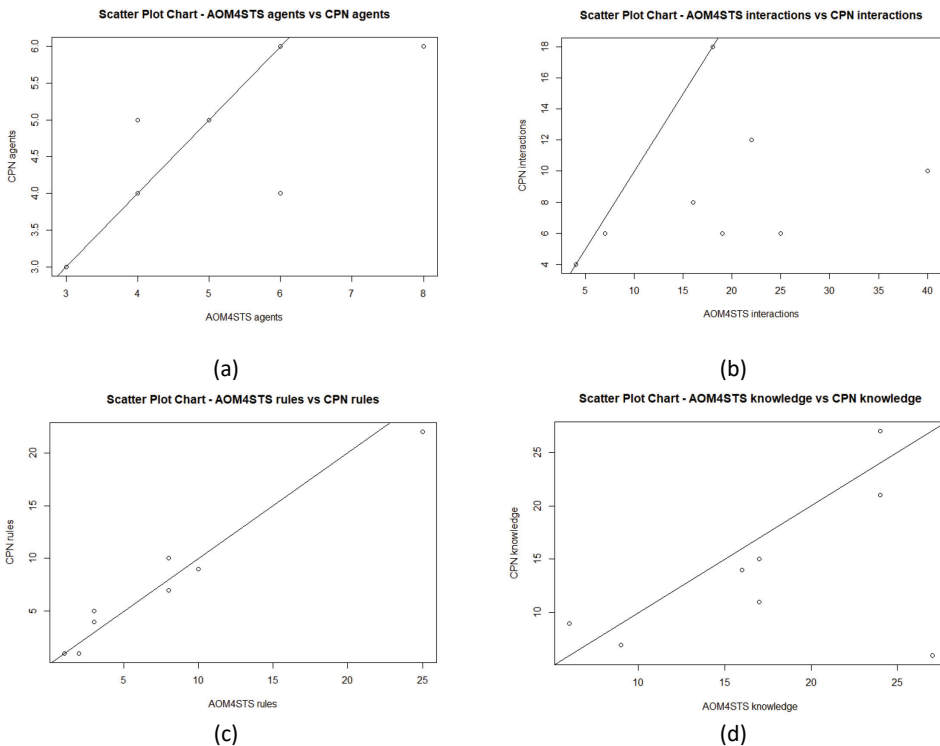


Figure 3-32: Scatter plot charts comparing the numbers of AOM4STS and CPN entities.

In summary, Figure 3-32 shows the following associations between AOM4STS entities and CPN entities. The scatter plot (a) shows a strong positive linear association between

agent types represented by AOM4STS and agent types represented in CPN. The scatter plot (b) shows a weak omnidirectional (i.e., no direction) association without any relationship pattern between interaction types represented by AOM4STS and interaction types represented in CPN. The scatter plot (c) shows a very strong positive linear association between rules represented by AOM4STS and rules represented in CPN. Lastly, the scatter plot (d) shows a strong positive linear association between knowledge entities represented by AOM4STS and knowledge items represented in CPN.

The descriptive statistics presented above has provided a better insight into the collected data, both in terms of what can be expected from the hypothesis testing and to potential problems that can be caused by weak associations.

3.4.4.2 Hypothesis Testing:

Each hypothesis formulated in Section 3.5.2 is evaluated by using a paired t-test. The data analysed during this experiment study can be found at the address provided in the footnote¹ of this page. The summary of the results from the paired and two-tailed t-tests is shown in Table 3-6.

Table 3-6: Results from the paired and two-tailed t-tests.

| Factors | Mean difference | Degree of Freedom (DF) | t-value | p-value |
|--------------|-----------------|------------------------|---------|---------|
| Agents | 0,375 | 7 | 1,0000 | 0,3506 |
| Interactions | 10,125 | 7 | 2,7219 | 0,0297 |
| Rules | 0,125 | 7 | 0,2047 | 0,8436 |
| Knowledge | 3,95 | 7 | 1,3970 | 0,2051 |

The following conclusions can be made based on the results in Table 3-6 :

For H1, the results fail to reject H_{10} . Therefore, there is no significant difference between the number of agent types modelled by AOM4STS and the number of agent types modelled by CPN. The p-value is relatively low meaning that the results are not highly significant.

For H2, the results reject H_{20} . Therefore, there is a significant difference between the number of interaction types modelled by AOM4STS and the number of interaction types modelled by CPN. The p-value is relatively low meaning that the results are not highly significant.

For H3, the results fail to reject H_{30} . Therefore, there is no significant difference between the number of rules modelled by AOM4STS and the number of rules modelled by CPN. The p-value is very high meaning that the results are highly significant.

For H4, the results fail to reject H_{40} . Therefore, there is no significant difference between the number of knowledge entities modelled by AOM4STS and the number of knowledge items modelled by CPN. The p-value is relatively low meaning that the results are not highly significant.

¹ goo.gl/dbp3QU

3.4.4.3 Additional Results

This Section provides additional results¹ that demonstrate how CPN Tools extends the AOM4STS methodology through visual simulation, scenario-based validation, and state space verification based on the experiments conducted with 8 different socio-technical systems.

(a) Visual Simulation

The visual simulation of the resulting CPN models captures activities (internal events) and interactions (external events) between various agents by using message-sequence charts.

Figure 3-33 represents a visual simulation scenario of a sociotechnical system (project ID 5) that aims to mitigate food waste, provide an assistance service, and overall improve the communities we live in by supporting provision of unused and leftover food to those in need of nutritious food. The visual simulation of this project by a message-sequence chart shows the knowledge exchange through interactions such as “create: food order (kesklinn)” and “deliver: packaged food (kesklinn, 10101)” between six different agents, where “kesklinn” stands for the city centre. Furthermore, this message-sequence chart visualises the internal activities of the agents (internal events) such as “pack: left over food (rimi, 10101)” and “sort: food (10101)”, where “rimi” stands for a particular supermarket.

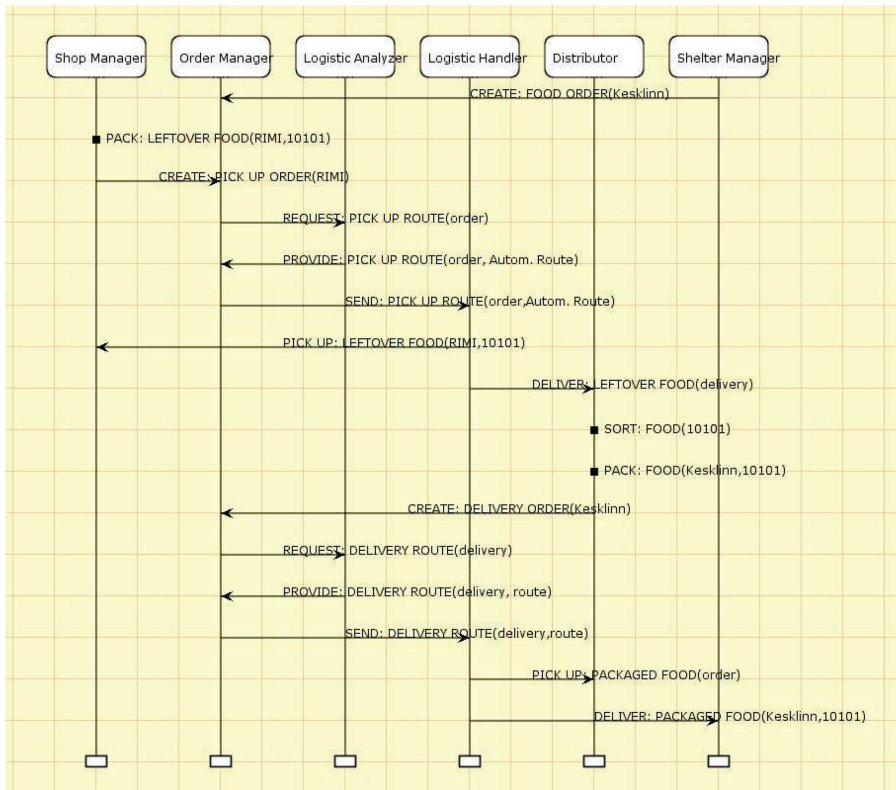


Figure 3-33: Visual simulation of the project with ID 5.

¹ <https://goo.gl/cPxjmr>

(b) Scenario-based validation

Table 3-7 summarises the results of scenario-based validation by showing the numbers of modelled scenarios, transitions, preconditions, postconditions, and rules of each resulting CPN model of a socio-technical system. The results reflect that each project modelled 2 to 4 scenarios for the validation purpose. These scenarios are represented by various transitions, preconditions and postconditions (places), and rules. 50% of the projects modelled 3 scenarios, 25% of the projects modelled 4 scenarios, and 25% of the projects modelled 2 scenarios.

Table 3-7: Results of scenario-based validation.

| ID | Scenarios | Transitions | Pre and Post Conditions | Rules |
|----|-----------|-------------|-------------------------|-------|
| P1 | 3 | 60 | 44 | 9 |
| P2 | 3 | 9 | 24 | 5 |
| P3 | 3 | 9 | 23 | 4 |
| P4 | 3 | 6 | 16 | 7 |
| P5 | 4 | 28 | 81 | 10 |
| P6 | 2 | 17 | 43 | 1 |
| P7 | 4 | 53 | 139 | 22 |
| P8 | 2 | 11 | 27 | 1 |

According to the results presented in Table 3-7, the projects with 4 scenarios contain the highest number of rules and the projects with 2 scenarios contain the lowest number of rules. These results indicate that business rules play an important role in scenario-based validation of the design of socio-technical systems by CPN Tools. These results are also similar to the results by Benner *et al.* [12] that emphasise the importance of business rules in creating scenarios for describing and clarifying the relevant properties of the problem domains, eliciting system requirements, evaluating design alternatives, and validating designs.

(c) State space verification

Table 3-8 shows the number of dead markings, dead transitions, and live transitions obtained after full or partial state space verification of each resulting CPN model of a socio-technical system. The state space verification shows that 25% of the projects result in a partial verification while the remaining 75% result in the full verification. The specific reason for each partial verification result is outside of the scope of this study. However, a partial space verification can be caused by either too much processing time during verification or by too big a generated state space to be stored in the available computer memory [69].

Furthermore, the related work in Section 2.3 shows that the presence of dead markings and live transitions does not imply a wrong design of the system but suggests further verification to cross check if the resulting dead markings and live transitions are correct or not. However, dead transitions correspond to parts of the model that can never be activated. Therefore, they can be removed from the model without changing the behaviour of the system [69].

Considering the projects included by Table 3-8 with full state space verification reports, only one project – P7 – contains dead transitions. Also, the same project P7 is

the only one that contains live transitions. According to the collected data¹, the project P7 contains the largest number of nodes, which is substantially higher than that of other projects. Therefore, the large size of the project P7 may be the reason why it is not possible to correctly remove dead transitions from its CPN model.

Table 3-8: Results of state space verification.

| ID | Status | Dead Markings | Dead Transitions | Live Transitions |
|----|---------|---------------|------------------|------------------|
| P1 | Partial | 162 | 11 | 0 |
| P2 | Full | 24 | 0 | 0 |
| P3 | Full | 1 | 0 | 0 |
| P4 | Full | 33 | 0 | 0 |
| P5 | Full | 2 | 0 | 0 |
| P6 | Full | 1 | 0 | 0 |
| P7 | Full | 0 | 38 | 10 |
| P8 | Partial | 301 | 6 | 0 |

3.5 Summary

Agents and their interactions, knowledge and behaviour are the key building blocks of sociotechnical systems. The AOM4STS methodology supports requirements engineering and design of sociotechnical systems but does not offer any software support for visual simulation, validation or verification of design properties. To fill in the identified gap, this Chapter proposes novel guidelines that support the mapping of design models of sociotechnical systems in CPN Tools for visual simulation, validation and verification. These guidelines are divided into the guidelines of CPN knowledge modelling in Section 3.2.1, CPN interaction modelling in Section 3.2.2 and CPN behaviour modelling in Section 3.2.3. Moreover, this Chapter describes the development of the intruder handling system with the help of CPN Tools to demonstrate the feasibility of the proposed CPN modelling guidelines.

Furthermore, this Chapter analysed the results of an empirical study that evaluates the effectiveness of the CPN modelling guidelines to support mapping of the design models of sociotechnical systems in CPN Tools. The CPN modelling guidelines were expected to represent design models of sociotechnical systems produced by AOM4STS methodology in CPN Tools without changing the scope of the sociotechnical system. Therefore, to determine the effectiveness of the CPN modelling guidelines, the empirical study compared the number of entities in the design models produced by the AOM4STS methodology and the number of entities in the CPN models produced through the proposed CPN modelling guidelines.

On the one hand, the analysis results show no significant difference between the number of agent types, rules and knowledge entities produced by the AOM4STS methodology and the number of agent types, rules and knowledge entities produced through the proposed CPN modelling guidelines. On the other hand, the analysis results show a significant difference between the number of interaction types produced by the AOM4STS methodology and the number of interaction types produced through the proposed CPN modelling guidelines. Therefore, these results conclude that the

¹ <https://goo.gl/cPxjmr>

proposed CPN modelling guidelines can effectively support modelling of agents and their knowledge and behaviours in CPN Tools for various applications of sociotechnical systems. However, the results suggest the need for further study on the effectiveness of the CPN interaction modelling guidelines.

Moreover, the additional results of the empirical study show the utility of CPN modelling guidelines in supporting visual simulation, scenario-based validation, and state space verification of the design models of sociotechnical systems through the CPN Tools. The results of visual simulations demonstrate the capabilities of Message Sequence Charts of CPN Tools in supporting visualisation of activities and interactions between various agents of different applications of sociotechnical systems. In addition, the scenario-based validation of the resulting CPN models of sociotechnical systems affirms that business rules are central building blocks in scenario-based validation of design properties of sociotechnical systems. Finally, the results of the state space verification demonstrates how the CPN formalisms in CPN Tools support quality improvement of design models produced by the AOM4STS methodology by identifying unwanted states and activities of the sociotechnical systems.

4 Prototyping by JADE Framework

4.1 Introduction

The design and development of various domain-specific applications can benefit from applying agent technology. The agent technology community introduces an entity (i.e. agent) that is autonomous, proactive and able to interact with other agents for task accomplishment [42]. This kind of software supports complex applications, such as ambient intelligence, e-business, peer-to-peer systems, bio-informatics, and negotiations [127] which demand the software to be robust, effective [80], co-operative in a broad range of environments, customizable to support user needs, secure, and capable of evolving over time to cope with changing requirements. Agent technology has been adopted in a range of areas. These areas include collaborative learning games [28], rural ICT [87], ubiquitous computing, e-commerce (business to business-B2B and business to client-B2C) [126], robotics [141], library management [121], e-learning, manufacturing, logistic [38], environment, and banking [48]. Other areas include construction [117], bioinformatics, accident management [5], power management [80], crisis management [138], sustainable software [29], mathematical modelling [148], and grid computing [119]. For example, agent technology can support a collaborative design environment among the participants of a construction application. It facilitates decision-making at various stages of a construction project like architectural design, engineering, and negotiations with contractors.

However, despite of the obvious benefits, agent technology has not been widely adopted by the software community [105]. The reasons for the setbacks are the diversity of agent-oriented software engineering methodologies and the lack of maturity in some of the methodologies [136]. The agent-oriented methodologies aid agent developers with the introduction of techniques, terminologies, notations, and guidelines for the development agent-based systems [140]. To date, about 30 agent-oriented methodologies have been introduced [78]. The reports show that some of the agent-oriented methodologies lack generality. They focus on specific systems and agent architectures [158]. In addition, some of the methodologies do not include a sufficient level of detail to be of real use [34]. The variety of agent-oriented methodologies and the fact that many of them have been directly or indirectly influenced by object-oriented methodologies can cause difficulties for industrial developers in selecting an appropriate agent-oriented methodology [136].

This Chapter presents a detailed case study to demonstrate the feasibility of the AOM4STS methodology for the development of sociotechnical systems. Although the AOM4STS methodology claims to be able to cope with complex system development [114,137], it is still not yet determined up to what extent this may be true. Therefore, it is vital to conduct a study to explore the feasibility of the AOM4STS methodology to promote agent technology to a wider community. The adoption of agent technology in the development of sociotechnical systems leads to several benefits. Most importantly, the agent technology supports decentralization, autonomy, fault tolerance and flexibility [42], and robustness and low coupling [51]. Hence, it is worth to study the adoption of the AOM4STS methodology for designing systems with these features.

Section 4.2 describes an extended version of the intruder handling case study that considers contextual details. Section 4.3 provides a comprehensive description of the

guidelines for developing design models of sociotechnical systems to be implemented by the JADE framework. Section 4.4 provides the analysis of the empirical evaluation of the proposed JADE guidelines. Section 4.5 summarises the chapter.

4.2 Extended Intruder Handling Case Study

This Section first addresses the requirements elicitation for the video surveillance system (VSS) through HOMER [153] – an agent-oriented requirements' elicitation method that we use for the AOM4STS methodology. Then, the conceptual domain modelling of the VSS is presented. This is followed by the elaboration of the conceptual models of VSS through platform-independent design. Finally, this Section demonstrates the transformation of the platform-independent design models into implementations in JADE. The main contribution of this Section is the validation of the AOM4STS methodology through a case study of the VSS development. Overall, this Section demonstrates the feasibility and applicability of the AOM4STS methodology for modelling complex distributed sociotechnical systems. In addition, detailed guidelines are provided for developers for engineering complex distributed sociotechnical systems in JADE.

A Human-Oriented Method for Eliciting Requirement – HOMER [153] – is used to elicit requirements for agent-based systems. HOMER is based on the organizational metaphor of “hiring new staff” to collect and identify the requirements for a given problem domain. The elicitation questions are shown in Table 4-1. In this case, software engineers elicit and reason on various considerations for recruiting new staff to solve a problem. From the answers gathered, the discovered requirements can be easily translated into the goal model, role model, organization model, and domain model based on the guidelines proposed by Cheah *et al.* [27]. In other words, the questions described in HOMER have a direct realization in the AOM4STS goal model and role models. Table 4-1 shows the elicitation answers for the intruder handling scenario. Several stakeholders are involved in working on the intruder handling scenario. They are the security manager, security personnel, family members (e.g. house owner and family members), visitors and neighbours. Each stakeholder has its own role and responsibilities. For example, the security manager has the responsibilities to observe the changes in the environment, alert the authorized personnel and a house owner about a detected intruder by sending a message, as well as by continuous monitoring and tracking of the intruder. The information presented in Table 4-1 is used to furnish the agent-oriented conceptual modelling process elaborated in the following Section.

Table 4-1: Elicitation questions and answers for intruder handling.

| From HOMER's question | Answer(s) |
|--|---|
| 1. If you were to hire more staff to handle your current problem, which positions would you need to fill? | <ol style="list-style-type: none"> 1. Security manager 2. Security personnel 3. Family members 4. Visitors 5. Neighbours |
| 2. For each position, we need to collect a "job description": (a) What is the purpose of this position? What aspects of the problem will this position solve or partially solve? | <ol style="list-style-type: none"> 1. Security manager <ul style="list-style-type: none"> • Observe the environment change. • Alert authorized personnel/person registered in the system of environment change. • Security manager will detect unauthorized person and send an alert message to the house owner, and other registered personnel in the system. • Send location of the threat to the security personnel for further action. 2. Security personnel <ul style="list-style-type: none"> • Respond immediately to home security threat. • Go to the location immediately after receiving alert message. 3. Family members, visitors, neighbours <ul style="list-style-type: none"> • Take immediate precaution and stay away from security threat location. |
| 2. For each position, we need to collect a "job description": (b) What tasks will commonly be required? | <ol style="list-style-type: none"> 1. Security manager <ul style="list-style-type: none"> • Send an alert message and location to security personnel, home owner, family members, visitors and neighbours. 2. Security personnel <ul style="list-style-type: none"> • Go to alert location and investigate the threat. 3. Family members, Visitors, Neighbours <ul style="list-style-type: none"> • Stay away from threat location. |
| 2(c). For each task above: i. What sub-tasks make up this task? | 1. Communication |
| 2(c). For each task above: ii. What constraints are there for this task? | Messages not delivered. |
| 2(d). Which system/people in the company does this person rely upon? | <ol style="list-style-type: none"> 1. Security manager – relies on intruder-handling system (services) 2. Security personnel – rely on security manager 3. Family members, visitors, neighbours – rely on security manager |

| | |
|---|---|
| 2(e). Who else in the company relies upon this person? | None |
| 2(f). What knowledge does this person require to perform his tasks correctly? | 1. Security manager – list of personnel and personal information in the company |
| 2(g). What resources, existing and new, are required by this person in fulfilling his position? | 1. Image and contact number. |
| 3. What code of behaviour must be observed by all of your employees? (a) Are there other codes of behaviour for certain positions, and what are they? | None |
| 4. What other rules and regulations must your company adhere to? | None |

4.3 Guidelines for Prototyping of Agent-Oriented Models on the JADE Framework

This Section provides a comprehensive description of the guidelines for developing design models of sociotechnical systems to be implemented by the JADE framework.

4.3.1 Conceptual models of the extended intruder handling system

This section presents the conceptual models of the extended intruder handling system described in Section 4.2.

4.3.1.1 Conceptual Domain Analysis Models

The conceptual domain analysis models reflect the stage of problem domain analysis and requirements engineering for an agent-oriented system. Requirements engineering is a common stage among various agent-oriented methodologies, which is used to elicit, represent, and analyse the requirements for developing an agent-oriented system and to model an agent-oriented system at a higher level of abstraction. The stage of requirements engineering is intended to present an overview of the system and determine its functionalities. Ignoring this stage can lead to misunderstanding the system to be designed [18]. Furthermore, the requirements engineering normally involves activities that provide the context in which the system is to be designed [18].

Figure 4-1 shows the goal model for intruder handling. The notion of goal provides an overview of the functionalities that should be achieved by an agent-oriented system. Goals can be divided into sub-goals. In addition to functional goals, there are quality goals that represent non-functional requirements for the system. Achieving a goal consumes resources and a goal is related to a particular role which indicates the actor or agent that is involved in achieving the goal [35]. A role is the capacity or position that is required for achieving goals. An agent is a software entity that is situated in an environment.

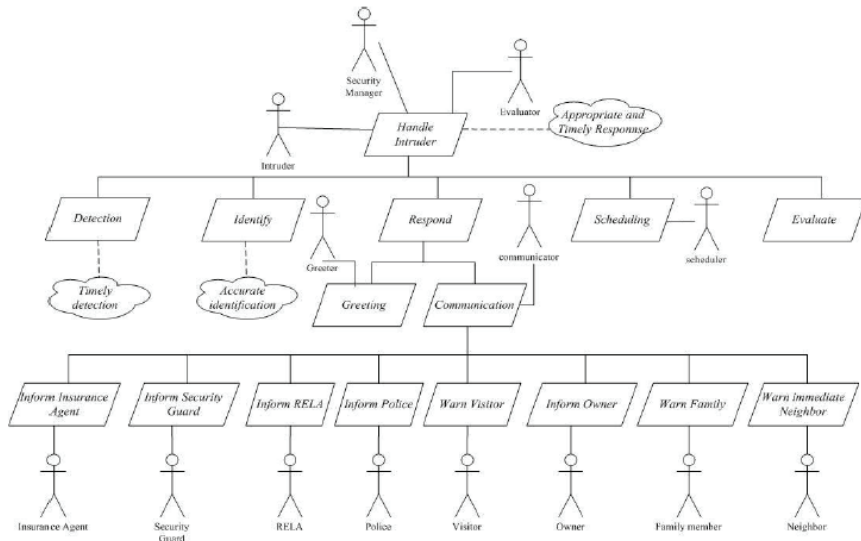


Figure 4-1: The goal model for the intruder handling scenario [128].

In Figure 4-1, quality goals are attached to the main goal “Handle intruder”, indicating that handling an intruder needs to be appropriate and timely. The three roles Security Manager, Intruder, and Evaluator are required to achieve the main goal. The main goal has been decomposed into the following five sub-goals: “Detection”, “Identify”, “Respond”, “Scheduling”, and “Evaluate”. Two quality goals – “Timely detection” and “Accurate identification” – are attached to the respective “Detection” and “Identify” sub-goals. The “Respond” sub-goal has, in turn, been expanded into the sub-goals “Greeting” and “Communication”. The “Communication” sub-goal is further divided into eight sub-goals. To accomplish these sub-goals, additional roles are involved, such as Police, Visitor, Security Guard, volunteer person (RELA), Insurance Agent, Family Member, Neighbour, and Owner.

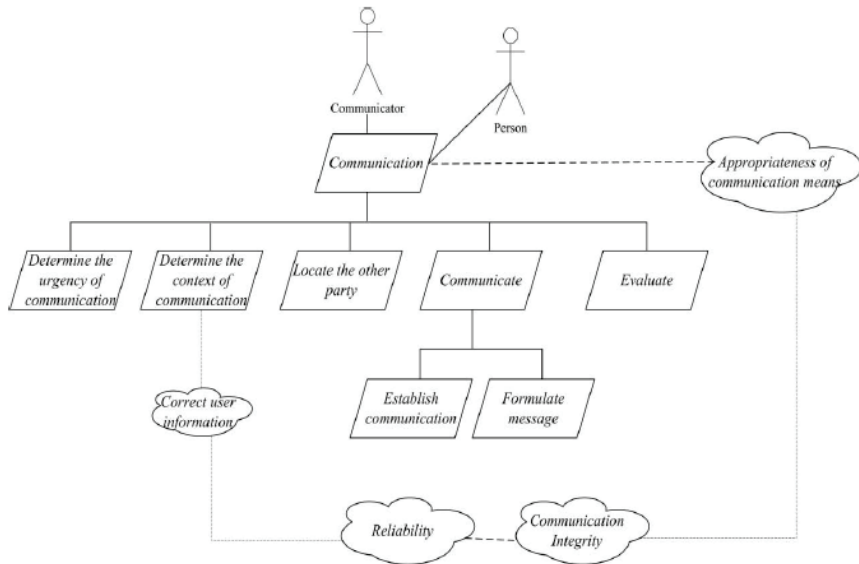


Figure 4-2: The goal model for communication [128].

In general, the Security Manager will first start the face recognition service to monitor the changes of the environment, which includes extra objects detected in the environment. These steps are covered by the sub-goals “Detection” and “Identify” modelled in Figure 4-1. Once the changes have been identified, the Security Manager will act as a communicator to establish communication with persons and software agents playing other roles as is shown in Figure 4-2 with the intention to alert them. The “Person” modelled in this system refers to the visitor, owner, family members, police, security guard, RELA, neighbour, insurance agent and any other relevant personnel.

Figure 4-3 shows the domain model that describes the relationships between different knowledge entities and/or roles involved in the intruder handling scenario. The PersonDescription domain entity caches the visual information captured about the person detected by the Security Manager. The Security Manager compares and matches the captured image against its database of people known by the system. The person is regarded as an intruder if the system fails to identify him or her. In this case, the person’s description will be forwarded to the Police, which may be able to identify the concrete suspect. To identify those who are authorized to be in the house such as plumbers or electricians, the Security Manager will consult the house schedule stored in the HouseSchedule domain entity. The HouseSchedule domain entity stores the start and end times of various activities that take place in the house, such as visits by friends and colleagues, family celebrations, and calls by service people. The schedule is created by the Owner.

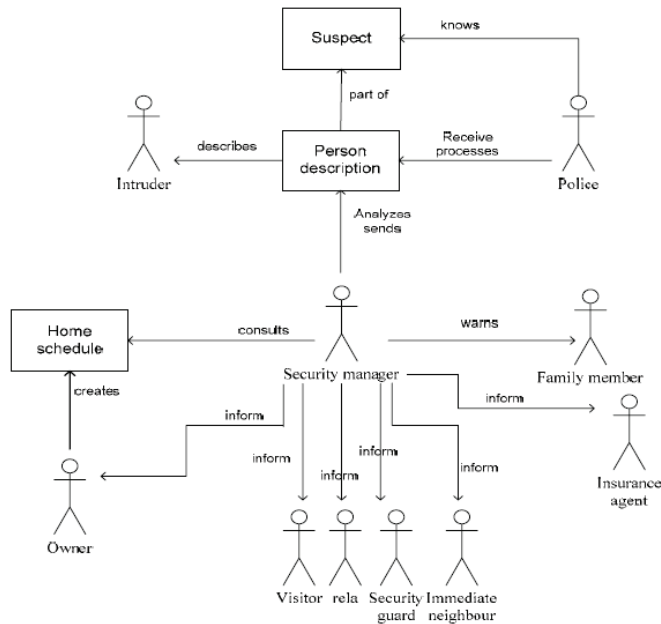


Figure 4-3: Domain model for intruder handling [128].

4.3.1.2 Platform-Independent Design Model

Platform-independent design models reflect the design of an agent-oriented system. Designing an agent-oriented system involves various design activities. Agents have knowledge about their environment and themselves. To reflect that, a knowledge model of an agent-oriented system need to be created, based on domain model. This involves activities to identify the knowledge entities, their attributes and predicates, and relations between the knowledge entities. In this chapter, we assume that there is a one-to-one mapping from roles to agent types. For example, the role Security Manager is mapped to the agent type "securityManagerAgent". Next, internal structure of agents needs to be designed. Design of an agent's internal structure involves activities to determine the arrangement of information flow, decision control, and inference steps for reasoning and action activation by an agent. Following that, interactions between agents are designed by detailing communication states of agents and presenting the syntax of messages and parameters exchanged between the agents.

The knowledge model for the intruder handling system is presented in the agent diagram shown in Figure 4-4. The knowledge model is constructed from the domain model depicted in Figure 4-3. The knowledge model represents the knowledge entities that are used by the agents. The PersonDescription object type is shared between agents of the SecurityAgent and PoliceAgent types. The HouseSchedule contains the attributes to store data, such as points of time, activities and reminders. It is owned privately by the SecurityManagerAgent. The SecurityManagerAgent will also execute the face recognition service, and initiate the communication service and house schedule service. To begin the intruder-handling process, the face recognition service is used to detect the changes of the environment, for instance, in the living room of a house. When a

change in the environment is detected with the presence of an unidentified object or person, the securityAgent is notified and the communication service is initiated.

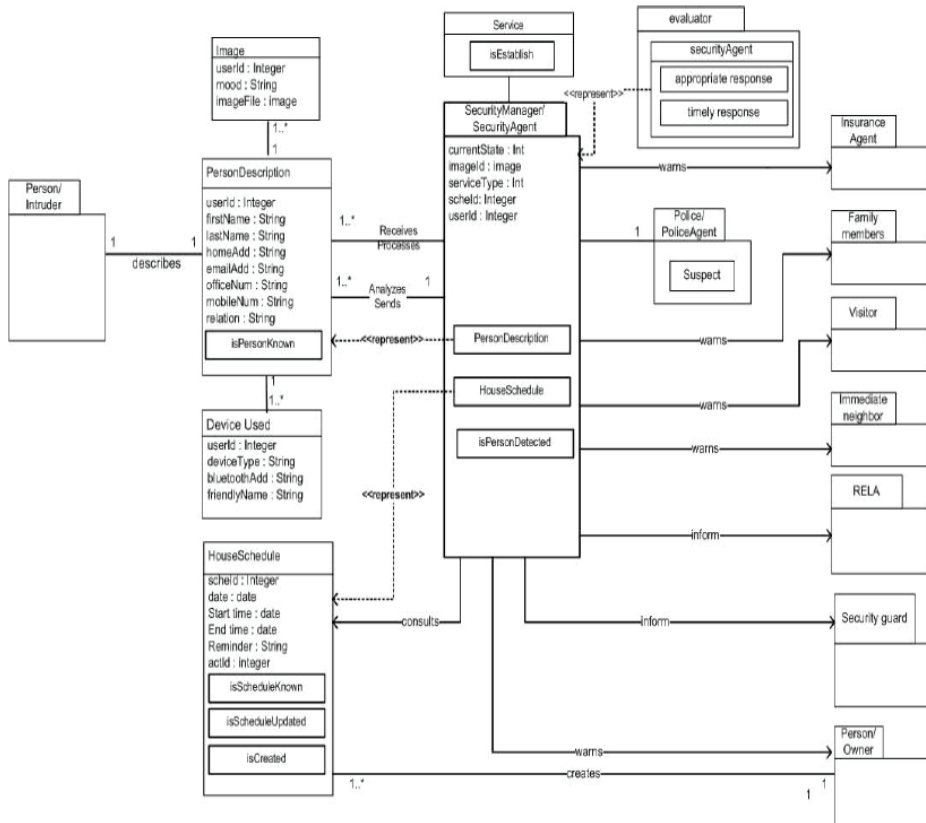


Figure 4-4: Knowledge model of intruder handling [128].

The sequence of interactions between various types of agents in intruder handling is modelled in Figure 4-5. The service of intruder handling is activated upon the detection of object changes or movements by the sensors. This will trigger the face recognition service to capture the image to determine the threat encountered. If the object is unidentifiable, the system will send an "intruder" event notification to the SecurityManagerAgent. Then, the SecurityManagerAgent will in turn send an "intruder detected" message to the policeAgent, RELA, security guard, and the house owner, and alert the visitors, neighbours, and family members to stay away from the possible threat. To ensure a quick response, the policeAgent will automatically assign any police within the vicinity to the scene to conduct further investigation. The RELA and security guard will also be sent to the location to assist in the investigation. During this incident, the policeAgent will commence communication with the house owner to update him or her about the situation.

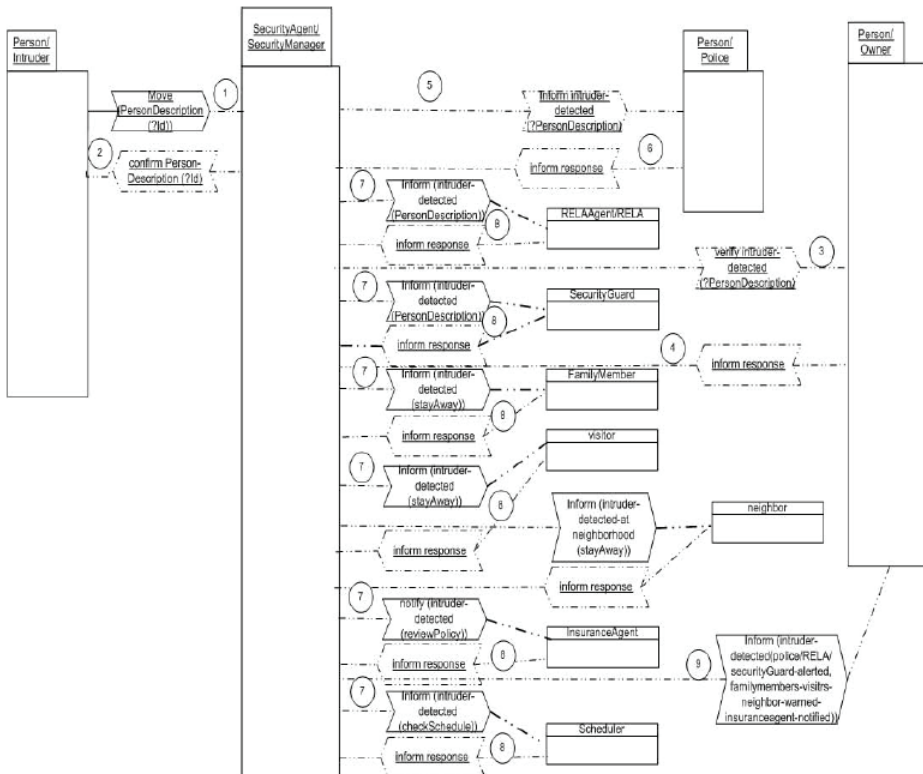


Figure 4-5: Interaction sequence diagram of intruder handling [128].

Figure 4-6 shows a simplified version of the behaviour model of intruder handling. The outermost activity is started by rule R1, which is triggered by an action event labelled as “move (?PersonDescription)”. This modelling construct represents in the form of an event a physical movement by an object or person detected by the security agent. Precision of the sensors is of no concern at this stage of the system engineering process. Rule R1 also creates an instance of the PersonDescription object type within the security agent to store important data on the person detected in the environment. A “Detect person” activity starts an “Identify intruder” sub-activity that triggers rule R2. This rule verifies the identity of the person through the “isKnown(PersonDescription)” function. If the person is unidentifiable, the activity “Respond” is executed. The “Respond” activity consists of a series of sub-activities that include alerting the relevant parties. The activity types modelled in the behaviour model should correspond to the goals represented by the goal model in Figure 4-6. This implies that each activity should achieve a goal modelled at the stage of conceptual domain analysis and requirements engineering. For example, the “Respond” activity should initiate the actions to achieve the “Respond” goal. The rules R3 and R4 are reaction rules depicted in Figure 4-6 that respond to the sub-activities by initiating communication through messages sent to the relevant recipients.

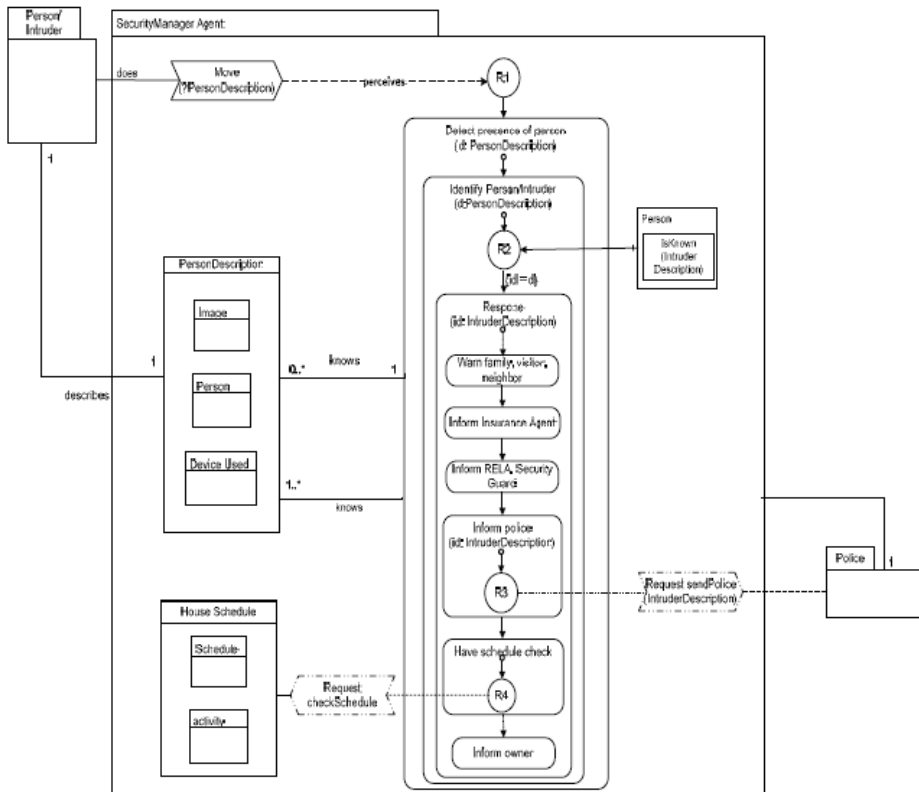


Figure 4-6: The behaviour model of intruder handling [128].

4.3.2 Prototyping of the extended intruder handling system

This Section demonstrates derivation of platform-specific models and programming constructs of the JADE framework based on the design models of the intruder-handling scenario presented in Section 4.3.1. These models constitute the lowest abstraction level of system design, moving towards implementation. This level provides design details that “specify how the system is to be implemented in a specific platform, architecture, and tool or programming language” [135]. It facilitates conversion from the design models to derive the skeleton programs or program templates that reflect the structure of the system. The design models are transformed into the programming constructs of the JADE framework based on the guidelines proposed by [27].

4.3.2.1 JADE-specific knowledge model

Figure 4-7 shows the snippets of the JADE-specific model of the PersonDescription knowledge entity type modelled in Figure 4-4. As mentioned before, the roles IntruderHandler, Police, Owner, Visitor, Family Member, Security Guard, RELA, Neighbour, and Insurance Agent are represented as the respective agent types of JADE. The shared knowledge entity type IntruderDescription and the private knowledge entity type Person are implemented as the corresponding Java object types. The predicate

isKnown is converted to the corresponding method attached to the Java object type Person.

```
import jade.content.*;
public class PersonDescription implements Concept {
    private int UserId;
    private String firstName;
    private String lastName;
    private String homeAdd;
    public int getUserId() {
        return UserId;
    }
    public String getfirstName() {
        return firstName;
    }
    public String getlastName() {
        return lastName;
    }
    //...others information model
}
```

Figure 4-7: JADE-specific knowledge model for the knowledge entity type PersonDescription [128].

The knowledge entity type PersonDescription is transformed into the corresponding JADE object type while the attributes of the knowledge entity type are declared as belonging to the String data type. The knowledge entity types Image and Device Used related to the PersonDescription knowledge entity type will be implemented in JADE as the corresponding object classes Image and DeviceUsed. Figure 4-8 shows a part of the coding of the face recognition service in JADE that captures image, analyses it, and informs the SecurityManagerAgent when an unknown person is detected.

```
//some opencv human detection code
void FrameGrabber_Standard(object sender, EventArgs e){
    //Get the current frame from capture device
    currentFrame = grabber.QueryFrame().Resize(320, 240, Emgu.CV.CvEnum.INTER.CV_INTER_CUBIC);
    //Convert it to Grayscale
    if (currentFrame != null){
        gray_frame = currentFrame.Convert<Gray, Byte>();
        //Face Detector
        MCvAvgComp[][] facesDetected = gray_frame.DetectHaarCascade(Face, 1.2, 10,
            Emgu.CV.CvEnum.HAAR_DETECTION_TYPE.DO_CANNY_PRUNING, new Size(50, 50));
        //Action for each element detected...
        //Draw the label for each face detected and recognized
        //add unknown face
        ActivateAgent();
        //...other processing code
    }
```

Figure 4-8: Coding of object detection by OpenCV¹ and sending of the corresponding intruder-handling event to the SecurityAgent [128].

¹ <https://opencv.org/>

4.3.2.2 JADE-specific interaction model

Figure 4-9 shows the JADE ontology of intruder handling for supporting interactions between the JADE agents of the intruder handling system. The ontology represents the knowledge that is shared among the agents through agent interactions by registering the types and properties of the corresponding knowledge entities. The ontology schema consists of an action schema and an information type schema to specify respectively the action types and knowledge entities involved while handling an intruder. The action schema includes the CreatePerson action type to create a new person object with detailed parameters corresponding to the attributes of the Person-Description knowledge entity type. The JADE-specific interaction model is a transformation of the interaction model and knowledge model that are presented in Figure 4-7 and Figure 4-8, respectively.

```
package ontologies;
import jade.content.onto.*;
import jade.content.schema.*;

public class IntruderOntology extends Ontology implements IntruderVocabulary {
    private static final long serialVersionUID = 1L;
    // ----->The name identifying this ontology
    public static final String ONTOLOGY_NAME = "Intruder-Ontology";
    // ----->The singleton instance of this ontology
    private static Ontology instance = new IntruderOntology();
    // -----> Method to access the singleton ontology object
    public static Ontology getInstance() { return instance; }
    // Private constructor
    private IntruderOntology() {
        super(ONTOLOGY_NAME, BasicOntology.getInstance());
        try {
            // ----- Add Concepts
            // Person
            ConceptSchema cs = new ConceptSchema(PERSON);
            add(cs, UserProfile.class);
            cs.add(PERSON_ID, (PrimitiveSchema) getSchema(BasicOntology.INTEGER),
                ObjectSchema.MANDATORY);
            cs.add(PERSON_FIRST_NAME, (PrimitiveSchema) getSchema(BasicOntology.STRING),
                ObjectSchema.MANDATORY);
            // House Schedule
            add(cs = new ConceptSchema(HOUSE_SCHEDULE), HouseSchedule.class);
            cs.add(HOUSE_SCHEDULE_ID, (PrimitiveSchema) getSchema(BasicOntology.INTEGER),
                ObjectSchema.MANDATORY);
            // ----- Add AgentActions
            catch (OntologyException oe) {
            }
        }
    }
}
```

Figure 4-9: Intruder Handling ontology for supporting JADE-specific interaction model [128].

4.3.2.3 JADE-specific behaviour model

Figure 4-10 shows a partial JADE-specific behaviour model of responding to intruder-handling. Once an intruder has been detected by the face recognition service, the SecurityManagerAgent will send a message labelled as "Intruder Alert" to all the relevant agents.

```
import jade.core.Agent;
import jade.core.behaviours.CyclicBehaviour;
import jade.domain.AMSService;
import jade.domain.FIPAAgentManagement.AMSAgentDescription;
import jade.domain.FIPAAgentManagement.SearchConstraints;
import jade.lang.acl.ACLMessage;
public class SecurityManagerAgent extends Agent{
private static final long serialVersionUID = 1L;
protected void setup() {
    AMSAgentDescription [] agents = null;
    try {
        SearchConstraints c = new SearchConstraints();
        c.setMaxResults( new Long(-1));
        agents = AMSService.search( this, new AMSAgentDescription (), c );
    }
    catch (Exception e) {
        System.out.println( "Problem searching AMS: " + e );
    }
    //once the intruder detection event is trigger, the securityAgent will send a messages to others
    ACLMessage msg = new ACLMessage(ACLMessage.INFORM);
    msg.setContent( "Intruder Alert" );

    for (int i=0; i<agents.length;i++)
        msg.addReceiver( agents[i].getName() );
    send(msg);
    addBehaviour(new CyclicBehaviour(this){
private static final long serialVersionUID = 1L;
        public void action() {
            ACLMessage msg= receive();
            if (msg!=null)
                System.out.println( "== " + " -> " + msg.getContent() + " from " + msg.getSender().getName() );
            msg=null;
        }
    });
}
```

Figure 4-10: JADE-specific behaviour model of responding to intruder-handling by the SecurityManagerAgent [128].

4.3.2.4 Prototype of the intruder handling system implemented in JADE

This Section presents the implementation of the intruder handling system on the multi-agent programming platform JADE. Figure 4-11 shows a part of the physical architecture and agents of the intruder handling system. All the nodes are connected wirelessly with static IPs for the face recognition service (192.168.1.50), intruder-handling system Main_Server (192.168.1.5), Police Main Server (192.168.1.4), and dynamic IPs for agents involved in the scenario. While some of the servers and agents can also be used in a real system, some other servers and agents, such as Police Main Server and policeAgent, are naturally just simulations.

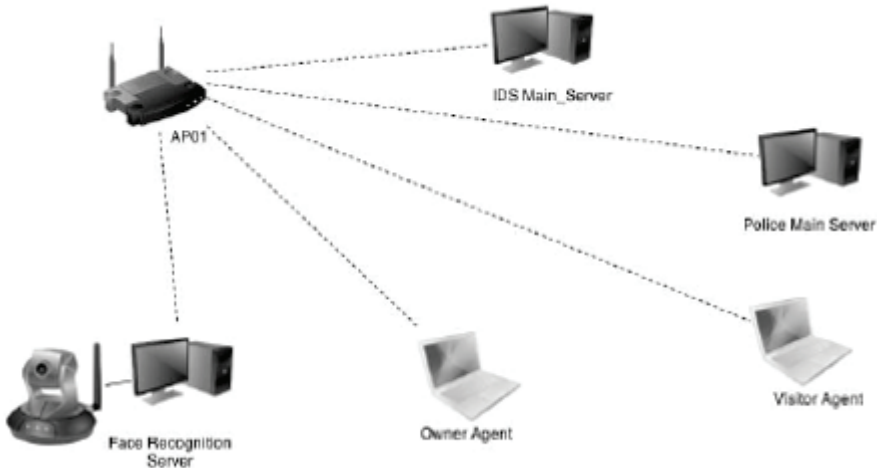


Figure 4-11: Implementation of the intruder handling system [128].

A webcam is connected to the face recognition system. An intruder-handling Main_Server acts as the context tier that performs all the work required for detecting changes in the environment and interpreting the data captured by the securityAgent to determine the context information. When a person enters the vicinity under scrutiny, the image recognizer (Web camera) detects the presence of a person and captures his or her face. The face recognition service extracts the data from the image and sends it to the securityAgent for contextual information processing.

```
Agent container Container-4@192.168.1.50 is ready.
== -> Intruder Alert from SecurityManager@192.168.1.5:1099/JADE
== -> noted from Visitor@192.168.1.5:1099/JADE
== -> Will stay away from the site. from Visitor@192.168.1.5:1099/JADE
== -> noted from family@192.168.1.5:1099/JADE
== -> noted from RELA_Officer_1@192.168.1.5:1099/JADE
== -> Will took appropriate action from family@192.168.1.5:1099/JADE
== -> Will investigate immediately from RELA_Officer_1@192.168.1.5:1099/JADE
```

Figure 4-12: Interactions between the securityManagerAgent, visitorAgent, familyAgent, and RelaOfficerAgent [128].

Once the person has been identified as an intruder, the securityAgent sends a message to the ownerAgent to further validate the identity of the person. When the ownerAgent is unable to recognize the person, the securityAgent will notify the policeManagerAgent and securityGuardAgent or RelaOfficerAgent about the intrusion.

Thereafter, the securityAgent contacts the agents of the scheduled visitors, family members, and neighbours to warn them to stay away. Meanwhile, the policeManagerAgent assigns the nearest police officer to the scene by comparing the distance between the scene and nearby police officers. Figure 4-12 represents the interactions between the securityManagerAgent, visitorAgent, familyAgent, and RelaOfficerAgent during the intruder handling. Meanwhile, Figure 4-13 presents a sample of interactions between the policeManagerAgent and respective policeAgent.

```
Agent container Container-3@192.168.1.50 is ready.  
Trying to assign police officer to MJC  
Found the following police agents:  
Police1@192.168.1.4:1099/JADE  
Police2@192.168.1.4:1099/JADE  
Intruder alert case at MJC was assign to Police1@192.168.1.4:1099/JADE  
Distance from location : 12  
Intruder alert from Police_HQ@192.168.1.4:1099/JADE terminating.
```

Figure 4-13: Interactions of the policeManagerAgent with the respective policeAgent [128].

4.4 Empirical Evaluation of the Guidelines for Prototyping of Agent-Oriented Models on the JADE Framework

This Section adopts the principles of experimentation in software engineering [154] to report the empirical study that evaluates the effectiveness of the guidelines proposed in Section 4.3 for development of agent-oriented models on the JADE Framework. From here on, we will refer to these guidelines as “JADE guidelines”.

4.4.1 Experiment Scoping

This Section describes the scope of the empirical study.

4.4.1.1 Goal definition

The objective of this empirical study is to determine the effectiveness of the JADE guidelines proposed in Section 4.3 for the development on the JADE Framework of design models of sociotechnical systems produced by the AOM4STS methodology.

For determining the effectiveness of the JADE guidelines proposed in Section 4.3, it is important to experiment and compare the results of developing design models of sociotechnical systems on the JADE framework by using the JADE guidelines against the results of developing design models of sociotechnical systems on the JADE framework without using the JADE guidelines.

4.4.1.2 Object of study

The object of the study are the JADE guidelines put forward in Section 4.3 that assist the development on the JADE framework of interaction, knowledge and behaviour design models of sociotechnical systems produced by the AOM4STS methodology.

4.4.1.3 Perspective

The perspective taken is that by the researchers and designers willing to make use of the JADE guidelines for developing sociotechnical systems from knowledge, interaction and behaviour design models produced by the AOM4STS methodology. This also includes people who would like to adopt the JADE guidelines in industry or conduct further research on the JADE guidelines.

4.4.1.4 Quality focus

The first effect studied in this experiment is the relation between the numbers of design features in JADE prototypes produced by using the JADE guidelines against the number of design features in JADE prototypes produced without using the JADE guidelines. These design features include the numbers of *types* of agents, interactions, conceptual objects, and behaviours.

The second effect studied in this experiment is the study of the development of JADE ontology in prototypes produced without using the JADE guidelines against the development of the JADE ontology in prototypes produced by using the JADE guidelines.

4.4.1.5 Context

The experiment was run in the context of agent-oriented development for sociotechnical systems. The experiment was conducted within the course of agent oriented modelling and multi-agent systems given at the Department of Software Science of Tallinn University of Technology in Estonia.

4.4.1.6 Summary of Scoping

Analyse the outcome of developing sociotechnical systems on the JADE Framework for evaluation with respect of using the JADE guidelines from the perspective of researchers and system designers in the context of agent-oriented development of sociotechnical systems.

4.4.2 Experiment Planning

This Section describes the plan for conducting this experiment.

4.4.2.1 Context Selection

The context of the experiment is the course on agent-oriented modelling of sociotechnical systems given at the university. The participants in the experiment are MSc and PhD students. Moreover, this experiment is specific because it is focused on agent-oriented modelling for sociotechnical systems in an educational environment. Later, this Section discusses the threats to the validity of the experiment and elaborates the ability to generalise the research findings from this specific context. This experiment addresses a real problem – the effectiveness of the JADE guidelines for developing on the JADE framework design models of sociotechnical systems produced by the AOM4STS methodology.

The usage of the course on agent-oriented modelling of sociotechnical systems as an experimental context provides other researchers with an opportunity to replicate the experiment. Furthermore, it means that there is no need to spend much effort in setting up the repeated experiment in terms of defining the experiment and creating the environment for running the experiment.

4.4.2.2 Research Question

An important aspect of any experiment is to know and clearly state the research question to be answered by the experiment. This experiment aims to provide an answer to the following research question:

What is the effectiveness of the JADE guidelines in prototyping sociotechnical systems on the JADE framework based on their design models produced by the AOM4STS methodology?

Before giving the answer to the research question, it is necessary to compare the key development features in prototypes developed without using the JADE guidelines against the development features in prototypes developed with using the JADE guidelines. Therefore, for guiding data collection and analysis, the research question is elaborated into more detailed research sub-questions and themes as is shown in Table 4-2.

Table 4-2: Research Sub-Questions and Themes [89].

| Research Sub-Question | Theme |
|--|-------------------------|
| (a) How many agent types were developed? | multi-agent development |
| (b) How many interaction types were developed? | interaction development |
| (c) How many conceptual object types were developed? | knowledge development |
| (d) How many behaviour types developed? | behaviour development |
| (e) Was ontology correctly developed and used by the agents? | knowledge development |

4.4.2.3 Variable Selection

The independent variables are the JADE guidelines presented in Section 4.3. The dependent variables are the prototypes¹ developed without using the JADE guidelines and prototypes² developed with using the JADE guidelines.

4.4.2.4 Selection of Subjects

The subjects were chosen based on convenience and interest, but not as a random sample in the sense that the subjects were students registered in 2012 and 2015 for the elective course on agent-oriented modelling of sociotechnical systems offered to MSc and PhD students of the School of Information Technology of Tallinn University of Technology. In this study, the students of 2012 are referred to as group 1 and the students of 2015 are referred to as group 2.

4.4.2.5 Experiment Design

The case studies were not assigned randomly to the subjects but the subjects chose their own case studies. The subjects of Group 1 were instructed to conduct requirements and design modelling for the selected case studies. Then, they were informed to use the resources from <http://jade.tilab.com/> and all available materials from the Internet to develop JADE prototypes based on the created design models. The subjects of Group 2 were instructed to use the same resources as Group 1 and the JADE guidelines presented in Section 4.3.

Furthermore, it would have been preferable to have a balanced dataset, but the experimental study was based on a course for which the subjects had registered. Therefore, it was impossible to influence the backgrounds of the subjects and this way balance the dataset.

¹ <https://goo.gl/UA6bjj>

² <https://goo.gl/2nRFpM>

4.4.2.6 Validity Evaluation

In this experiment, there are two major threats to the internal validity [154]. The first threat is that the modelling of case studies was conducted in groups of two to four students that were assigned based on convenience. All the subjects were novices in agent-oriented modelling for sociotechnical systems and in the development on the JADE Framework. The second major threat to the internal validity is that as the duration of the modelling experiment was 16 weeks, some of the subjects missed some weekly sessions. However, each week there were at least two subjects present from each student group.

Concerning the external validity [154], there is a high probability that similar results will be obtained when running this experiment in a similar way with other subjects because the subjects of this experiment decided to register for the course of agent-oriented modelling of sociotechnical systems based on their convenience and interest in advanced software engineering.

The major threat with respect to the conclusion validity [154] is the quality of the data collected during the course on agent-oriented modelling of sociotechnical systems. The students were expected to provide comprehensive reports that contain large amounts of models and descriptions of the models as essential parts of the course. This involves a risk that the data is faked or simply not correct due to human mistakes. However, the data inconsistencies among the models of sociotechnical systems is not believed to have a significant impact on the usage of the JADE guidelines. Hence, the conclusion validity is not considered to be critical.

The major threat to construct validity [154] is that the experiment was conducted as a part of the course where the students were graded. This implies that the students may bias their data, as they believe that it will give them better grades. However, in the beginning of the course it was emphasised that the grade did not depend on the actual data. The grade was instead based on the completeness of the requirements, proper delivery, and the understanding of the topics expressed in the reports that were handed in by the students at the end of the course.

4.4.3 Experiment Operation

This Section explains the execution of the study.

4.4.3.1 Preparation

The subjects of this experiment were not aware of what aspects were going to be evaluated. They were only told that the researchers wanted to study the outcome of the course on agent-oriented modelling of sociotechnical systems with respect to the usage of the JADE Framework. They were, however, not aware of the actual research questions to be answered by the results of the experiment. The subjects, from their point of view, did not primarily participate in an experiment but were just taking a course. All students were guaranteed anonymity. The materials of the experiment were prepared in advance. The course itself was based on the textbook [135] about agent-oriented modelling of sociotechnical systems and the information about the course was provided on the course website¹.

¹ http://maurus.ttu.ee/sts/?page_id=36

4.4.3.2 Execution

The experiment was executed over a period of 16 weeks, during which subjects participated in the JADE workshops, giving presentations and writing the reports. The data for the experiment was primarily collected from the source code of the developed JADE prototypes and from the reports submitted by the students at the end of the study.

As was stated before, the experiment was run within the course on agent-oriented modelling of sociotechnical systems and was conducted in the university environment. The design of the experiment was in line with the course objectives and therefore did not affect the study plan.

4.4.3.3 Data Validation

Data was collected from 13 projects in Group 1 and 13 projects in Group 2. Each group consisted of two to five students and focused on one case study. After the development experiment, the source code of the developed prototypes and reports were collected for analysis and interpretation. Some students found more convenient to do the projects individually. Their data was removed and regarded as invalid. Some student groups decided to use other agent development platforms, such as Jason¹. Their data was also removed and regarded as invalid because that data was irrelevant for the given experiment. Some subjects did not follow the guidelines and created incomplete sets of design models for sociotechnical systems. Their data was also removed and regarded as invalid to avoid biased results.

After removal of the invalid projects, a total of 16 projects remained and were considered in this empirical study: 8 projects in Group 1 and 8 projects in Group 2.

4.4.4 Experiment Results and Interpretation

In this Section, we analyse the data collected from developing JADE prototypes by using mean, median, and mode. This analysis aims to compare what was achieved by the subjects in Group 1 and Group 2 independently of their choices of projects.

According to the results represented in Table 4-3, the median and mean for the numbers of types of agents, conceptual objects and behaviours implemented in the prototypes of Group 1 is nearly the same as those implemented in the prototypes of Group 2 with an exception of interactions. According to the collected data², project with the identifier 1 in Group 1 has a very large number of implemented interaction types (17) compared to the average number of implemented interaction types in Group 1 (8.875). Also, the same project identified by 1 has the highest number of agent types (12) compared to the average number of agent types implemented in Group 1 (3.875). Although these results do not say much about the usage of the JADE guidelines, they show that an increase in the number of agent types in the system results in the increase in the number of types of interaction between the agents.

¹ <http://jason.sourceforge.net/wp/>

² <https://goo.gl/MMMCXM>

Table 4-3: Median and Mean comparison of prototypes implemented in Group 1 and Group 2 [89].

| ENTITY TYPE | MEDIAN | | MEAN | |
|-------------------|---------|---------|---------|---------|
| | GROUP 1 | GROUP 2 | GROUP 1 | GROUP 2 |
| AGENT | 3 | 3,5 | 3,875 | 4 |
| INTERACTION | 9 | 6 | 8,875 | 5,75 |
| CONCEPTUAL OBJECT | 0,5 | 2 | 1,75 | 2,5 |
| BEHAVIOUR | 5 | 6 | 4,875 | 5,875 |

Figure 4-14 presents the mode comparison results of the key components of sociotechnical systems in prototypes developed in Group 1 and Group 2. The results show that majority of the projects in both groups implemented simple attributes rather than types of conceptual objects. However, the results in Figure 4-15 show that more projects in Group 2 (63%) implemented conceptual object types than in Group 1 (50%). These results indicate that the JADE guidelines together with the JADE website resources are more effective for the development of conceptual objects of sociotechnical systems on the JADE Framework compared to using just the resources of the JADE website for the same purpose.

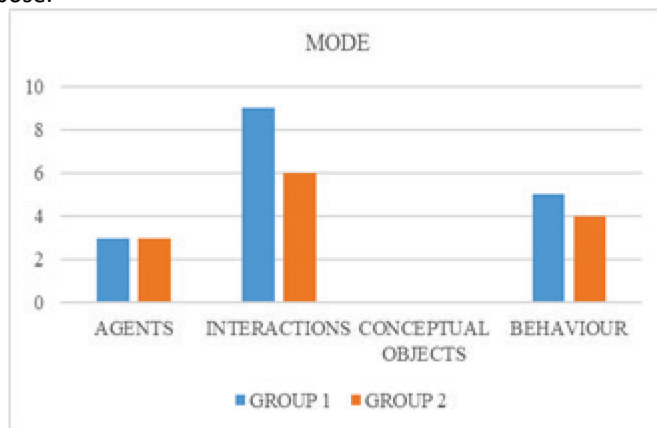


Figure 4-14: Mode comparison of prototypes implemented in Group 1 and Group 2 [89].

Moreover, the results in Figure 4-15 show that 50% of the prototypes developed in Group 2 implemented the ontology and the implemented agents share knowledge among them through the ontology while in Group 1 none of the prototypes implemented the ontology. Again, these results indicate that the JADE guidelines together with the JADE website resources are more effective for the development of ontologies of sociotechnical systems on the JADE framework than just using the JADE website resources for the same purpose. In summary, the JADE guidelines together with the JADE website resources are more effective for the development on the JADE framework of agent knowledge for sociotechnical systems than using only the JADE website resources for the same purpose.

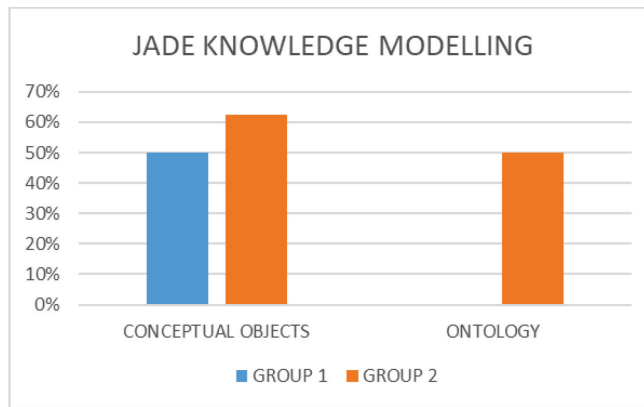


Figure 4-15: Types of conceptual objects and ontology used in prototypes of Group 1 and Group 2 [89].

Table 4-4 refines the comparison between the number of conceptual object types and ontologies implemented in the prototypes developed by the subjects of Group 2. The results clearly show that all the prototypes that developed the ontology also managed to develop conceptual object types but not all the prototypes that implemented the conceptual object types managed to implement the ontology. These results indicate that conceptual object types may be necessary components for the development of an agent ontology on the JADE Framework.

Table 4-4: Conceptual object types and ontologies developed in Group 2 [89].

| PROJECT ID | CONCEPTUAL OBJECTS | ONTOLOGY |
|------------|--------------------|----------|
| 1 | 4 | Yes |
| 2 | 2 | Yes |
| 3 | 0 | No |
| 4 | 0 | No |
| 5 | 6 | Yes |
| 6 | 0 | No |
| 7 | 2 | Yes |
| 8 | 6 | No |

4.5 Summary

Agents and their interactions, knowledge and behaviours are the key building blocks of sociotechnical systems. Therefore, this Chapter proposes novel guidelines that support the development of design models of sociotechnical systems on the JADE framework. These guidelines are divided into JADE-specific interaction development, behaviour development, and knowledge development as described in Section 4.3. The latter involves implementation of conceptual objects and ontology for the given sociotechnical system in JADE framework. Furthermore, this Chapter describes the development of an intruder handling system on the JADE Framework to demonstrate the feasibility of the JADE prototyping guidelines. Lastly, this Chapter analyses the results of an empirical

study that evaluates the effectiveness of the JADE guidelines together with the JADE website resources against the usage of the JADE website resources alone.

The study involved the development of 16 sociotechnical systems of different problem domains by two groups. Group 1 developed 8 prototypes using just the JADE website resources, while Group 2 developed 8 prototypes using the JADE guidelines together with the JADE website resources. The results of this empirical study do not show substantial difference between Group 1 and Group 2 in the development of agents and their interactions and behaviours on the JADE Framework. However, the results show that more prototypes in Group 2 (63%) implemented conceptual objects than in Group 1 (50%). Furthermore, the results express that 50% of the prototypes developed in Group 2 implemented the ontology that was used for sharing knowledge among the agents, while in Group 1 none of the prototypes implemented the ontology.

Therefore, the results of the empirical study conclude that the JADE guidelines together with the JADE website resources are more effective for the development of agent knowledge for sociotechnical systems on the JADE framework than using only the JADE website resources for the same purpose. Additionally, these empirical results affirm the research findings by [41] that conceptual objects are necessary building blocks for the development of ontologies.

5 Tool Support for Requirements and Design Modelling

5.1 Introduction

Chapter 3 proposes novel Coloured Petri Nets (CPN) modelling guidelines that support modelling of agent-oriented design models of sociotechnical systems in CPN Tools. Subsequently, provide visualisation, validation and verification of sociotechnical systems through simulation in CPN Tools. Moreover, Chapter 4 proposes novel Java Agent Development (JADE) prototyping guidelines that support prototyping of agent-oriented design models of sociotechnical systems in JADE framework. Subsequently, enhance the understanding of the customer requirements at an early stage of the development process of sociotechnical systems. However, the use of the proposed guidelines requires conceptual agent models to be syntactically correct and consistent with each other with respect to the AOM4STS methodology. Therefore, the objective of this Section is to describe key features of a novel software tool that aims to reduce modelling effort and improve the effectiveness during requirements elicitation and design of sociotechnical systems by employing visual effects and providing the support for consistency checking and information propagation. To the best of our knowledge, AOM4STS tool support is a novel agent-oriented software tool that focuses on conceptual domain analysis and design modelling of sociotechnical systems by employing the AOM4STS methodology. The close supervision by the author of this thesis led to the development, writing and successful defence of the Master's thesis [123] about AOM4STS software tool.

5.2 The viewpoint framework

In the centre of the AOM4STS methodology lies the viewpoint framework [135] depicted in Table 5-1. It consists of a matrix with three rows representing different abstraction layers and three columns representing the viewpoint aspects of interaction, information, and behaviour. The abstraction layers of the viewpoint framework are "problem domain analysis," "plat-form-independent design," and "platform-specific design and prototyping." In Table 5-1 these layers are entitled for short as "Analysis," "Design," and "Prototyping." Each cell in this matrix represents a specific viewpoint, such as "interaction analysis," "information design," or "behaviour simulation." The cells of the viewpoint framework represent artefacts – tabular models, graphical models, documents, and program code – that are produced by AOM4STS methodology. Conceptually, we consider arte-facts as abstractions reducing the complexity of a sociotechnical system for better understanding of the system's aspects and their impact on its behaviour.

Table 5-1: The viewpoint framework adapted from [135].

| Abstraction layer | Viewpoint aspect | | |
|-------------------|-------------------------------------|------------------|--|
| | Interaction | Information | Behaviour |
| Analysis | Role models and organization model | Domain model | Goal models and motivational scenarios |
| Design | Agent models and interaction models | Knowledge models | Scenarios and behaviour models |
| Prototyping | Platform-specific models | | |

Agent-oriented models for problem domain analysis act as a bridge between information technology (IT) and non-IT experts during the requirements elicitation phase in the development of the sociotechnical system. These models provide a high-level description of sociotechnical systems and use visual notations to enable all project stakeholders to obtain a common understanding on the system requirements.

5.3 Functionality of the AOM4STS Tool

The AOM4STS tool has been designed and implemented to support the AOM4STS methodology. The AOM4STS methodology [2,135,137] involves incremental refinement of models in an iterative manner. Therefore, consistency checking becomes a necessary feature of the AOM4STS tool to ensure that the modelling artefacts represented in Section 2.1.3 and Section 2.1.4 remain consistent with each other. The following from Sections 5.3.1 to 5.3.5 describe key features of the AOM4STS tool.

5.3.1 Saving and Loading

Among common functionalities of any modelling software is the ability to save and load the models. The AOM4STS tool provides the ability to save requirements models that represent problem domain analysis phase of the AOM4STS methodology. This feature uses Extensible Markup Language (XML¹) as an appropriate standard for data encoding due to its ability to represent information across the internet in a simple, generic and usable way [17]. Furthermore, the XML standard represents data in a format that is both human-readable and machine-readable. Figure 5-1 depicts XML representation of modelling artefacts of the AOM4STS methodology in the intruder-handling case study [134].

¹ <https://www.w3.org/XML/>

```

16 <responsibilities></responsibilities>
17 </shape>
18 <shape>
19 <id>36</id>
20 <x>241</x>
21 <y>287</y>
22 <width>139</width>
23 <height>59</height>
24 <imagetype>FG</imagetype>
25 <generalImagetype>BehaviourAgentChild</generalImagetype>
26 <name>Notice</name>
27 <description></description>
28 <constraints></constraints>
29 <responsibilities></responsibilities>
30 </shape>
31 <shape>
32 <id>39</id>
33 <x>468</x>
34 <y>284</y>
35 <width>143</width>
36 <height>61</height>
37 <imagetype>FG</imagetype>
38 <generalImagetype>BehaviourAgentChild</generalImagetype>
39 <name>Identify</name>
40 <description></description>
41 <constraints></constraints>
42 <responsibilities></responsibilities>
43 </shape>

```

Figure 5-1: XML representation of represents requirements models of AOM4STS methodology.

5.3.2 Online Diagramming

The AOM4STS tool enables analysts and designers to apply the AOM4STS methodology in a user-friendly way. This means that with the help of the AOM4STS tool one can create models of AOM without the need to download or install any software. The results of testing the AOM4STS tool demonstrate that the tool works correctly on different web browsers such as Google Chrome¹, Mozilla Firefox², and Internet Explorer³. Figure 5-2 depicts a part of the goal model of the intruder-handling case study [134] on (a) Google Chrome (b) Mozilla Firefox and (c) Internet Explorer web browser.

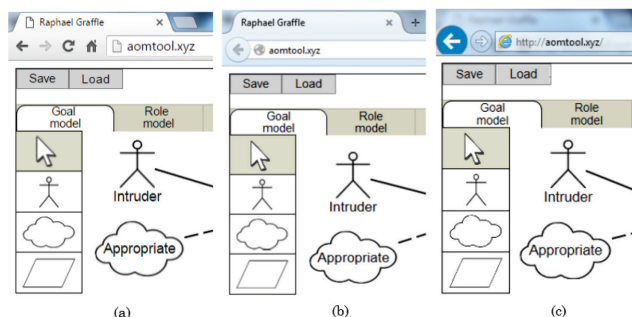


Figure 5-2: The AOM4STS tool on (a) Google Chrome (b) Mozilla Firefox and (c) Internet Explorer web browser.

¹ <https://www.google.com/chrome/>

² <https://www.mozilla.org/>

³ <http://windows.microsoft.com/en-us/internet-explorer/>

The advantages of using an online diagramming software include avoiding conflicts between installed software, saving time, and facilitating conducting for the participants of a workshop a fast training session on AOM4STS methodology.

5.3.3 Graphical User Interface

The AOM4STS tool provides a direct manipulation of graphical interface that helps analysts to create all requirements models of the AOM4STS methodology [2,135,137], which include goal model, role models, organisation model, and domain model. As presented in Section 2.1, the AOM4STS methodology uses notations, syntax and model types different from the existing AOSE methodologies. Moreover, among the major differences between the AOM4STS methodology and other AOSE methodologies is the inclusion of humans as essential parts of the system. For example, Figure 5-3 represents a goal model of the intruder-handling case study [133] in which the clouds represent quality goals, parallelograms represent functional goals and the stickmen represents roles. The latter can be mapped to a software, device or human during the design phase.

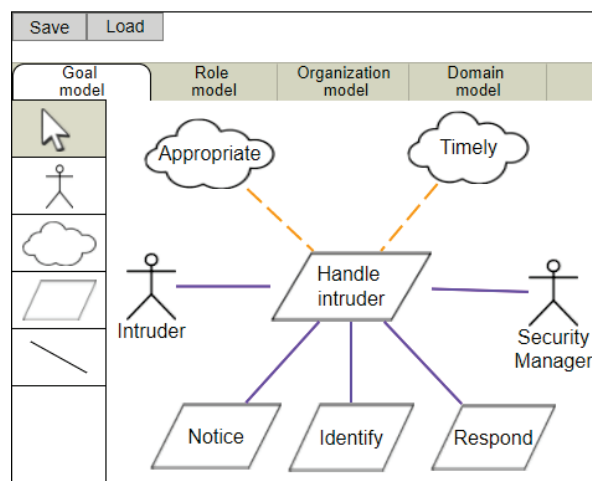


Figure 5-3: Modelling notations captured in the AOM4STS tool [88].

5.3.4 Information propagation

According to Table 5-1, models in the AOM4STS methodology are divided horizontally along three abstraction layers and vertically along three viewpoint perspectives. Considering this, during the modelling process, the AOM4STS tool propagates information vertically across abstraction layers and horizontally across viewpoint perspectives.

In the vertical information propagation, models for problem domain analysis act as input for platform-independent design models while platform-independent design models act as input for platform-specific design models and prototypes. For example, domain model at the problem domain analysis layer acts as an input for knowledge model at the platform-independent design layer. Furthermore, knowledge model at the platform-independent design layer becomes an input for data and service models at the platform-specific design layer.

In horizontal information propagation, AOM4STS propagates information across models of different viewpoint perspectives but within the same abstraction layer. The problem domain analysis layer contains five different models as described in Section 2.1.2. The information in these models are propagated horizontally across the three viewpoint perspectives. For example, all the roles identified during goal modelling are horizontally propagated to role models, organisation model and domain model. Figure 5-4 represents a screenshot of the AOM4STS tool that shows a list of roles identified during goal modelling of the intruder-handling case study [133] and propagated to the role modelling tab.

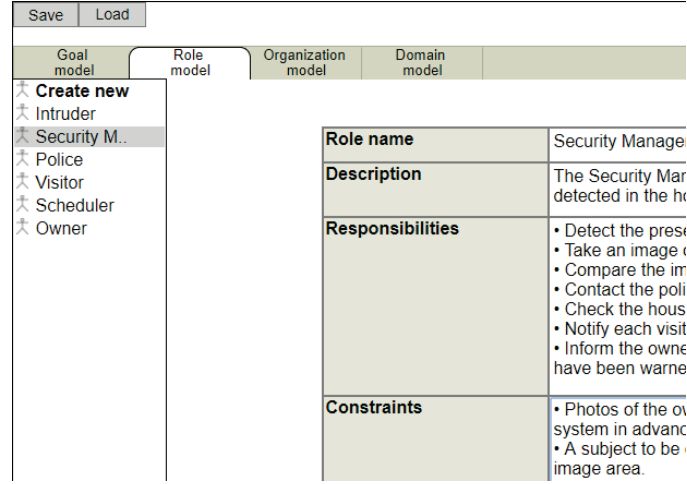


Figure 5-4: Information propagated from goal model to role model in the AOM4STS tool.

5.3.5 Consistency checking

The AOM4STS tool continuously performs consistency checking to prevent certain errors from being made in the first place. The errors checked against are definition errors, simple typing errors, and violations of scope. The principle of detecting definition errors is that it is only possible to create a reference to an entity after the entity has been defined. For example, it is only possible to create a reference to a role in a domain knowledge model after that role has been defined in the goal model. Moreover, when a user deletes an entity, the tool deletes all references to the deleted entity.

The principle of detecting simple typing errors is that the tool allows users to create only syntactically correct connections between component types. The tool prevents all syntactically wrong connections and generates the corresponding error messages in the bottom frame of the tool containing user activity logs. For example, according to the AOM4STS methodology, it is syntactically wrong to create a connection between a role and quality goal in the goal model. Figure 5-5 depicts the AOM4STS tool displaying an error message because of an attempt to create a syntactically wrong connection between the role Intruder and quality goal Appropriate.

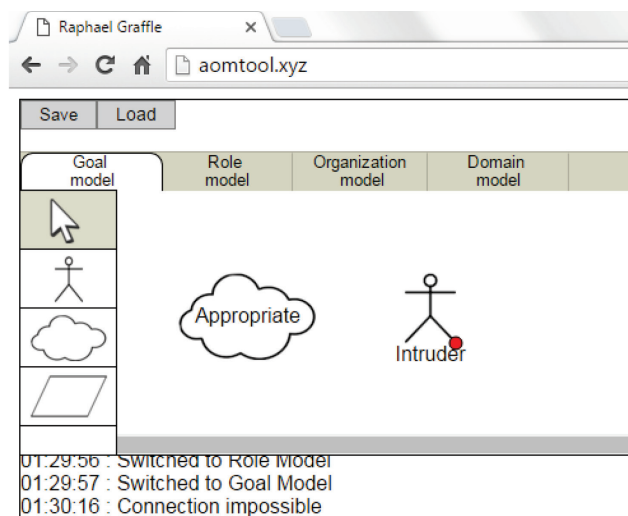


Figure 5-5: A syntactical error message displayed in the activity logs' frame.

Lastly, in preventing violations of scope constraints, the tool allows an analyst to neither increase nor decrease the scope of the project identified during the earlier modelling stages. That is to say, the goal modeller has defined the scope of the project, role modeller, organisation modeller and domain modeller can only refine the goal model but not increase or decrease the scope of the project. For example, the roles of sociotechnical system can only be added using goal models and propagated to other models of problem domain analysis for further refinement. Apart from goal model, other models of the problem domain are prevented by the tool from adding or deleting any role of the sociotechnical system. Moreover, the addition or deletion of a role in the goal model results to the addition or deletion of the corresponding role in other models. If the need to either increase or decrease the scope of the project appears during the modelling process, the request needs to be sent to the responsible modeller to change the scope of the project to be able to incorporate those changes in the corresponding model. This makes the AOM4STS tool suitable for an iterative (agile) modelling process that supports the AOM4STS methodology.

5.4 Summary

This Chapter describes the AOM4STS tool support, which is a novel agent-oriented software tool that focuses on conceptual domain analysis and design modelling of sociotechnical systems by employing the AOM4STS methodology. To provide a clear understanding of the AOM4STS tool support, this chapter briefly describes the viewpoint framework to show the relationships between models produced by AOM4STS methodology. Furthermore, this Chapter describes key features of the AOM4STS tool that aims to reduce modelling effort and to improve the effectiveness during requirements elicitation and design of sociotechnical systems. The following Chapter investigates the modelling effort and the effectiveness of using pen and paper in comparison with using the AOM4STS tool support for the requirements modelling of sociotechnical systems.

6 Empirical Evaluation of AOM4STS Tool

6.1 Introduction

The AOM4STS tool, presented in Chapter 5, is an online diagramming software tool that supports the methodology of requirements engineering for sociotechnical systems put forward in Section 2.1 of this thesis. In this Chapter, we present an empirical study for evaluating requirements modelling for a socio-technical system with the AOM4STS tool in comparison with modelling the requirements for the same sociotechnical system using pen and paper.

The evaluation of a modelling approach can be characterized by two main aspects [154]: (1) the effort during modelling; and (2) the effectiveness of the modelling process. Accordingly, we first evaluated if the effort of modelling requirements with the AOM4STS tool was lower than the effort of modelling requirements on paper. Secondly, we evaluated if the effectiveness of modelling requirements with the AOM4STS tool was higher than the effectiveness of modelling requirements on paper.

With the objective of evaluating possible benefits of using the AOM4STS tool for modelling requirements for sociotechnical systems, in comparison with the use of pen and paper, we defined the following two research questions.

RQ3(b): What is the impact of the software tool on the effort needed for requirements and design modelling of sociotechnical systems employing the AOM4STS methodology?

RQ3(c): What is the effectiveness of the software tool for requirements and design modelling of sociotechnical systems employing AOM4STS methodology?

The design of the experiments follows the guidelines by Wohlin *et al.* [154] on how to set up and document empirical studies in software engineering in order to give answers to the identified research questions.

6.2 Experiment Planning

This Section describes the plan for the experiment that was followed during the empirical study for evaluating modelling the requirements for a socio-technical system with the AOM4STS tool.

6.2.1 Goal of the study

The goal of the empirical study is to compare software-based processes of modelling requirements for sociotechnical system against paper-based processes of modelling the same requirements to find out if the benefits expected from the AOM4STS tool are also obtained, if novice users of the AOM4STS methodology and tool use it. Hence, the main factors of this experiment are the modelling approaches that we want to compare, i.e. Modelling on Paper (MoP) and Modelling on Software (MoS).

6.2.2 Context selection

The experiment was run in a lecture room at Tartu University in Tartu, Estonia. The participants of the experiment were postgraduate students (MSc and PhD) taking the requirements engineering course. The experiment was run off-line, as a blocked subject-object study [154]. The two objects, i.e. the requirement specifications of two sociotechnical system were assigned to each participant (subject).

6.2.3 Objects of study

To achieve expressive results in a study, the experiment must be fair and not give an advantage to any one of the treatments – the MoP and MoS for this study. The former allows subjects to use pencil and paper to create the models of the requirements, while the latter allows subjects to use the AOM4STS¹ tool for the same purpose. This is an online diagramming software tool that supports the principles of visual expressiveness, information propagation across the requirements models, and consistency checking during requirements modelling. The difficulty lied in the selection of modelling tasks that can be performed with both approaches and are not only tailored towards the functionalities of the AOM4STS tool but are nonetheless challenging enough to prompt significant results.

The objects of this study were two small sociotechnical systems – a Meeting Scheduler System (MSS) [156] and a Personalized Emergency System (PES) [94]. The former is a computer-based service that supports setting up meetings while the latter is a system that supports a person, generally an older person, to remain living at home longer.

6.2.4 Subjects

The participants of the experiment were 8 post-graduate students (MSc and PhD) taking the requirements engineering course. Among various sub-topics of this course are goal-oriented approaches and agent-oriented methodologies for requirements engineering. These participants of this study were not students taught by experimenters. Furthermore, they were using paper and paper in their requirements engineering course.

6.2.5 Experiment design

This study adopted a paired, counterbalanced experiment that was conducted for 3 hours over two consecutive days, i.e. 90 minutes in day 1 and another 90 minutes in day 2. In this experiment design, each subject performed the experiment tasks with both objects and with both treatments. This means that, in day 1, half of the subjects were given the PES object and the remaining half of the subjects were given the MSS object. Moreover, half of those who received PES conducted the modelling on paper and the other half with the software tool. Similarly, half of those subjects who received MSS conducted the modelling on paper and the other half with the software tool. In day 2, each subject changed the object and treatment. This experiment design mitigates the learning effects between the two objects and between the two treatments.

The subjects were randomly divided into 4 groups wherein the two treatments and two objects, PES and MSS, were associated as described in Table 6-1. The paired design, with the same number of subjects in each group, enables a better comparison and the application of more precise statistical methods.

¹ <http://www.tud.ttu.ee/im/Msury.Mahunnah/AOM4STS/>

Table 6-1: Assignment of objects and treatments to subjects in the two days of the study.

| | Day 1 | Day 2 |
|--------|-----------------|-----------------|
| Case 1 | PES on Paper | MSS on Software |
| Case 2 | MSS on Software | PES on Paper |
| Case 3 | PES on Software | MSS on Paper |
| Case 4 | MSS on Paper | PES on Software |

6.3 Modelling Experiment

This Section shows the design, procedure, analysis, and results of the performed experiment.

6.3.1 Aspects for Research Questions

These two research questions RQ3(b) and RQ3(c) are characterised by two abstract terms, *effort* and *effectiveness* respectively. These terms need to be detailed to associate them with the variables that can be evaluated by the experiment. Therefore, we decompose RQ3(b) and RQ3(c) into various aspects that characterise them and subsequently, define the terms effort and effectiveness for the scope of the study. RQ3(b) is decomposed into aspects based on the time spent, the effort perceived by the subjects, and the difficulties encountered by the subjects during modelling. The following are detailed aspects for RQ3(b):

- (a1) overall time spent on a modelling task;
- (a2) adequateness of the effort required for creating the models. We want to know if the modelling effort is subjectively perceived to be adequate by the subjects;
- (a3) effort distribution. We seek to study the changes in the distribution of the time spent on the following activities:
 - (a) reading a description of the modelling language;
 - (b) reading a description of the system requirements;
 - (c) modelling the system requirements.
- (a4) difficulties encountered in modelling. We want to know if the subjects perceived some difficulties in modelling the requirements and in using the modelling concepts. Here we want to study:
 - (a) the difficulty of creating a goal model;
 - (b) the difficulty of creating a domain model.

The aspects for RQ3(c) consider the expressiveness and effectiveness of the functionalities of the modelling software tool, as perceived by the subjects. The following are detailed aspects for RQ3(c):

- (a5) the perceived expressiveness of the modelling AOM4STS tool. We are interested in the subject's opinion on the adequateness of the features provided by the modelling software tool for describing the requirements of sociotechnical system.
- (a6) perceived effectiveness of models created for the development of sociotechnical system: the subjects, as potential users of the modelling software tool, should comment on the utility of the created requirements models for software designers and developers.

(a7) perceived utility of capabilities of the AOM4STStool for modelling requirements for sociotechnical system. We would like to know the opinion of the subjects on the following functionalities of the AOM4STS tool:

- (a) information propagation;
- (b) consistency checking;
- (c) coloured visual variables;
- (d) overall usability.

In this study, each aspect has been evaluated with its own research question of the same form as the high-level research questions RQ3(b) and RQ3(c). For instance, let us consider the aspect (a1). Its research question is RQa1: "Is the time required to model requirements with the AOM4STS tool higher than the time required to model the requirements on paper?"

6.3.2 Variables and measures

The independent variable and main factor of this study is the modelling approach used for modelling requirements for sociotechnical system, considering the treatments of modelling with the AOM4STS tool and modelling on paper. This variable is manipulated and controlled and should, therefore, be independent of the objects, subjects, and experimental tasks to reduce threats to the validity of the results, as explained in Section 6.5.4.

The dependent variables are the 7 aspects (a1,...,a7) identified and assessed through questionnaires filled by subjects before and after each experimental task. These dependent variables are grouped according to the continuous and Likert scale. The continuous variables are a1 and a3 that measure the time spent on the whole modelling process and the fractions of time (in percentages) spent on various modelling activities. The Likert scale is applied to the variables a4,...,a7. Normally, a Likert scale variable specifies the level of agreement with a statement. In this study, the Likert scale is defined using an ordinal scale from 1 to 5 as follows: 1 – strongly disagree; 2 – disagree; 3 – not certain (neutral response); 4 – agree; 5 – strongly agree.

6.3.3 Experiment procedure and materials

The experiment procedure consisted of the following steps:

1. Tutorial on the AOM4STS methodology.
2. Short demo of using the AOM4STS modelling tool.
3. Filling of pre-questionnaire.
4. Modelling of case studies for day 1.
5. Filling of questionnaires for day 1.
6. Modelling of case studies for day 2.
7. Filling of questionnaires for day 2.
8. Filling of post-questionnaire.

Since the subjects had different levels of experience with goal-oriented modelling, requirements engineering, and sociotechnical system, to prepare them for the experiment we gave a presentation on the AOM4STS methodology for 15 minutes and a short demo of the AOM4STS modelling tool for another 15 minutes. The questionnaires and modelling tasks were done individually by each participant in a time-frame of approximately 2 hours. Out of this duration, 1 hour was spent on modelling case studies and filling in the questionnaires for day 1 and another hour was spent on modelling case studies and filling in the questionnaires for day 2.

To perform the experiment, each participant received the prepared input materials – the detailed description of the experiment procedure, the pre-questionnaire, the post-questionnaire, and the following materials on each day: (1) requirements specification for the case study, and (2) a questionnaire based on the case study modelled by the subject. A complete set of documents is

As the task for the experiment, the subjects had to create two requirements models – a goal model and a domain model – for two case studies, one of which had to be modelled on paper and another one with the AOM4STS tool, as is described in Table 6-1. Each case study had to be modelled with as many details as possible, with the given treatment, and by following the step-by-step description of the requirements for each case study. Before the beginning of the modelling task, each subject had to fill in a pre-questionnaire. After completing the modelling task for each case study, the subject had to fill in a questionnaire about the corresponding case study and treatment used. Finally, each subject had to fill in a post-questionnaire. A collection of questions for each type of questionnaire is provided in Table 6-2, where “preq” stands for pre-questionnaire, “q” for the questionnaire and “postq” for post-questionnaire.

Table 6-2: A set of the questions in the questionnaires, with answers on a 1 . . . 5 Likert scale [88].

| 1 – Strongly disagree 2 – Disagree 3 – Neutral 4 – Agree 5 – Strongly agree | |
|--|--|
| preq4 | Basic principles of AOM4STS modelling are clear |
| preq5 | The visual notations in the AOM4STS methodology are clear |
| preq6 | Basic knowledge of using the AOM4STS tool has been acquired |
| q4 | The description of the case study was clear to me |
| q5 | I had no difficulties in modelling the goal model |
| q6 | I had no difficulties in modelling the domain model |
| q7 | I had enough time for accomplishing the modelling task |
| q8 | Goal decomposition was very useful for this task |
| q9 | The concepts of the AOM4STS methodology were detailed enough to model the requirements of the system |
| q10 | The effort of modelling seems too high for an efficient use of the methodology in practice |
| postq2 | The propagation of roles created in the goal model into the domain model is helpful for the modeller |
| postq3 | The propagation of changes made to the roles in the goal model into the domain model helps to reduce the modelling effort |
| postq4 | The modelling software supports creation of syntactically correct models by preventing and reporting syntactically wrong connections |
| postq5 | The use of coloured connections in the creation of the models by the modelling software helps to improve the readability of the resulting models |

The pre-questionnaire aimed to assess the knowledge of the subjects with respect to computing studies, requirements engineering, and agent-oriented modelling. Moreover, the questions from preq4 to preq6, as represented in Table 6-2, aimed to assess the knowledge of the AOM4STS methodology acquired after completing the

tutorial, and therefore measured the adequateness of the given tutorial before the modelling experiment.

The questionnaire associated with each treatment included the questions from q4 to q10 as described in Table 6-2, which evaluated the adequateness of the case study objects and the time used; and collected the perceptions by subjects based on the specific treatment applied. Furthermore, the overall time needed for completing the experimental task was recorded before filling in the corresponding questionnaire. The participants were also asked to keep track in fractions (in %) of the time spent on various activities. An indicative time of 1 hour was given to the subjects as a suggestion for performing the experimental modelling task on each day of the experiment, but the subjects were free to take as much time as they required for completing the experimental task. The questions about the time spent on activities in each experiment are presented in Table 6-3.

Table 6-3: Questions about the time spent on the activities in each experiment [88].

| Question label | Question description |
|----------------|---|
| duration | Time used for the task, in minutes |
| q1 | Reading the description of the AOM4STS methodology in % |
| q2 | Reading and understanding the case study in % |
| q3 | Modelling the case study in % |

Finally, the questions in the post-questionnaire from postq2 to postq5, as listed in the bottom of Table 6-2, collected data about the effectiveness of modelling requirements with the AOM4STS tool as compared with modelling requirements on paper.

6.4 Data Analysis

For analysing the treatments, i.e. Modelling on Paper (MoP) in comparison to Modelling on Software (MoS), we perform the following tasks. First, mapped each aspect from a1 to a7, as is described in Section 6.3.1, to one or more questions in the questionnaires. Second, applied measures of central tendency, i.e., mean, median and mode to compare the impact of each aspect on the treatments. Third, grouped the results based on the aspects identified to answer the two research questions RQ3(b) and RQ3(c).

The results of the applied measures of central tendency are presented in Section 6.5. In addition to the measures of central tendency. To evaluate the two treatments, the questions from q1 to q10 were repeated for each treatment used during the experiment. Similarly, the answers to the questions included in the post-questionnaire that mostly focused on the evaluation of the AOM4STS tool were compared with respect to the value 3, which is the neutral answer in the Likert scale used in the study. The answers to the questions q1 to q3 captured the relative time spent on reading the tutorial, understanding the case study and modelling requirements for the system in percentage. Then, were multiplied with the overall time used by the subject in that experiment in order to obtain time measurements that could be compared between the two treatments.

6.5 Results and Interpretation

This Section presents the results of the empirical study by considering the median, mean and mode values of the data collected from the subjects for each aspect and provides an interpretation of the results to answer RQ3(b) and RQ3(c).

6.5.1 Adequateness of the experimental settings

Before analysing the main factors of the empirical study, we consider whether the settings for the experiment were adequate. The pre-questionnaire contained several questions to evaluate if the subjects encountered any difficulty with the modelling methodology, if the experiment was performed under time pressure, and if the description case studies were clear. Moreover, the pre-questionnaire asked about the subject's experience in the fields of computing, requirements analysis, and agent-oriented modelling in order measure the influence of these co-factors on the study.

Although all the subjects were postgraduate students in the requirements engineering course, they had different levels of experience in computing. Figure 6-1 shows the distribution of the subjects with respect to their experience in computing. Half of the subjects had little knowledge in computing – 38% had gained experience through research projects and 12% through working as computing professional in IT companies.

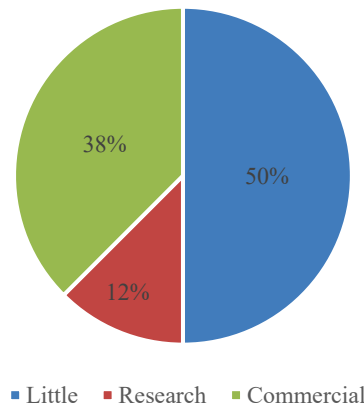


Figure 6-1: Distribution of the subjects with respect to their experience in computing.

As has been explained in Section 6.2.4, all the subjects were registered for the requirements engineering course to acquire new knowledge or improve their knowledge of requirements engineering. The results in Figure 6-2 show the distribution of the subjects with respect to their experience in requirements analysis. 75% of the subjects had little such experience, while the remaining 25% had research experience in requirements analysis.

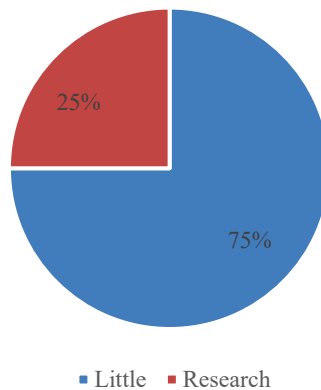


Figure 6-2: Distribution of the subjects with respect to their experience in requirements analysis.

Before the subjects started the experiment, we gave a short tutorial about agent-oriented requirements modelling. The tutorial provided overviews of agent-oriented goal modelling and domain knowledge modelling, as described in Section 2.1.3 of this thesis and in [135]. After the tutorial, we did a short demonstration on agent-oriented goal modelling and domain knowledge modelling with the AOM4STS tool. To be able to measure the effectiveness of the tutorial for the subjects, we decided to measure the prior experience of the subjects in agent-oriented modelling. The results in Figure 6-3 show the distribution of the subjects with respect to their prior such experience. 75% of the subjects did not have any experience in agent-oriented modelling, while 25% had little experience.

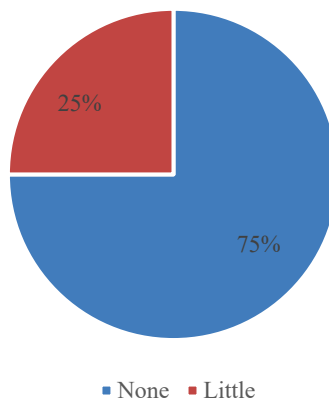


Figure 6-3: Distribution of the subjects with respect to their prior experience in agent-oriented modelling.

The results in Table 6-4 provide a summary of the adequateness of the experiment settings. In the pre-questionnaire, the questions preq4 to preq6 aim to assess the

subjects' understanding of goal-modelling and domain knowledge modelling, their modelling notations, and of the supporting modelling tool after completing the tutorial.

Table 6-4: Results of the adequateness of the subjects [88].

| Reference | Question | Median |
|-----------|---|--------|
| preq4 | Basic principles of AOM4STS modelling are clear | 5 |
| preq5 | The visual notations in the AOM4STS methodology are clear | 5 |
| preq6 | Basic knowledge of using the AOM4STS tool has been acquired | 5 |

The results presented in Table 6-4 show that the median value for answers to the questions preq4 to preq6 is 5, which stands for “strongly agree”. This means that the subjects strongly agreed that the basic principles of AOM4STS modelling were clear, the visual notations in the AOM4STS methodology were clear, and the basic knowledge for using the AOM4STS tool had been acquired after attending the tutorial and demonstration of the modelling software. These results show that the subjects acquired an adequate understanding of the AOM4STS methodology and the modelling tool for participating in the modelling experiment and giving undistorted feedback.

The results presented in Table 6-5 provide a summary of the adequateness of the objects used in the experiment. After completing the modelling task, each subject was asked questions q4 and q7 independently of the treatment. Question q4 evaluates if the description of the case study was clear, while question q7 evaluates if the subject had enough time for accomplishing the modelling task.

Table 6-5: Results of the adequateness of the objects [88].

| Reference | Question | Median (PES) | Median (MSS) |
|-----------|--|--------------|--------------|
| q4 | The description of the case study was clear to me | 5 | 4 |
| q7 | I had enough time for accomplishing the modelling task | 4 | 4 |

In light of the answers given to question q4, the median value for the PES case study was 5, while the median value for the MSS case study was 4. Thus, the subjects considered that the descriptions of both cases studies were nearly equally clear, although the description of the PES case study was seen as clearer compared to the description of the MSS case study. While there was a slight variation according to the Likert scale, we believe the objects were adequate to provide unbiased results, because both results were above the median value 3. On the other hand, for question q7, the median value for both the PES and MSS case studies was 4. In other words, the subjects agreed that they had enough time for accomplishing the modelling task. The value 4 for each case study reduces the possibility of having biased results with respect to the time allocated for the experiment (it should be noted that there was no fixed time limit given to the subjects). Moreover, since the result for question q4 was above the neutral value – 3 in the 1...5 Likert scale – the subjects were not under time pressure when performing

the modelling tasks. Consequently, the time allocated for the experiment did not have any influence on the results. Therefore, we can claim that overall, the settings for the experiment were adequate.

6.5.2 Main factor: results and interpretation

In this section, we provide the results for the main factor of the experiment – the approach used – and compare the two treatments.

6.5.2.1 Evaluation of modelling effort

In this Section, we provide an answer to the research question RQ3(b) addressing the modelling effort, which was stated in Section 6.1, based on the mean values represented in Figure 6-4 and variance values shown in Figure 6-5.

The question q0 records the overall time used by a subject for modelling a case study. The mean for modelling on paper (30) was nearly the same as the mean for modelling with the tool (29.6). However, the variance for modelling on paper (5.8) is noticeably higher than that for modelling with the tool (3.5).

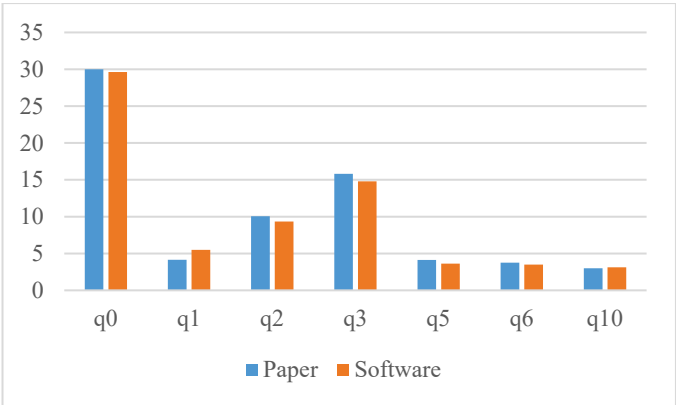


Figure 6-4: Mean values for comparing modelling effort of the two treatments [88].

The question q10 records modelling effort perceived by the subjects. The mean value of the modelling effort perceived by the subjects for modelling on paper and modelling with the tool were both close to 3 and their variances close to 0.7. Therefore, the subjects perceived the modelling effort on paper to be the same as the modelling effort with the tool.

The question q1 records the time used by the subjects for reading and understanding the modelling methodology. The mean time used by the subjects for reading and understanding the modelling methodology was slightly higher for subjects who conducted modelling with the tool (5.5) compared to those who modelled on paper (4.1). The variance of the time used by the subjects for reading and understanding the methodology was noticeably higher for subjects who conducted modelling with the tool (2.6) compared to those who modelled on paper (1.3).

Moreover, the question q2 records the time used by the subjects for reading and understanding the description of the case study. The mean time used by the subjects for reading and understanding the case study was slightly lower for subjects who conducted modelling with the tool (9.3) compared to those who modelled on paper (10.1).

The variance of the time used by the subjects for reading and understanding the case study was noticeably lower for subjects who conducted modelling with the tool (3.1) compared to those who modelled on paper. Furthermore, the question q3 records the time consumed by the subjects for conducting the actual modelling using the two treatments. The mean time used by the subjects for conducting the actual modelling with the tool (14.8) was slightly lower than that for those who conducted the actual modelling on paper (15.8). The variance of the time used by the subjects for conducting the actual modelling with the tool was noticeably lower (3.6) than that for conducting the actual modelling on paper (6.7).

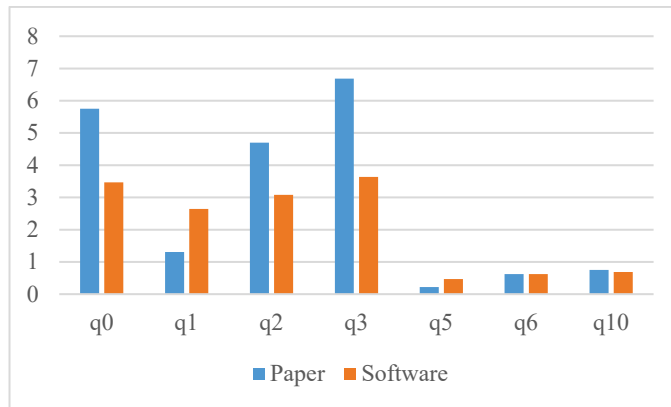


Figure 6-5: Variance values for comparing modelling effort of the two treatments [88].

The question q5 records the difficulty perceived by the subjects during goal modelling while q6 records the difficulty perceived by the subjects during domain knowledge modelling. For q5, the mean value of the difficulty perceived during goal modelling when modelling with the tool (3.6) is slightly lower than that for modelling on paper (4.1). However, the variance of the difficulty perceived by subjects during goal modelling when modelling with the tool (0.5) is noticeably higher than that for modelling on paper (0.2). For q6, the mean value of the difficulty perceived during domain modelling when modelling with the tool is slightly lower (3.5) than that for modelling on paper (3.8) but their variances are the same (0.6).

Considering all the collected data for q0 to q6 and q10, we must answer the research question RQ3(b) as follows: the modelling effort on paper is nearly the same as the modelling effort with the AOM4STS software tool. However, the variance values for comparing the modelling efforts of the two treatments are considerably different. In the reported study the higher variance values for the modelling effort on paper dominate as compared with the modelling effort with the tool. An explanation for this is that the tool imposes more constraints on the requirements modelling activities. The higher variances of the time for the questions q1 and q5 when using the tool require further research.

6.5.2.2 Evaluation of modelling effectiveness

In this Section, we provide an answer to the research question RQ3(b) addressing the modelling effort, which was stated in Section 6.1, based on the mean values represented in Figure 6-6 (a) and variance values shown in Figure 6-6 (b). The question q8 records the usefulness of goal decomposition during goal modelling. The mean value for the

usefulness of goal decomposition on paper (3.8) as perceived by the subjects was slightly higher than that of modelling with the tool (3.5) with the variance of 0.8 when modelling on paper and 0.9 when modelling with the tool.

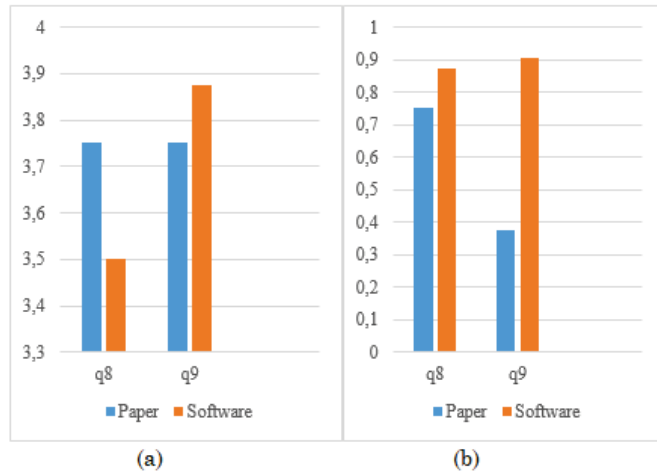


Figure 6-6: (a) Mean and (b) Variance for comparing modelling effectiveness of the two treatments [88].

The question q9 records the utility of the concepts of goal modelling and domain knowledge modelling perceived by the subjects for requirements modelling. The mean value of the subjects who conducted modelling with the tool was 3.9 while the same value for those who conducted modelling on paper was 3.8. The variance of the subjects who conducted modelling with the tool was 0.9 while the same value for those who conducted modelling on paper was 0.4.

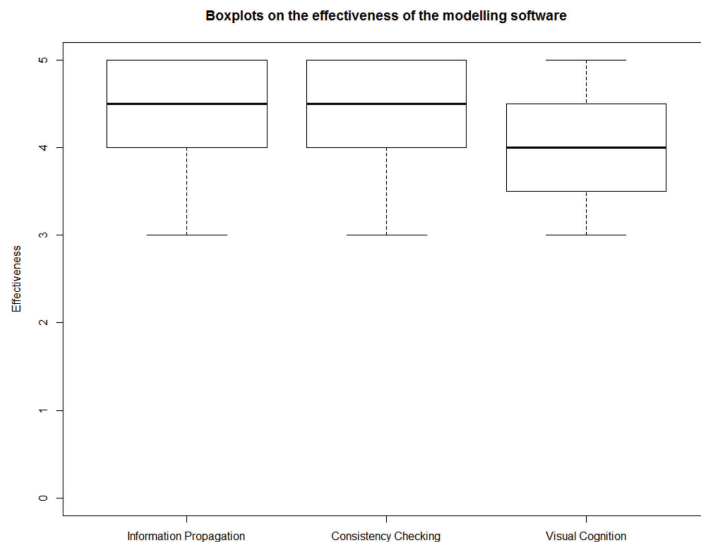


Figure 6-7: Boxplot on the effectiveness of the modelling tool with respect to information propagation, consistency checking, and visual cognition [88].

Furthermore, the subjects agreed on the effectiveness of the key features of the AOM4STS modelling tool that are not present in paper-based modelling with the median values 4.5 for information propagation, 4.5 – for consistency checking, and 4 – for visual cognition. The distribution of these results is depicted by the boxplot presented in Figure 6-7. Moreover, the results in the boxplot clearly show that none of the subjects disagrees or strongly disagrees with the effectiveness of the AOM4STS modelling tool with respect to information propagation, consistency checking, and visual cognition.

Considering all the collected data from q8 and q9 and the postquestionnaire (information propagation, consistency checking, and visual cognition), we must answer the research question RQ3(c) as follows: the effectiveness of modelling requirements with the modelling tool was higher than the effectiveness of modelling requirements on paper except for goal decomposition which was slightly more effective when modelled on paper compared to modelling with the tool.

6.5.3 Additional results

This Section presents additional findings of the empirical study that are not directly related to RQ3(b) and RQ3(c), but are important for a better understanding of the relationship between the two treatments. The graph in Figure 6-8 compares the average time spent (in minutes) on different activities of the experiment – reading the methodology, reading the case study, and modelling.

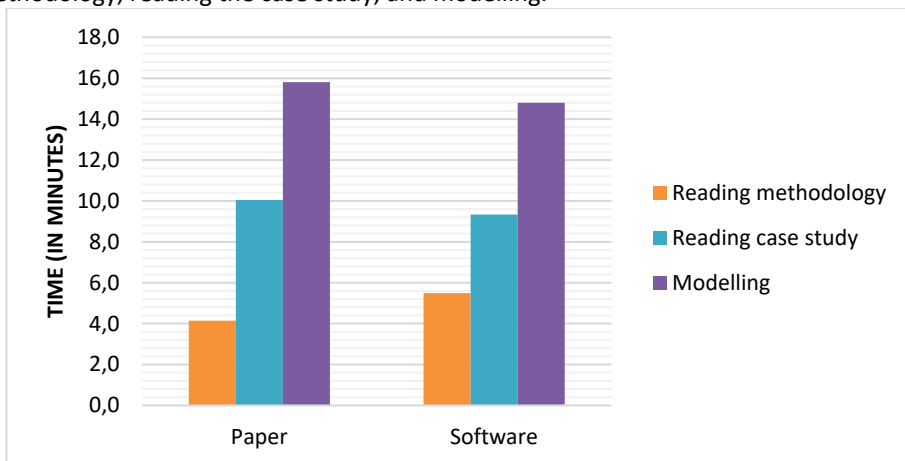


Figure 6-8: Comparison of the average time spent on different activities.

These results demonstrate that the average time spent on modelling with the tool is slightly lower than the average time spent on modelling on paper, while the average time spent on reading the methodology is higher for modelling with the tool as compared with modelling on paper. These results can be explained by the fact that modelling with the tool requires a clearer understanding of the modelling methodology than modelling on paper because modelling with the tool only accepts syntactically correct connections between the nodes of models while modelling on paper allows for any kind of connection between the nodes.

Furthermore, the results in Figure 6-9 represent the boxplots of the time spent on reading the methodology (in minutes) per treatment. Although the median value for both treatments is 4 minutes, the values for reading the methodology in case of modelling with the modelling tool range from 3.1 to 12.1 minutes, while the same values

for modelling on paper range from 1.1 to 7 minutes. These results support the finding presented in Figure 6-8, according to which modelling with the tool requires a more thorough understanding of the methodology as compared with modelling on paper.

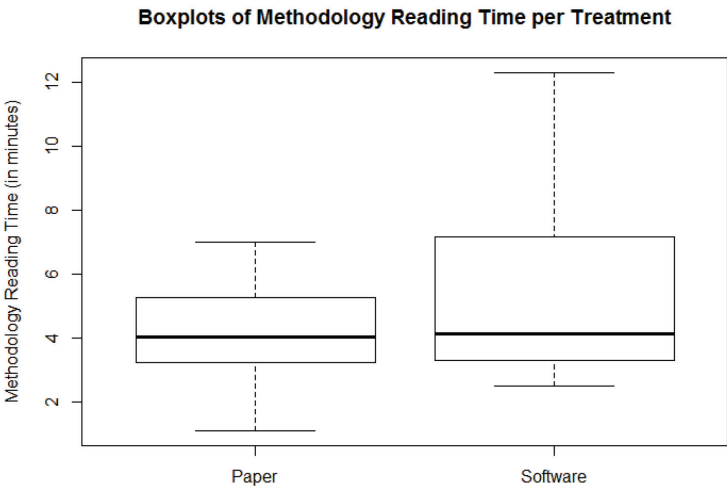


Figure 6-9: Boxplots of methodology reading time per treatment.

As was explained in Section 6.5.2 in the answer to question 10, the time used by subjects for reading the description of the case study is independent of the treatment. The results in Figure 6-10 represent boxplots that describe the distribution of the time used by subjects for reading the case study for each treatment. These results clearly show that the distribution of values for the time used for reading the case studies is nearly the same for both modelling on paper and modelling with the tool, with the exception of one outlier depicted for modelling on paper.

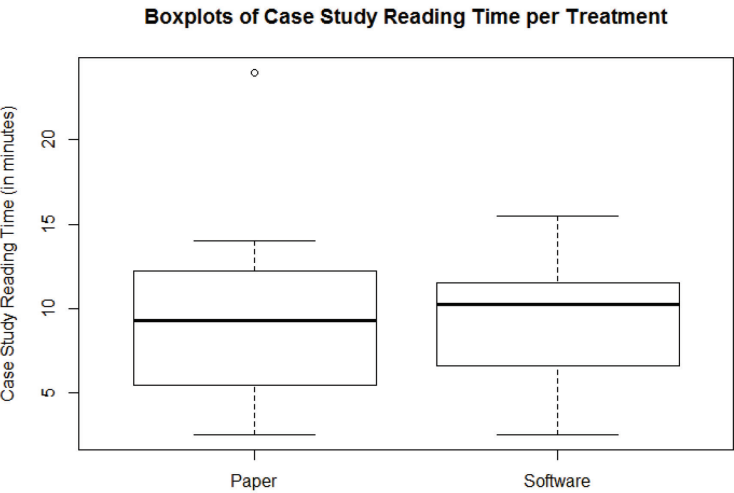


Figure 6-10: Boxplots of case study reading time per treatment.

The boxplot in Figure 6-11 shows the distribution of the time used by subjects for performing the modelling task in each treatment. Even when the median value for the modelling time used in each treatment is nearly the same, the distributions of the modelling times in the two treatments are very different. The range of values for modelling on paper is significantly wider than that for modelling with the modelling tool. These results indicate that modelling with the tool increased the modelling time of slow subjects but also decreased the modelling time of fast subjects.

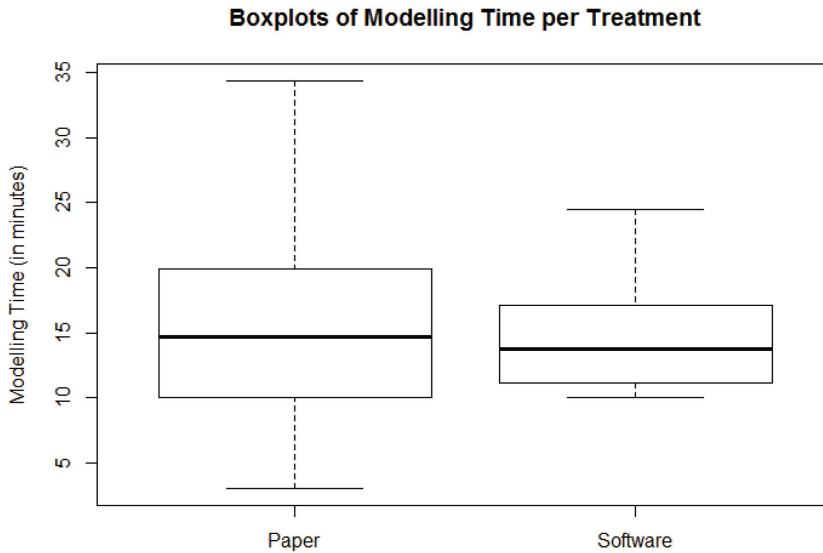


Figure 6-11: Boxplots of modelling time per treatment.

The boxplot in Figure 6-12 shows the distribution of the overall time used by subjects for performing the whole task of the experiment, which for each treatment consisted of reading the methodology, reading the case study, and performing the actual modelling. Even when the median value for the modelling time used in each treatment is nearly the same (30 minutes on paper and 30.5 minutes with the modelling tool), the distribution of the total experiment times of the two treatments is very different. The range of values for the overall experiment time is significantly wider for modelling on paper than for modelling with the modelling tool. These results indicate that modelling with the tool decreased the experiment time for slow subjects, while also increasing the experiment time for fast subjects.

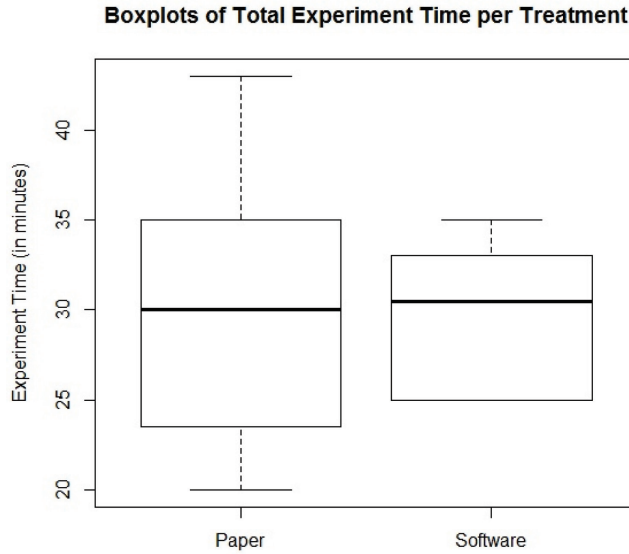


Figure 6-12: Boxplots of total experiment time per treatment.

6.5.4 Threats to validity

In this Section, we present four (4) threats to the validity of the results of the conducted experiment – conclusion validity, internal validity, construct validity, and external validity.

In this experiment, there is one major threat to the internal validity [154]. This empirical study was not conducted by professionals in the industrial environment. According to [146], empirical evaluation by professionals in the real environment embraces all of the complexities of human practice in real organisations, gives stronger internal validity, and assures a more rigorous assessment of the effectiveness of the artefact. However, the research results by [43,120] show that professionals and students perform similarly in empirical evaluations of software engineering artefacts, especially when they apply a new approach for the first time.

Concerning the external validity [154], it is highly probable that similar results will be obtained when running this experiment in a similar way with other subjects because the subjects of this experiment decided to register for the course of agent-oriented modelling of sociotechnical systems based on their interest in advanced software engineering and convenience. However, all the resources used in this experiment are publicly available in the experiment package to encourage repetition of the study.

The threat to conclusion validity [154] relates to the sample size during the empirical study which involved modelling of 8 sociotechnical systems. According to [74], a large sample size helps to statistically observe nearly any legitimate differences between experimental conditions. Moreover, a large sample size improves the quality of research contributions. However, the systematic review of 1,700 software engineering papers published from 2001 to 2011 [77] on controlled experiments of software engineering tools with human participants reports on a large range of participants from 1 to 2,600 (the latter was a field deployment) with a median of 10 participants. Therefore, the sample size during this empirical study is very close to the median sample size of similar empirical studies.

The construct validity [154] includes two major threats. The first threat to the construct validity is that the used metrics may not be appropriate ones for evaluating the effectiveness of the modelling guidelines. For example, is “the comparison between the number of entities produced by the AOM approach and the number of the resulting entities of CPN in CPN Tools” an appropriate metric for evaluating the effectiveness of the modelling guidelines? The second threat to the construct validity is that the experiment was conducted as a part of the course, where the students were graded. This implies that the students may bias their data, as they believe that it will give them better grades. However, in the beginning of the course it was emphasised that the grade did not depend on the actual data. The grade was instead based on the completeness of the requirements, proper delivery, and the understanding of the topics expressed in the reports that were handed in by students at the end of the course.

6.6 Summary

This Chapter provides the analysis and results of an empirical study that aimed to evaluate the effort and effectiveness of applying the goal modelling and domain knowledge modelling using a paper and pen in comparison with using the modelling tool developed for the AOM4STS methodology. Thus, two research questions were identified RQ3(b) and RQ3(c). The former states that, “What is the impact of the software tool on the effort needed for requirements and design modelling of sociotechnical systems employing the AOM4STS methodology?” while the latter states that, “What is the effectiveness of the software tool for requirements and design modelling of sociotechnical systems employing AOM4STS methodology?”

The empirical study involved experimental tasks of modelling requirements for sociotechnical systems and was completed by 8 post-graduate (MSc and PhD) students registered for the requirements engineering course at the University of Tartu. The subjects had created two requirements models – goal model and domain model – for two case studies, one of which modelled on paper and another one with the AOM4STS tool. The assessment results of experimental settings show that a short tutorial about goal modelling and domain knowledge modelling, and a demonstration of the newly developed modelling tool provided the subjects with sufficient knowledge to adequately perform the modelling tasks.

According the analysis results of the empirical data, the answer to RQ3(b) leads to the conclusion that the modelling effort on paper is nearly the same as the modelling effort with the AOM4STS software tool. However, the higher variance values for the modelling effort on paper dominate as compared with the modelling effort with the tool. An explanation for this is that the tool imposes more constraints on the requirements modelling activities. Moreover, as the answer to RQ3(c) leads to the conclusion that the effectiveness of modelling requirements with the modelling tool was higher than the effectiveness of modelling requirements on paper by considering information propagation, consistency checking, and visual cognition. However, goal decomposition activity was slightly more effective when modelled on paper compared to modelling with the tool. The answers to the research questions and particularly the answer to RQ3(c) allow us to conclude that the support by modelling tools is essential for engineering requirements for sociotechnical systems because for such systems requirements should be modelled at different abstraction levels and from different perspectives that should be consistent with each other.

However, we must remark that the results of this empirical study are limited to the medium of applying the AOM4STS methodology – i.e. on paper as opposed to using the modelling tool – and are therefore not generalizable to the AOM4STS methodology. Moreover, as the case studies PES and MSS had a specific focus on sociotechnical systems, the findings of this study cannot be generalised to any other kind of system. Furthermore, as the two case studies are small, scalability issues expected to arise in larger applications were not considered.

7 Conclusions, Limitations and Future Work

This Chapter provides the conclusions of the research work reported in this thesis, describes the limitations of the research contributions, and suggests new research directions that would further explore the key findings of the reported research work.

7.1 Research Summary and Conclusions

This Section summaries the answers to the research questions in Section 1.4 and concludes the thesis.

7.1.1 Guidelines for Modelling of Agent-Oriented Models in CPN Tools

This thesis proposes novel guidelines that support the mapping of design models of sociotechnical systems in CPN Tools for visual simulation, validation and verification. These guidelines are divided into the guidelines for CPN knowledge modelling, CPN interaction modelling and CPN behaviour modelling. Moreover, this thesis demonstrates the feasibility of the proposed CPN modelling guidelines by describing the development of the intruder handling system in CPN Tools.

Furthermore, this thesis analyses and discusses the results of an empirical study that evaluates the effectiveness of the proposed CPN modelling guidelines for modelling and simulating in CPN Tools the design models of sociotechnical systems produced by the AOM4STS methodology. The application of the proposed CPN modelling guidelines produces CPN models in CPN Tools with the same number of entities as the corresponding design modes produced by the AOM4STS methodology. Therefore, this study compares the number of entities in the design models produced by the AOM4STS methodology with the number of entities in the CPN models produced by applying the proposed CPN modelling guidelines to 8 different sociotechnical systems to evaluate the effectiveness of the proposed CPN modelling guidelines.

On the one hand, the analysis results show no significant difference between the number of agent types, rules and knowledge entities created by the AOM4STS methodology and the number of agent types, rules and knowledge entities produced in CPN Tools by applying the proposed CPN modelling guidelines. On the other hand, the analysis results show a significant difference between the number of interaction types produced by the AOM4STS methodology and the number of interaction types produced by applying the proposed CPN modelling guidelines. Therefore, these results lead to the conclusion that the proposed CPN modelling guidelines can effectively support the modelling of agents and their knowledge and behaviours in CPN Tools for various applications of sociotechnical systems. However, the results suggest the need for further study on the effectiveness of the CPN interaction modelling guidelines.

Moreover, the results of this empirical study show the utility of CPN Tools in supporting visual simulation, scenario-based validation, and state space verification of the design models of sociotechnical systems. The results of visual simulations demonstrate the capabilities of Message Sequence Charts (MSC) of CPN Tools for visualising behaviours by various agents and interactions between them in different applications of sociotechnical systems. In addition, the scenario-based validation of the resulting CPN models of sociotechnical systems affirms that business rules are central building blocks in scenario-based validation of design properties of sociotechnical systems. Finally, the results of the state space verification demonstrates how the CPN

formalisms in CPN Tools enhances quality of design models produced by the AOM4STS methodology by identifying unwanted states and activities of sociotechnical systems. These results lead to the conclusion that CPN Tools is practically useful for supporting visual simulation, scenario-based validation, and state space verification of the design models of sociotechnical systems. Subsequently, CPN Tools improves the quality of modelling artefacts for sociotechnical systems.

7.1.2 Guidelines for Prototyping of Agent-Oriented Models on the JADE Framework

This thesis proposes novel guidelines that support the development of design models of sociotechnical systems on the JADE framework. These guidelines are divided into guidelines for JADE-specific interaction development, behaviour development, and knowledge development. The latter involves the development and implementation of conceptual objects and the ontologies for sociotechnical systems on the JADE framework. Moreover, this thesis demonstrates the feasibility of the JADE prototyping guidelines by developing the intruder handling system on the JADE Framework.

Further, this thesis analyses and discusses the results of an empirical study that evaluates the effectiveness of the JADE guidelines together with the JADE website resources against the usage of the JADE website resources alone. This empirical study involved the development of 16 sociotechnical systems for different problem domains divided into two groups. Group 1 developed 8 prototypes using just the JADE website resources, while Group 2 developed 8 prototypes using the JADE guidelines together with the JADE website resources.

The results of this empirical study do not demonstrate a substantial difference between Group 1 and Group 2 in the development of agents and their interactions and behaviours on the JADE Framework. However, the results show that more prototypes in Group 2 (63%) implemented conceptual objects than in Group 1 (50%). Furthermore, the results express that 50% of the prototypes developed in Group 2 implemented the ontology that was used for sharing knowledge among the agents, while in Group 1, none of the prototypes implemented the ontology. Therefore, the results of the empirical study lead to the conclusion that the JADE guidelines together with the JADE website resources are more effective for the development of agent knowledge for sociotechnical systems on the JADE framework than using only the JADE website resources for the same purpose. Additionally, these empirical results affirm the research findings by [147], according to which conceptual objects are necessary building blocks for the development of ontologies.

7.1.3 Support by the AOM4STS Tool for Modelling Sociotechnical Systems

This thesis describes the support by the AOM4STS tool, which is a novel agent-oriented software tool that focuses on conceptual domain analysis and design modelling of sociotechnical systems by employing the AOM4STS methodology. Furthermore, this thesis uses the intruder detection case study to describe the key features of the AOM4STS tool that aim to reduce the modelling effort and improve the modelling effectiveness during the requirements elicitation and design of sociotechnical systems.

Furthermore, this thesis analyses and discusses the results of an empirical study that evaluates the impact of the AOM4STS tool on the effort needed for requirements and design modelling of sociotechnical systems employing the AOM4STS methodology. Moreover, this study evaluates the effectiveness of the AOM4STS tool for requirements and design modelling of sociotechnical systems employing the AOM4STS methodology.

This study involved experimental tasks of modelling requirements for sociotechnical systems and was completed by 8 subjects. The subjects created two requirements models – goal model and domain knowledge model – for two case studies, one of which was modelled on paper and the other one with the AOM4STS tool.

The assessment results of the experiment show that a short tutorial about goal modelling and domain knowledge modelling, and a demonstration of the newly developed modelling tool provided the subjects with sufficient knowledge to adequately perform the modelling tasks. Furthermore, the analysis results of this study lead to the conclusion that the modelling effort on paper is nearly the same as the modelling effort with the AOM4STS software tool. However, the higher variance values for the modelling effort on paper dominate as compared with the modelling effort with the tool. An explanation for this is that the tool imposes more constraints on the requirements modelling activities.

Moreover, the results of the empirical study lead to the conclusion that the effectiveness of modelling requirements with the modelling tool is higher than the effectiveness of modelling requirements on paper by considering information propagation, consistency checking, and visual cognition. However, goal decomposition activity is slightly more effective when modelling on paper compared to modelling with the tool. Generally, these findings conclude that the support by modelling tools is essential for engineering requirements for sociotechnical systems because such systems involve modelling of requirements at different abstraction levels and from different perspectives that should be consistent with each other.

7.2 Limitations of the Research

The results of this thesis contribute to the existing body of scientific knowledge described in Chapter 2. However, there is a boundary to these contributions. Therefore, this Section describes the limitations of this thesis.

There are two major limitations of this research work. First, the empirical evaluations of research contributions were not conducted by professionals in an industrial environment. Instead, the evaluations of research contributions were conducted by Master's and PhD students. According to [146], empirical evaluation by professionals in a real environment embraces all of the complexities of human practice in real organisations, gives stronger internal validity and assures a more rigorous assessment of the effectiveness of the artefact. This encourages the need for conducting an empirical evaluation by professionals in their professional work environment. However, the research results by [43,120] on the use of students and professionals as subjects in software engineering experiments show that professionals and students perform similarly in empirical evaluations of software engineering artefacts, especially when they apply a new approach for the first time.

The second limitation relates to the sample size during the evaluation of research contributions. In the empirical evaluation of the CPN modelling guidelines in Chapter 3, the participants conducted the modelling of 8 different sociotechnical systems in CPN Tools. In the empirical evaluation of the JADE prototyping guidelines in Chapter 4, the participants developed 16 different prototypes of sociotechnical systems with the JADE framework. Furthermore, the empirical evaluation of the support by the AOM4STS tool reported in Chapter 6 involved 8 participants. According to [74], a large sample size helps to statistically observe nearly any legitimate difference between experimental

conditions. Subsequently, a large sample size improves the quality of research contributions. Nevertheless, the systematic review of 1,700 software engineering papers published from 2001 until 2011 [77] on controlled experiments of software engineering tools with human participants finds a large range of participants, from 1 to 2,600 (the latter was a field deployment) with a median of 10 participants. Therefore, the sample sizes during empirical evaluations of the research contributions of this thesis are very close to the sample sizes in similar empirical evaluations.

7.3 Future Work

The research contributions reported in this thesis take forward the existing body of knowledge on agent-oriented modelling of sociotechnical systems by proposing the CPN modelling guidelines, JADE prototyping guidelines and support by the AOM4STS software tool. Nevertheless, these research contributions have limitations as has been described in Section 7.2. Therefore, this section establishes further research directions to improve the research contributions and also suggests research methods that enable to overcome the limitations reported in this thesis.

The use of students as participants remains a valid simplification of reality needed in laboratory contexts, which has been proven to be an effective way to advance software engineering theories and technologies [43]. Moreover, a major differentiating factor that affects the results of empirical studies is the experience levels of the subjects [120]. For example, a classroom setting can have students who possess industrial experience or industrial setting can have professionals who are novices to a particular software engineering method. Therefore, this thesis proposes a new research direction to provide another evaluation of the research contributions by using R3 – Characterization Scheme – and compare the results of the two evaluations. The R3 – Characterization Scheme – is a development in empirical software engineering that focuses on the characterization of the actual experience of a subject rather than using simplistic role-oriented labels such as student or professional [43]. R3 stands for Real, Relevant, and Recent experience. Real aims to determine the extent to which a subject has real experience. Relevant aims to determine the extent to which the real experience by a subject is relevant. Recent aims to determine the extent to which the recent experience by a subject is relevant.

Moreover, according to [77], the empirical studies in software engineering – in particular tool evaluations – are too difficult to conduct, and for this reason they sometimes do not lead to firm conclusions or negative results. Similarly, the majority of studies published in psychology lack the power of a statistical test, resulting in a confusing literature with apparently contradictory results [93]. The power of the statistical test indicates whether the given experimental setup is capable of detecting the effect under study. It is a function of sample size, population effect size and significance criteria [74]. Therefore, this thesis proposes a new research direction that aims to improve the quality of the research contributions by increasing the power of statistical tests.

Lastly, the research contributions of this thesis relate to three tools. First, a novel AOM4STS tool for requirements modelling and design of sociotechnical system. Second, the CPN Tools for visual simulation, scenario-based validation, and state space verification of the design models of sociotechnical systems produced by the AOM4STS

tool. Third, the JADE framework for prototyping sociotechnical systems after validation and verification of their design models in CPN Tools. Similarly to Multi-Paradigm Modelling (MPM) [145], these tools aim to address and integrate three orthogonal research directions – meta-modelling, model abstraction and multi-formalism. Meta-modelling is concerned with the description (models of models) of classes of models, which allows formalism specification. Model abstraction is concerned with the relationship between the models represented at different levels of abstraction. Multi-formalism modelling is concerned with the coupling of and transformation between models described by different formalisms [144]. Accordingly, this thesis proposes a further research direction towards specifying formalisms for the AOM4STS tool through meta-modelling and transforming models between the AOM4STS tool, CPN Tools and the JADE framework through multi-formalism modelling.

List of Figures

| | |
|--|----|
| Figure 1-1: A framework for design-science research in information systems adapted from [4]..... | 15 |
| Figure 2-1: CPN model of a resource allocation system and variable declarations adapted from [69]..... | 27 |
| Figure 2-2: Software architecture of a JADE framework [10]..... | 30 |
| Figure 3-1: Sample representation of components knowledge model components. | 36 |
| Figure 3-2: Sample representation of a conceptual object type in a knowledge model. ... | 37 |
| Figure 3-3: An agent sending a message | 38 |
| Figure 3-4: CPN representation of an agent sending a message..... | 40 |
| Figure 3-5: An agent receiving a message | 40 |
| Figure 3-6: CPN representation of an agent receiving a message..... | 41 |
| Figure 3-7: Agent initialisation | 42 |
| Figure 3-8: CPN representation of an agent initialisation | 43 |
| Figure 3-9: A composite main activity consisting of two sub-activities..... | 43 |
| Figure 3-10: CPN representation of an activity containing two sub-activities. | 44 |
| Figure 3-11: Reactive behaviour of an agent..... | 45 |
| Figure 3-12: CPN representation of a reactive behaviour by an agent | 46 |
| Figure 3-13: (a) Pre-conditional looping, (b) Post-conditional looping. | 47 |
| Figure 3-14: CPN representation of pre-conditional looping | 48 |
| Figure 3-15: CPN representation of post-conditional looping..... | 49 |
| Figure 3-16: Rule-based activity. | 50 |
| Figure 3-17: CPN representation of a rule-based activity | 50 |
| Figure 3-18: An agent passing knowledge between activities..... | 51 |
| Figure 3-19: CPN representation of knowledge passing between activities. | 52 |
| Figure 3-20: Goal model of the intruder handling system [135] | 54 |
| Figure 3-21: Knowledge model of the intruder handling system | 55 |
| Figure 3-22: Combined interaction model and agent behaviour models of the intruder handling system..... | 56 |
| Figure 3-23: Representation of the agent knowledge model in CPN Tools..... | 58 |
| Figure 3-24: Transformation guidelines identified in the combined agent interaction and behaviour models of the intruder handling system | 60 |
| Figure 3-25: CPN model corresponding to the agent-oriented models of the intruder handling system..... | 61 |
| Figure 3-26: Functions that support to visualise the behaviour of intruder handling system | 62 |
| Figure 3-27: Visualisation of Scenario 1 of the intruder handling system by CPN Tools ... | 65 |
| Figure 3-28: Visualisation of Scenario 2 of the intruder handling system by CPN Tools ... | 67 |
| Figure 3-29: Visualisation of Scenario 3 of the intruder handling system by CPN Tools ... | 69 |
| Figure 3-30: Frequency distribution of agent types in the selected case studies | 75 |
| Figure 3-31: Frequency distribution of action event types in the selected case studies... | 75 |
| Figure 3-32: Scatter plot charts comparing the numbers of AOM4STS and CPN entities. 76 | |
| Figure 3-33: Visual simulation of the project with ID 5 | 78 |
| Figure 4-1: The goal model for the intruder handling scenario [128]. | 86 |
| Figure 4-2: The goal model for communication [128]. | 87 |
| Figure 4-3: Domain model for intruder handling [128]. | 88 |
| Figure 4-4: Knowledge model of intruder handling [128]. | 89 |

| | |
|--|-----|
| Figure 4-5: Interaction sequence diagram of intruder handling [128]. | 90 |
| Figure 4-6: The behaviour model of intruder handling [128]. | 91 |
| Figure 4-7: JADE-specific knowledge model for the knowledge entity type PersonDescription [128]. | 92 |
| Figure 4-8: Coding of object detection by OpenCV and sending of the corresponding intruder-handling event to the SecurityAgent [128]. | 92 |
| Figure 4-9: Intruder Handling ontology for supporting JADE-specific interaction model [128]. | 93 |
| Figure 4-10: JADE-specific behaviour model of responding to intruder-handling by the SecurityManagerAgent [128]. | 94 |
| Figure 4-11: Implementation of the intruder handling system [128]. | 95 |
| Figure 4-12: Interactions between the securityManagerAgent, visitorAgent, familyAgent, and RelaOfficerAgent [128]. | 95 |
| Figure 4-13: Interactions of the policeManagerAgent with the respective policeAgent [128]. | 96 |
| Figure 4-14: Mode comparison of prototypes implemented in Group 1 and Group 2 [89]. | 102 |
| Figure 4-15: Types of conceptual objects and ontology used in prototypes of Group 1 and Group 2 [89]. | 103 |
| Figure 5-1: XML representation of represents requirements models of AOM4STS methodology. | 107 |
| Figure 5-2: The AOM4STS tool on (a) Google Chrome (b) Mozilla Firefox and (c) Internet Explorer web browser. | 107 |
| Figure 5-3: Modelling notations captured in the AOM4STS tool [88]. | 108 |
| Figure 5-4: Information propagated from goal model to role model in the AOM4STS tool. | 109 |
| Figure 5-5: A syntactical error message displayed in the activity logs' frame. | 110 |
| Figure 6-1: Distribution of the subjects with respect to their experience in computing. | 117 |
| Figure 6-2: Distribution of the subjects with respect to their experience in requirements analysis. | 118 |
| Figure 6-3: Distribution of the subjects with respect to their prior experience in agent-oriented modelling. | 118 |
| Figure 6-4: Mean values for comparing modelling effort of the two treatments [88]. | 120 |
| Figure 6-5: Variance values for comparing modelling effort of the two treatments [88]. | 121 |
| Figure 6-6: (a) Mean and (b) Variance for comparing modelling effectiveness of the two treatments [88]. | 122 |
| Figure 6-7: Boxplot on the effectiveness of the modelling tool with respect to information propagation, consistency checking, and visual cognition [88]. | 122 |
| Figure 6-8: Comparison of the average time spent on different activities. | 123 |
| Figure 6-9: Boxplots of methodology reading time per treatment. | 124 |
| Figure 6-10: Boxplots of case study reading time per treatment. | 124 |
| Figure 6-11: Boxplots of modelling time per treatment. | 125 |
| Figure 6-12: Boxplots of total experiment time per treatment. | 126 |

List of Tables

| | |
|--|-----|
| Table 2-1: A list of AOSE methodologies. | 22 |
| Table 2-2: Summary of models for conceptual domain analysis..... | 23 |
| Table 2-3: Summary of models for platform-independent design | 24 |
| Table 2-4: A list of case studies that apply AOM4STS methodology | 25 |
| Table 3-1: Behavioural interface model of the intruder handling system..... | 57 |
| Table 3-2: Description of the CPN variables of the intruder handling system | 59 |
| Table 3-3: Summary of validation results for Scenario 1 of simulating the intruder handling system..... | 64 |
| Table 3-4: Summary of validation results for the Scenario 2 of simulating the intruder handling system..... | 66 |
| Table 3-5: Summary of validation results for Scenario 3 of simulating the intruder handling system..... | 68 |
| Table 3-6: Results from the paired and two-tailed t-tests | 77 |
| Table 3-7: Results of scenario-based validation | 79 |
| Table 3-8: Results of state space verification. | 80 |
| Table 4-1: Elicitation questions and answers for intruder handling..... | 84 |
| Table 4-2: Research Sub-Questions and Themes [33]. | 99 |
| Table 4-3: Median and Mean comparison of prototypes implemented in Group 1 and Group 2 [33]. | 102 |
| Table 4-4: Conceptual object types and ontologies developed in Group 2 [33]. | 103 |
| Table 5-1: The viewpoint framework adapted from [13] | 106 |
| Table 6-1: Assignment of objects and treatments to subjects in the two days of the study..... | 113 |
| Table 6-2: A set of the questions in the questionnaires, with answers on a 1 . . . 5 Likert scale [34]. | 115 |
| Table 6-3: Questions about the time spent on the activities in each experiment [34]. .. | 116 |
| Table 6-4: Results of the adequateness of the subjects [34]..... | 119 |
| Table 6-5: Results of the adequateness of the objects [34]. | 119 |

References

- [1] J.-R. Abrial, M. Butler, S. Hallerstede, T.S. Hoang, F. Mehta, L. Voisin, Rodin: an open toolset for modelling and reasoning in Event-B, *Int. J. Softw. Tools Technol. Transf.* 12 (2010) 447–466.
- [2] Y. Abushark, T. Miller, J. Thangarajah, M. Winikoff, J. Harland, Requirements specification via activity diagrams for agent-based systems, *Auton. Agents Multi-Agent Syst.* 31 (2017) 423–468.
- [3] G.A. Agha, *Actors: A model of concurrent computation in distributed systems.*, DTIC Document, 1985.
- [4] R.H. von Alan, S.T. March, J. Park, S. Ram, Design science in information systems research, *MIS Q.* 28 (2004) 75–105.
- [5] C. Anirudh, Development of multi-agent system for fire accident detection using gaia methodology, *Int. J. Comput. Sci. Inf. Secur.* 8 (2010) 190–194.
- [6] E.R. Babbie, *The basics of social research*, Cengage Learning, 2013.
- [7] R. Badham, C. Clegg, T. Wall, *Socio-technical theory*, *Handb. Ergon.* N. Y. NY John Wiley. (2000).
- [8] G. Baxter, I. Sommerville, Socio-technical systems: From design methods to systems engineering, *Interact. Comput.* 23 (2011) 4–17.
- [9] F. Bellifemine, A. Poggi, G. Rimassa, Developing multi-agent systems with a FIPA-compliant agent framework, *Softw.-Pract. Exp.* 31 (2001) 103–128.
- [10] F.L. Bellifemine, G. Caire, D. Greenwood, *Developing multi-agent systems with JADE*, John Wiley & Sons, 2007.
- [11] F.L. Bellifemine, G. Caire, D. Greenwood, *Developing multi-agent systems with JADE*, John Wiley & Sons, 2007.
- [12] K.M. Benner, M.S. Feather, W.L. Johnson, L.A. Zorman, Utilizing scenarios in the software development process, *Inf. Syst. Dev. Process.* 30 (2014) 117–134.
- [13] C. Bernon, V. Camps, M.-P. Gleizes, G. Picard, Engineering adaptive multi-agent systems: The adelfe methodology, *Agent-Oriented Methodol.* (2005) 172–202.
- [14] E. Best, H. Wimmel, Structure theory of Petri nets, in: *Trans. Petri Nets Models Concurr.* VII, Springer, 2013: pp. 162–224.
- [15] W.R. Bischofberger, G. Pomberger, *Prototyping-oriented software development: Concepts and tools*, Springer Science & Business Media, 2012.
- [16] M. Brambilla, J. Cabot, M. Wimmer, Model-driven software engineering in practice, *Synth. Lect. Softw. Eng.* 1 (2012) 1–182.
- [17] T. Bray, J. Paoli, C.M. Sperberg-McQueen, E. Maler, F. Yergeau, *Extensible markup language (XML) 1.0, W3C recommendation*, 2008.
- [18] P. Bresciani, A. Perini, P. Giorgini, F. Giunchiglia, J. Mylopoulos, Tropos: An agent-oriented software development methodology, *Auton. Agents Multi-Agent Syst.* 8 (2004) 203–236.
- [19] G. Caire, W. Coulier, F. Garijo, J. Gomez, J. Pavón, F. Leal, P. Chainho, P. Kearney, J. Stark, R. Evans, others, Agent oriented analysis using MESSAGE/UML, in: *Agent-Oriented Softw. Eng. II*, Springer, 2002: pp. 119–135.
- [20] P. Carayon, Human factors of complex sociotechnical systems, *Appl. Ergon.* 37 (2006) 525–535.

- [21] P. Carayon, P. Hancock, N. Leveson, I. Noy, L. Sznelwar, G. Van Hootehem, Advancing a sociotechnical systems approach to workplace safety—developing the conceptual framework, *Ergonomics*. 58 (2015) 548–564.
- [22] J.C. Carver, E. Syriani, J. Gray, Assessing the Frequency of Empirical Evaluation in Software Modeling Research., *EESMod*. 785 (2011).
- [23] R.K. Chatterjee, A. Sarkar, S. Bhattacharya, Modeling and analysis of agent oriented system: Petri-net based approach, in: 11th Int. Conf. Softw. Eng. Res. Pract. SERP 11, Citeseer, 2011: pp. 17–23.
- [24] M.R. Chaudron, M. Genero, S. Abrahão, P. Mohagheghi, L. Pareto, Summary of the first international workshop on experiences and empirical studies in software modelling, in: *Int. Conf. Model Driven Eng. Lang. Syst.*, Springer, 2011: pp. 119–122.
- [25] A. Chavez, P. Maes, Kasbah: An agent marketplace for buying and selling goods, in: *Proc. First Int. Conf. Pract. Appl. Intell. Agents Multi-Agent Technol.*, London, UK, 1996.
- [26] W. Cheah, A.B. Masli, E. Mit, Sustainability Modelling of e-Commerce for Rural Community: A Case from Long Lamai e-Commerce Initiative, in: *Inform. Creat. Multimed. ICICM 2013 Int. Conf. On, IEEE*, 2013: pp. 282–287.
- [27] W. Cheah, L. Sterling, K. Taveter, Task knowledge patterns reuse in multi-agent systems development, in: *Int. Conf. Princ. Pract. Multi-Agent Syst.*, Springer, 2010: pp. 459–474.
- [28] W.S. Cheah, E. Mit, M.V.A.M.V. AiSiok, Designing a Shared Single Display Education Application through Interactive Patterns, *J. Softw. Eng. Appl.* 7 (2014) 1074.
- [29] L. CheeWyai, W. Cheah, A.K. Chowdhury, C. Gulden, Engineering sustainable software: A case study from offline computer support collaborative annotation system, in: *Softw. Eng. Conf. MySEC 2015 9th Malays.*, IEEE, 2015: pp. 272–277.
- [30] S. Choi, H. Kim, E. Byun, C. Hwang, M. Baik, Reliable asynchronous message delivery for mobile agents, *IEEE Internet Comput.* 10 (2006) 16–25.
- [31] V.P. da Conceição, J. Dahlman, A. Navarro, What is maritime navigation? Unfolding the complexity of a Sociotechnical System, in: *Proc. Hum. Factors Ergon. Soc. Annu. Meet.*, SAGE Publications Sage CA: Los Angeles, CA, 2017: pp. 267–271.
- [32] M. Cossentino, From requirements to code with the PASSI methodology, *Agent-Oriented Methodol.* 3690 (2005) 79–106.
- [33] F. Dalpiaz, E. Paja, P. Giorgini, Security requirements engineering: designing secure socio-technical systems, MIT Press, 2016.
- [34] H.K. Dam, M. Winikoff, Towards a next-generation AOSE methodology, *Sci. Comput. Program.* 78 (2013) 684–694.
- [35] K.H. Dam, M. Winikoff, Comparing agent-oriented methodologies, in: *Agent-Oriented Inf. Syst.*, Springer, 2004: pp. 78–93.
- [36] P. Davidsson, Multi agent based simulation: beyond social simulation, in: *Int. Workshop Multi-Agent Syst. Agent-Based Simul.*, Springer, 2000: pp. 97–107.
- [37] M.C. Davis, R. Challenger, D.N. Jayewardene, C.W. Clegg, Advancing socio-technical systems thinking: A call for bravery, *Appl. Ergon.* 45 (2014) 171–180.

- [38] P.-E. Dossou, P. Pawlewski, P. Mitchell, The use of Multi-agent Systems for Improving a Logistic Platform in a GRAI Environment, in: *Int. Conf. Pract. Appl. Agents Multi-Agent Syst.*, Springer, 2015: pp. 126–135.
- [39] H. Du, K. Taveter, M.N. Huhns, Simulating a Societal Information System for Healthcare, in: *Comput. Sci. Inf. Syst. FedCSIS 2012 Fed. Conf. On, IEEE*, 2012: pp. 1239–1246.
- [40] K. Eason, Before the internet: the relevance of socio-technical systems theory to emerging forms of virtual organisation, *Int. J. Sociotechnology Knowl. Dev. IJSKD*. 1 (2009) 23–32.
- [41] T. Edgington, B. Choi, K. Henson, T.S. Raghu, A. Vinze, Adopting ontology to facilitate knowledge sharing, *Commun. ACM*. 47 (2004) 85–90.
- [42] N. Ejaz, U. Manzoor, S. Nefti, S.W. Baik, A collaborative multi-agent framework for abnormal activity detection in crowded areas, *Int J Innov Comput Inf. Control*. 8 (2012) 4219–4234.
- [43] D. Falessi, N. Juristo, C. Wohlin, B. Turhan, J. Münch, A. Jedlitschka, M. Oivo, Empirical software engineering experts on the use of students and professionals in experiments, *Empir. Softw. Eng.* (2017) 1–38.
- [44] FIPA, FIPA ACL Message Structure Specification, Found. Intell. Phys. Agents [Httpwww Fipa Orgspecsipa00061SC00061G Html](http://www.fipa.org/specs/fipa00061SC00061G.html) 306 2004. (2002).
- [45] S. Franklin, A. Graesser, Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents, in: *Intell. Agents III Agent Theor. Archit. Lang.*, Springer, 1997: pp. 21–35.
- [46] A. Freitas, L. Hilgert, S. Marczak, F. Meneguzzi, R.H. Bordini, R. Vieira, A multi-agent systems engineering tool based on ontologies, in: *34th Int. Conf. Concept. Model. Stockh. Swed. Ser Lect. Notes Comput. Sci. Springer*, 2015.
- [47] R. Fuentes-Fernández, I. García-Magariño, A.M. Gómez-Rodríguez, J.C. González-Moreno, A technique for defining agent-oriented engineering processes with tool support, *Eng. Appl. Artif. Intell.* 23 (2010) 432–444.
- [48] S. Gao, D. Xu, Conceptual modeling and development of an intelligent agent-assisted decision support system for anti-money laundering, *Expert Syst. Appl.* 36 (2009) 1493–1504.
- [49] J.C. Garcia-Ojeda, S.A. DeLoach, others, agentTool process editor: supporting the design of tailored agent-based processes, in: *Proc. 2009 ACM Symp. Appl. Comput.*, ACM, 2009: pp. 707–714.
- [50] J.C. Garcia-Ojeda, S.A. DeLoach, W.H. Oyenar, J. Valenzuela, others, O-MaSE: a customizable approach to developing multiagent development processes, Springer, 2008.
- [51] J.M. Gascueña, A. Fernández-Caballero, M.T. López, A.E. Delgado, Knowledge modeling through computational agents: application to surveillance systems, *Expert Syst.* 28 (2011) 306–323.
- [52] F.D. Giraldo, S. España, Ó. Pastor, Evidences of the mismatch between industry and academy on modelling language quality evaluation, *ArXiv Prepr. ArXiv160602025*. (2016).
- [53] T. Gorschek, E. Tempero, L. Angelis, On the use of software design models in software development practice: An empirical investigation, *J. Syst. Softw.* 95 (2014) 176–193.
- [54] A. Greasley, *Simulation modelling for business*, Routledge, 2017.

- [55] D. Harel, P.S. Thiagarajan, Message sequence charts, in: *UML Real*, Springer, 2003: pp. 77–105.
- [56] B. Hayes-Roth, An architecture for adaptive intelligent systems, *Artif. Intell.* 72 (1995) 329–365.
- [57] B. Hayes-Roth, L. Brownston, R. van Gent, Multiagent Collaboration in Directed Improvisation., in: *ICMAS*, 1995: pp. 148–154.
- [58] B. Hayes-Roth, R. Washington, R. Hewett, M. Hewett, A. Seiver, Intelligent Monitoring and Control., in: *IJCAI*, 1989: pp. 243–249.
- [59] C. Hewitt, H. Baker, Actors and continuous functionals, (1977).
- [60] H. Hsu, P.A. Lachenbruch, Paired t test, *Wiley Encycl. Clin. Trials.* (2008).
- [61] J. Huang, N.R. Jennings, J. Fox, Agent-based approach to health care management, *Appl. Artif. Intell. Int. J.* 9 (1995) 401–420.
- [62] L. Iocchi, D. Nardi, M. Salerno, Reactivity and deliberation: a survey on multi-robot systems, in: *Balanc. React. Soc. Deliberation Multi-Agent Syst.*, Springer, 2001: pp. 9–32.
- [63] W. Iqbal, S. Yousaf, Formal modeling of agent based cloud computing services using petri nets, *VFAST Trans. Softw. Eng.* 1 (2013) 1–6.
- [64] N.R. Jennings, The ARCHON system and its applications, (1994).
- [65] N.R. Jennings, On agent-based software engineering, *Artif. Intell.* 117 (2000) 277–296.
- [66] N.R. Jennings, An agent-based approach for building complex software systems, *Commun. ACM.* 44 (2001) 35–41.
- [67] N.R. Jennings, P. Faratin, M.J. Johnson, T.J. Norman, P. O'brien, M.E. Wiegand, Agent-based business process management, *Int. J. Coop. Inf. Syst.* 5 (1996) 105–130.
- [68] N.R. Jennings, M. Wooldridge, Applications of intelligent agents, in: *Agent Technol.*, Springer, 1998: pp. 3–28.
- [69] K. Jensen, *Coloured Petri nets: basic concepts, analysis methods and practical use*, Springer Science & Business Media, 2013.
- [70] K. Jensen, L.M. Kristensen, Colored Petri nets: a graphical language for formal modeling and validation of concurrent systems, *Commun. ACM.* 58 (2015) 61–70.
- [71] K. Jensen, L.M. Kristensen, L. Wells, Coloured Petri Nets and CPN Tools for modelling and validation of concurrent systems, *Int. J. Softw. Tools Technol. Transf.* 9 (2007) 213–254.
- [72] A. Jones, C. Moulin, J. Barthes, D. Lenne, A. Kendira, T. Gidel, Personal assistant agents and multi-agent middleware for cscw, in: *Comput. Support. Coop. Work Des. CSCWD 2012 IEEE 16th Int. Conf. On*, IEEE, 2012: pp. 72–79.
- [73] T. Juan, A. Pearce, L. Sterling, ROADMAP: extending the gaia methodology for complex open systems, in: *Proc. First Int. Jt. Conf. Auton. Agents Multiagent Syst. Part 1*, ACM, 2002: pp. 3–10.
- [74] M. Kaptein, J. Robertson, Rethinking statistical analysis methods for CHI, in: *Proc. SIGCHI Conf. Hum. Factors Comput. Syst.*, ACM, 2012: pp. 1105–1114.
- [75] T.K. Kim, T test as a parametric statistic, *Korean J. Anesthesiol.* 68 (2015) 540–546.
- [76] D. Kinny, M. Georgeff, A. Rao, A methodology and modelling technique for systems of BDI agents, in: *Agents Break. Away*, Springer, 1996: pp. 56–71.

- [77] A.J. Ko, T.D. Latoza, M.M. Burnett, A practical guide to controlled experiments of software engineering tools with human participants, *Empir. Softw. Eng.* 20 (2015) 110–141.
- [78] P. Koutsabasis, J. Darzentas, Methodologies for agent systems development: underlying assumptions and implications for design, *AI Soc.* 23 (2009) 379–407.
- [79] K. Kravari, N. Bassiliades, A survey of agent platforms, *J. Artif. Soc. Soc. Simul.* 18 (2015) 11.
- [80] X. Liu, K. Wang, Active Power Control Simulation Platform Research of Wind Farm Based on Multi-Agent, in: *MATEC Web Conf.*, EDP Sciences, 2015.
- [81] R. Lock, I. Sommerville, Modelling and analysis of socio-technical system of systems, in: *Eng. Complex Comput. Syst. ICECCS 2010 15th IEEE Int. Conf. On*, IEEE, 2010: pp. 224–232.
- [82] P. Maes, Artificial life meets entertainment: lifelike autonomous agents, *Commun. ACM.* 38 (1995) 108–114.
- [83] P. Maes, others, Agents that reduce work and information overload, *Commun. ACM.* 37 (1994) 30–40.
- [84] M. Maguire, Socio-technical systems and interaction design–21st century relevance, *Appl. Ergon.* 45 (2014) 162–170.
- [85] M. Mahunnah, A. Koorts, K. Taveter, Towards Distributed Sociotechnical System for Reporting Critical Laboratory Results., in: *HEALTHINF*, 2013: pp. 269–276.
- [86] M. Mahunnah, A. Norta, L. Ma, K. Taveter, Heuristics for Designing and Evaluating Socio-technical Agent-Oriented Behaviour Models with Coloured Petri Nets, in: *Comput. Softw. Appl. Conf. Workshop COMPSACW 2014 IEEE 38th Int.*, IEEE, 2014: pp. 438–443.
- [87] M. Mahunnah, K. Taveter, A scalable multi-agent architecture in environments with limited connectivity: Case study on individualised care for healthy pregnancy, in: *Digit. Ecosyst. Technol. DEST 2013 7th IEEE Int. Conf. On*, IEEE, 2013: pp. 84–89.
- [88] M. Mahunnah, K. Taveter, R. Matulevicius, An Empirical Evaluation of the Requirements Engineering Tool for Sociotechnical Systems., in: *26th IEEE Int. Requir. Eng. Conf. Workshop REW*, IEEE, 2018.
- [89] M. Mahunnah, K. Taveter, C. WaiShiang, S. WaiYee, An Empirical Evaluation of Guidelines for Prototyping Sociotechnical Systems in JADE Framework., in: *3rd IEEE Int. Symp. Agents Multi-Agent Syst. Robot.*, IEEE, 2018.
- [90] U. Manzoor, B. Zafar, Multi-Agent Modeling Toolkit–MAMT, *Simul. Model. Pract. Theory.* 49 (2014) 215–227.
- [91] J. Marshall, Agent-based modelling of emotional goals in digital media design projects, in: *Innov. Methods User-Friendly Tools Coding Des. Approaches People-Oriented Program.*, IGI Global, 2018: pp. 262–284.
- [92] M. Marsilio, A. Torbica, S. Villa, Health care multidisciplinary teams: The sociotechnical approach for an integrated system-wide perspective, *Health Care Manage. Rev.* 42 (2017) 303–314.
- [93] S.E. Maxwell, The persistence of underpowered studies in psychological research: causes, consequences, and remedies., *Psychol. Methods.* 9 (2004) 147.
- [94] T. Miller, S. Pedell, A.A. Lopez-Lorca, A. Mendoza, L. Sterling, A. Keirnan, Emotion-led modelling for people-oriented requirements engineering: The case study of emergency systems, *J. Syst. Softw.* 105 (2015) 54–71.

- [95] P. Mohagheghi, W. Gilani, A. Stefanescu, M.A. Fernandez, An empirical study of the state of the practice and acceptance of model-driven engineering in four industrial cases, *Empir. Softw. Eng.* 18 (2013) 89–116.
- [96] M. Morandini, F. Dalpiaz, C.D. Nguyen, A. Siena, The tropos software engineering methodology, in: *Handb. Agent-Oriented Des. Process.*, Springer, 2014: pp. 463–490.
- [97] T. Murata, Petri nets: Properties, analysis and applications, *Proc. IEEE.* 77 (1989) 541–580.
- [98] G. Mussbacher, D. Amyot, R. Breu, J.-M. Buel, B.H. Cheng, P. Collet, B. Combemale, R.B. France, R. Heldal, J. Hill, The relevance of model-driven engineering thirty years from now, in: *Int. Conf. Model Driven Eng. Lang. Syst.*, Springer, 2014: pp. 183–200.
- [99] K. Myers, P. Berry, J. Blythe, K. Conley, M. Gervasio, D.L. McGuinness, D. Morley, A. Pfeffer, M. Pollack, M. Tambe, An intelligent personal assistant for task and time management, *AI Mag.* 28 (2007) 47.
- [100] N.C. Narendra, A. Norta, M. Mahunnah, F. Maggi, Modelling Sound Conflict Management for Virtual-Enterprise Collaboration, in: *Serv. Comput. SCC 2014 IEEE Int. Conf. On, IEEE*, 2014: pp. 813–820.
- [101] A. Norta, M. Mahunnah, T. Tenso, K. Taveter, N.C. Narendra, An Agent-Oriented Method for Designing Large Socio-technical Service-Ecosystems, in: *Serv. Serv. 2014 IEEE World Congr. On, IEEE*, 2014: pp. 242–249.
- [102] I. Nunes, C.J. De Lucena, D. Cowan, U. Kulesza, P. Alencar, C. Nunes, Developing multi-agent system product lines: from requirements to code, *Int. J. Agent-Oriented Softw. Eng.* 4 (2011) 353–389.
- [103] H.S. Nwana, Software agents: An overview, *Knowl. Eng. Rev.* 11 (1996) 205–244.
- [104] C.I. Nyulas, M.J. O'Connor, S.W. Tu, D.L. Buckeridge, A. Okhmatovskaia, M.A. Musen, An ontology-driven framework for deploying jade agent systems, in: *Proc. 2008 IEEEWICACM Int. Conf. Web Intell. Intell. Agent Technol.-Vol. 02, IEEE Computer Society*, 2008: pp. 573–577.
- [105] A. Oluyomi, S. Karunasekera, L. Sterling, Description templates for agent-oriented patterns, *J. Syst. Softw.* 81 (2008) 20–36.
- [106] L. Padgham, M. Winikoff, Prometheus: A methodology for developing intelligent agents, in: *Agent-Oriented Softw. Eng. III, Springer*, 2003: pp. 174–185.
- [107] E. Paja, F. Dalpiaz, P. Giorgini, STS-tool: Security requirements engineering for socio-technical systems, in: *Eng. Secure Future Internet Serv. Syst.*, Springer, 2014: pp. 65–96.
- [108] P. Papapanagiotou, J.D. Fleuriot, A theorem proving framework for the formal verification of web services composition, *ArXiv Prepr. ArXiv11082348.* (2011).
- [109] H.V.D. Parunak, Manufacturing experience with the contract net, *Distrib. Artif. Intell.* 1 (1987) 285–310.
- [110] J. Pavón, J.J. Gómez-Sanz, R. Fuentes, The INGENIAS methodology and tools, *Agent-Oriented Methodol.* 9 (2005) 236–276.
- [111] S. Pedell, T. Miller, L. Sterling, F. Vetere, S. Howard, Substantiating agent-based quality goals for understanding socio-technical systems, in: *Adv. Agent Technol.*, Springer, 2012: pp. 80–95.

- [112] M. Pipattanasomporn, H. Feroze, S. Rahman, Multi-agent systems in a distributed smart grid: Design and implementation, in: Power Syst. Conf. Expo. 2009 PSCE09 IEEE PES, IEEE, 2009: pp. 1–8.
- [113] J.L. Posadas, J.L. Poza, J.E. Simó, G. Benet, F. Blanes, Agent-based distributed architecture for mobile robot control, Eng. Appl. Artif. Intell. 21 (2008) 805–823.
- [114] K. Qayumi, A. Norta, Business-Intelligence Mining of Large Decentralized Multimedia Datasets with a Distributed Multi-Agent System, World Acad. Sci. Eng. Technol. Int. J. Comput. Electr. Autom. Control Inf. Eng. 10 (2016) 1160–1169.
- [115] I. Rahwan, T. Juan, L. Sterling, Integrating social modelling and agent interaction through goal-oriented analysis, Int. J. Comput. Syst. Sci. Eng. 21 (2006) 87–98.
- [116] W. Reisig, Petri nets: an introduction, Springer Science & Business Media, 2012.
- [117] Z. Ren, C.J. Anumba, Multi-agent systems in construction—state of the art and prospects, Autom. Constr. 13 (2004) 421–434.
- [118] S. Russell, P. Norvig, A. Intelligence, A modern approach, Artif. Intell. Prentice-Hall Englewood Cliffs. 25 (1995).
- [119] D. Rwegasira, I.B. Dhaou, A. Anagnostou, A. Kondoro, N. Shililiandumi, A. Kelati, S.J. Taylor, N. Mvungi, H. Tenhunen, A framework for load shedding and demand response in DC microgrid using multi agent system, in: Open Innov. Assoc. FRUCT 2017 21st Conf. Of, IEEE, 2017: pp. 284–289.
- [120] I. Salman, A.T. Misirli, N. Juristo, Are students representatives of professionals in software engineering experiments?, in: Proc. 37th Int. Conf. Softw. Eng.-Vol. 1, IEEE Press, 2015: pp. 666–676.
- [121] J.A. Sánchez, C.A. López, J.L. Schnase, An agent-based approach to the construction of floristic digital libraries, in: Proc. Third ACM Conf. Digit. Libr., ACM, 1998: pp. 210–216.
- [122] A. Santi, M. Guidi, A. Ricci, Jaca-android: An agent-based platform for building smart mobile applications, in: Lang. Methodol. Dev. Tools Multi-Agent Syst., Springer, 2011: pp. 95–114.
- [123] A. Sapožnikov, Design and Development of a Graphical Tool for Agent-Oriented Modelling. MSc thesis (in Estonian)., (2016).
- [124] R.G. Sargent, Verification and validation of simulation models, J. Simul. 7 (2013) 12–24.
- [125] A. Serenko, B. Detlor, Intelligent agents as innovations, Ai Soc. 18 (2004) 364–381.
- [126] C.W. Shiang, A.A. Halin, M. Lu, G. CheeWhye, Long Lamai Community ICT4D E-Commerce System Modelling: An Agent Oriented Role-Based Approach, Electron. J. Inf. Syst. Dev. Ctries. 75 (2016) 1–22.
- [127] C.W. Shiang, N. Kulathuramaiyer, S.W. Loke, Software Agent Negotiation Development: An Experience Report, in: Intell. Syst. Des. Appl. 2006 ISDA06 Sixth Int. Conf. On, IEEE, 2006: pp. 881–886.
- [128] C.W. Shiang, B.T. Onn, F.S. Tee, M.A. bin Khairuddin, M. Mahunnah, Developing Agent-Oriented Video Surveillance System through Agent-Oriented Methodology (AOM), CIT J. Comput. Inf. Technol. 24 (2016) 349–368.
- [129] I. Shvartsman, K. Taveter, Agent-oriented knowledge elicitation for modeling the winning of “hearts and minds,” in: Comput. Sci. Inf. Syst. FedCSIS 2011 Fed. Conf. On, IEEE, 2011: pp. 605–608.

- [130] I. Shvartsman, K. Taveter, From agent-oriented models to profile driven military training scenarios, in: *Intell. Distrib. Comput. VII*, Springer, 2014: pp. 317–322.
- [131] P.-O. Siebers, C.M. Macal, J. Garnett, D. Buxton, M. Pidd, Discrete-event simulation is dead, long live agent-based simulation!, *J. Simul.* 4 (2010) 204–210.
- [132] I. Sommerville, D. Cliff, R. Calinescu, J. Keen, T. Kelly, M. Kwiatkowska, J. Mcdermid, R. Paige, Large-scale complex IT systems, *Commun. ACM.* 55 (2012) 71–77.
- [133] L. Sterling, K. Taveter, Building agent-based appliances with complementary methodologies, in: *Proc. 2006 Conf. Knowl.-Based Softw. Eng. Proc. Seventh Jt. Conf. Knowl.-Based Softw. Eng.*, IOS Press, 2006: pp. 223–232.
- [134] L. Sterling, K. Taveter, others, Building Agent-Based Appliances with Complementary Methodologies, in: *Proc. 2006 Conf. Knowl.-Based Softw. Eng. Proc. Seventh Jt. Conf. Knowl.-Based Softw. Eng.*, IOS Press, 2006: pp. 223–232.
- [135] L.S. Sterling, K. Taveter, *The art of agent-oriented modeling*, MIT Press, 2009.
- [136] A. Sturm, O. Shehory, Agent-oriented software engineering: revisiting the state of the art, in: *Agent-Oriented Softw. Eng.*, Springer, 2014: pp. 13–26.
- [137] K. Taveter, M. Meriste, How Can Agents Help in Designing Complex Systems?, *J. Telecommun. Electron. Comput. Eng. JTEC.* 9 (2017) 1–8.
- [138] K. Taveter, M. Meriste, K.R. TTU, P. Dihé, M.S. CIS, S. Guarino, Validated simulation tool for crisis management strategies and planned actions, (2015).
- [139] K. Taveter, G. Wagner, Towards radical agent-oriented software engineering processes based on AOR modelling, *Agent-Oriented Methodol.* 10 (2005) 277–316.
- [140] Q.-N.N. Tran, G. Low, MOBMAS: A methodology for ontology-based multi-agent systems development, *Inf. Softw. Technol.* 50 (2008) 697–722.
- [141] A.J. Trappey, T.-H. Lu, L.-D. Fu, Development of an intelligent agent system for collaborative mold production with RFID technology, *Robot. Comput.-Integr. Manuf.* 25 (2009) 42–56.
- [142] M. Ughetti, T. Trucco, D. Gotta, Development of agent-based, peer-to-peer mobile applications on ANDROID with JADE, in: *Mob. Ubiquitous Comput. Syst. Serv. Technol. 2008 UBICOMM08 Second Int. Conf. On, IEEE*, 2008: pp. 287–294.
- [143] W.M. Van der Aalst, Pi calculus versus Petri nets: Let us eat “humble pie” rather than further inflate the “Pi hype,” *BPTrends.* 3 (2005) 1–11.
- [144] S. Van Mierlo, A multi-paradigm modelling approach for engineering model debugging environments, PhD Thesis, University of Antwerp, 2018.
- [145] Y. Van Tendeloo, A foundation for multi-paradigm modelling, PhD Thesis, University of Antwerp, 2018.
- [146] J. Venable, J. Pries-Heje, R. Baskerville, FEDS: a framework for evaluation in design science research, *Eur. J. Inf. Syst.* 25 (2016) 77–89.
- [147] M. Vrbaski, D. Petriu, D. Amyot, Tool support for combined rule-based and goal-based reasoning in Context-Aware systems, in: *Requir. Eng. Conf. RE 2012 20th IEEE Int.*, IEEE, 2012: pp. 335–336.
- [148] C. WaiShiang, S. YeeWai, S. Nizam, L. CheeWyai, Agent oriented requirement engineering for lake mathematical modelling: Preliminary study, *J. Telecommun. Electron. Comput. Eng. JTEC.* 8 (2016) 5–10.
- [149] P. Wavish, Situated action approach to implementing characters in computer games, *Appl. Artif. Intell.* 10 (1996) 53–74.

- [150] J. Whittle, J. Hutchinson, M. Rouncefield, The state of practice in model-driven engineering, *IEEE Softw.* 31 (2014) 79–85.
- [151] J. Whittle, J. Hutchinson, M. Rouncefield, H. akan Burden, R. Heldal, A taxonomy of tool-related issues affecting the adoption of model-driven engineering, *Softw. Syst. Model.* 16 (2017) 313–331.
- [152] B. Whitworth, A. Ahmad, Socio-technical system design, *Encycl. Hum.-Comput. Interact.* 2nd Ed. (2013).
- [153] D. Wilmann, L. Sterling, Guiding agent-oriented requirements elicitation: HOMER, in: *Qual. Softw. 2005QSIC 2005 Fifth Int. Conf. On, IEEE, 2005*: pp. 419–424.
- [154] C. Wohlin, P. Runeson, M. Höst, M.C. Ohlsson, B. Regnell, A. Wesslén, *Experimentation in software engineering*, Springer Science & Business Media, 2012.
- [155] M. Wooldridge, N.R. Jennings, Agent theories, architectures, and languages: a survey, in: *Intell. Agents*, Springer, 1995: pp. 1–39.
- [156] E.S. Yu, Towards modelling and reasoning support for early-phase requirements engineering, in: *Requir. Eng. 1997 Proc. Third IEEE Int. Symp. On, IEEE, 1997*: pp. 226–235.
- [157] H. Yu, Z. Shen, C. Miao, Intelligent software agent design tool using goal net methodology, in: *Proc. 2007 IEEEWICACM Int. Conf. Intell. Agent Technol., IEEE Computer Society, 2007*: pp. 43–46.
- [158] F. Zambonelli, N.R. Jennings, M. Wooldridge, Organisational rules as an abstraction for the analysis and design of multi-agent systems, *Int. J. Softw. Eng. Knowl. Eng.* 11 (2001) 303–328.
- [159] F. Zambonelli, N.R. Jennings, M. Wooldridge, Developing multiagent systems: The Gaia methodology, *ACM Trans. Softw. Eng. Methodol. TOSEM.* 12 (2003) 317–370.
- [160] H. Zhou, F. Chen, H. Yang, Developing application specific ontology for program comprehension by combining domain ontology with code ontology, in: *Qual. Softw. 2008 QSIC08 Eighth Int. Conf. On, IEEE, 2008*: pp. 225–234.
- [161] E. Zupancic, D. Trcek, K. Taveter, Agent-oriented Engineering of Trust Management Systems., in: *AT, 2012*: pp. 349–350.

Acknowledgements

Foremost, I bless God for his mercy and great love that I experienced during the whole period of my doctoral studies. This period was a great journey in search of societal excellence and new knowledge through exploration and exploitation of scientific knowledge.

I would like to thank Professor Kuldar Taveter for supervising and assisting in research from the beginning of my doctoral studies to this moment of submitting my doctoral thesis for defence. His endorsements, support, and encouragement to write this thesis had immense impact on how the thesis was finalised.

I consider myself very fortunate for having Associate Professor Alexander Norta as a second supervisor, mentor and officemate during the second half of this research journey. His support had a vital impact on how the thesis finally came together. I would like to deeply thank Dr. Cheah Wai Shiang from University of Malaysia at Sarawak for his fruitful comments on the key contributions of this thesis. Special thanks go to Aleksandr Sapožnikov, the former master's student of Tallinn University of Technology (TUT) for his assistance during the development of proof-of-concept prototype for this research.

I am grateful to different institutions that have financially supported this research work, including European Social Fund through Activity 4 of the DoRa programme, the government of Tanzania through the Institute of Finance Management, Tallinn University of Technology, Estonian IT Academy, and European Cooperation in Science and Technology through COST Action IC0801 – Agreement Technologies.

The final version of this doctoral thesis was scientifically shaped by many professors from TUT and the PhD committee. I am highly thankful to all committee members and all the professors for invaluable support and comments.

Beyond everything, special thanks to my beautiful, loving and wonderful wife, Jackline, who supported me all along the way during my PhD studies, and to our loving children Nathan, Elvis, Andres, Teresa and Mikaela, whose presence gives meaning to my life. Last but not the least, I thank my parents, brothers and sisters who helped and cared for me and my family during the whole period of our stay in Estonia. I would like to express my deepest gratitude to them for their prayers, love and support.

Abstract

Simulation and Prototyping of Sociotechnical Systems Using Agent-Oriented Modelling

Sociotechnical systems are complex collaborative systems consisting of machines, software, humans, and their environments. Many of the involved collaborating parties are autonomous, social, reactive and proactive and can therefore be termed as active entities or agents. This poses a challenge for requirements analysis, design and implementation of sociotechnical systems. The results of adapting agent-oriented software engineering (AOSE) methodologies to the engineering of sociotechnical systems are promising. However, such efforts have reported to be inadequate for effective support of a modelling process, quality assurance of modelling artefacts and prototyping. This, in turn, hinders conducting efficient development processes and achieving high-quality artefacts of sociotechnical systems.

This thesis proposes novel guidelines for representing agent-oriented design models of sociotechnical systems in Coloured Petri Nets (CPN) for simulation of the design models by CPN Tools. The thesis provides the support for system visualisation and for validation and verification of sociotechnical systems through simulations by CPN Tools. Further, this thesis proposes novel guidelines that support the prototyping of sociotechnical systems on the JADE framework based on agent-oriented design models. This prototyping possibility enhances the understanding of the user requirements at an early stage of the development process of a sociotechnical system. Lastly, this thesis describes a novel software tool that aims to reduce the modelling effort and improve the effectiveness of requirements analysis and design of sociotechnical systems by employing visual features and providing the support for consistency checking of models and information propagation between the models.

The validation results of the CPN modelling guidelines ascertain that they are more effective for simulating agent knowledge and behaviour design models and less effective for simulating interaction design models of sociotechnical systems by CPN Tools. Moreover, the validation results of the CPN modelling guidelines suggest that CPN Tools effectively supports system visualisation through message-sequence charts (MSC), validation of design properties through scenario-based analysis, and identification of unwanted design properties through state space verification. Furthermore, the validation results of the JADE prototyping guidelines ascertain that the usage of the guidelines is more effective for the development of shared and private knowledge by agents of sociotechnical systems compared to the current practice of using for prototyping only JADE website resources instead of applying the guidelines. Moreover, the validation results of the JADE prototyping guidelines demonstrate that conceptual objects of sociotechnical systems are necessary building blocks in developing JADE ontologies. Lastly, the validation results of the novel software tool demonstrate that the effectiveness of requirements modelling with the software tool is higher than the effectiveness of the practice of requirements modelling on paper except for goal decomposition which is slightly more effective when modelled on paper compared to modelling with the tool.

Kokkuvõte

Sotsiotehniliste süsteemide simulatsioon ja prototüüpimine kasutades agentorienteeritud modelleerimist

Sotsiotehnilised süsteemid on keerulised koostööl põhinevad süsteemid, mis koosnevad masinatest, tarkvarast, inimestest ja keskkondadest. Paljud koostöös osalevatest pooltest on autonoomsed, sotsiaalsed, reaktiivsed ja proaktiivsed, mistõttu neid võib nimetada aktiivseteks olemiteks ehk agentideks. See loob väljakutse sotsiotehniliste süsteemide nõuete analüüsile, kavandamisele ja teostamisele. Agentorienteeritud tarkvaratehnika (AOSE) metoodikate kohandamise tulemused sotsiotehniliste süsteemide loomiseks on lootustandvad. Aga samas on niisuguste jõupingutuste kohta teada, et need ei ole piisavad modelleerimisprotsessi efektiivseks toetamiseks, modelleerimise tehiste kvaliteedi kindlustamiseks ja prototüüpimiseks. See omakorda takistab efektiivsete arendusprotsesside läbiviimist ning sotsiotehniliste süsteemide kõrgekvaliteediliste tehiste saavutamist.

Käesolev väitekiri pakub välja uudsed juhtnöörid sotsiotehniliste süsteemide disainimudelite esitamiseks värvitud Petri võrkudena (CPN) disainimudelite simuleerimiseks tööriista CPN Tools abil. Väitekiri pakub tuge süsteemi käitumise visualiseerimiseks ning sotsiotehniliste süsteemide valideerimiseks ja verifitseerimiseks CPN Tools abil teostatud simulatsioonide kaudu. Edasi pakub väitekiri välja uudsed juhtnöörid, mis toetavad sotsiotehniliste süsteemide prototüüpimist raamistiku Java Agent Development Environment (JADE) abil nende süsteemide agentorienteeritud disainimudelite põhjal. See prototüüpimise võimalus suurendab arusaamist kasutajanõuetest sotsiotehnilise süsteemi arendusprotsessi varajases staadiumis. Lõpuks pakub väitekiri välja uudse tarkvaralise tööriista, mille eesmärgiks on vähendada modelleerimise jõupingutusi ning parandada sotsiotehnilise süsteemi nõuete analüüsi ja disaini tõhusust, kasutades visuaalseid võimalusi ning pakkudes tuge mudelite kooskõllalisuse kontrollimisele ja informatsiooni edasikandmisele mudelite vahel.

Värvitud Petri võrkude (CPN) abil modelleerimise juhtnööride valideerimise tulemused kinnitavad, et juhtnöörid on rohkem tõhusad agentide teadmiste ja käitumise disainimudelite simuleerimiseks ja vähem tõhusad agentide suhtlemise disainimudelite simuleerimiseks. Lisaks sellele näitavad CPN abil modelleerimise juhtnööride valideerimise tulemused, et CPN Tools toetab tõhusalt süsteemi käitumise visualiseerimist teadete järgnevuse skeemide (message-sequence charts, MSC) abil, disaini omaduste valideerimist stsenaariumipõhise analüüsi abil ning mittesoovitud disaini omaduste identifitseerimist olekuruumi verifitseerimise abil. Peale selle, JADE abil prototüüpimise juhtnööride valideerimise tulemused kinnitavad, et juhtnööride kasutamine on sotsiotehnilise süsteemi agentide jagatud ja privaatsete teadmiste realiseerimiseks tõhusam võrreldes prototüüpimisega ainult JADE veebiressursside kasutamise põhjal. Sellele lisaks demonstreerivad JADE prototüüpimise juhtnööride valideerimise tulemused, et sotsiotehniliste süsteemide kontseptuaalsed objektid on hädavajalikeks ehituskomponentideks JADE ontoloogiatega arendamisel. Lõpuks demonstreerivad uudse tarkvaralise tööriista valideerimise tulemused, et nõuete modelleerimise tõhusus tööriistaga on kõrgem kui nõuete paberil modelleerimise praktika välja arvatud eesmärkide liigendamise puhul, mis on paberil modelleerituna mõnevõrra tõhusam kui tööriista abil modelleerituna.

Appendix A

Msury Mahunnah, Alex Norta, Lixin Ma, and Kuldar Taveter. "Heuristics for Designing and Evaluating Sociotechnical Agent-Oriented Behaviour Models with Coloured Petri Nets." In Computer Software and Applications Conference Workshops (COMPSACW), 2014 IEEE 38th International, pp. 438–443. IEEE, 2014.

Heuristics for Designing and Evaluating Socio-Technical Agent-Oriented Behaviour Models with Coloured Petri Nets

Msury Mahunnah*, Alex Norta*, Lixin Ma*[†], Kuldar Taveter*

*Department of Informatics, Tallinn University of Technology, Tallinn, Estonia,

[†]Department of Computer Science, University of Shanghai for Science and Technology, Shanghai, China

Email: msury@gmail.com; alex.norta@gmail.com; malixin.usst@gmail.com; kuldar.taveter@ttu.ee

Abstract—Software agents are a means to support socio-technical decentralised systems that increase the complexity of daily life. Designing multi-agent systems involves modelling methods for which it is currently not possible to check for soundness before a technical implementation. To improve the design process, the agent models require a mapping to a formalisation that is sufficiently expressive to represent equivalent model properties. The formalized presentation must cater for evaluating the model soundness, simulation and performance experimentations with different test data. The paper gives a set of mapping heuristics from agent models to a sufficiently expressive formalisation representation that follows a real life running case stemming from the healthcare domain.

Keywords—Socio-Technical, Agents, Coloured Petri Nets, Heuristics, Behaviour, Design, Evaluation.

I. INTRODUCTION

Our society is becoming increasingly dependant on complex information technology (IT) systems for carrying out daily activities. The complexity of IT systems, mainly, stems from the integration and orchestration of independently managed software systems that are distributed in dynamic environments [1], such as healthcare, aviation, air traffic control, telecommunications, and so on. In addition, the behaviour of people who work across organizational, geographical, cultural and temporal boundaries [2] influences the complexity of such socio-technical IT systems [3] and thus, poses a great engineering challenge. We define a socio-technical system as an approach to complex organizational work design that recognizes the interaction between people and technology in workplaces.

In recent years, researchers have undertaken various studies in modelling the behaviour and knowledge sharing, of socio-technical systems, among interacting technical, societal and organisational aspects. These studies have focus on domains such as healthcare [4], military [5] and sociology [6], [7] using an agent-oriented paradigm [8]. The latter is a top-down holistic approach for modelling socio-technical systems by engaging all stakeholders during the analysis and design phases of a system's development life cycle. However, a gap exists in formalising and evaluating agent-oriented behaviour, knowledge and interaction models before the actual implementation of these kinds of systems.

In this paper, we fill the identified gap by answering the research question, how to systematically formalise and evaluate agent-oriented behaviour models for socio-technical systems?

To establish complexity-reducing separation of concerns, we deduce the following sub-questions: What is a suitable way for conceptualizing the behaviour of socio-technical agent-oriented systems based on a set of heuristics? What formalization is suitable for the first syntactically correctly designed agent-oriented behaviour models? What means exist to evaluate the soundness of agent-oriented behaviour models? This set of sub-questions assumes that a syntactic designing of agent-oriented behaviour models precedes the mapping to formalizations that carry equivalent model properties.

The paper structure is as follows. Section II describes a running case from the healthcare domain that helps in clarifying what artefacts we consider for this paper. Section III presents an agent-oriented goal model and behaviour-interface model for capturing socio-technical system behaviour. Section IV gives mapping heuristics towards a formalization and evaluation of agent-oriented behaviour models. Section V shows and explains the resulting formalised model of the running case and evaluates simulation results. Section VI presents related work and finally, Section VII gives the conclusion and provides future work.

II. RUNNING CASE FROM THE HEALTHCARE DOMAIN

Healthcare organizations aim to provide better quality of care by improving the information logistics among caregivers and patients, who work in a distributed way. In this paper, we consider a case study [9] for reporting Critical Laboratory Results (CLRs) to an appropriate caregiver from the North Estonian Medical Centre (NEMC) laboratory.

The case study identifies two weaknesses in the procedure for reporting CLRs at the NEMC laboratory. First, the procedure for reporting CLRs involves many people. This leads to two major problems: (1) high risk of human errors, (2) delay in reporting CLRs. A second weakness arises from the handling of CLRs similar to Normal Laboratory Results (NRLs). When a doctor who orders the laboratory tests is unreachable by phone, the laboratory guidelines suggest that staff make a phone call instead to the departmental nurse who tries to find another appropriate doctor. If the departmental nurse is also unreachable, the CLRs are sent to the Hospital Information System (HIS) comparable to NRLs. The delay poses the risk that patients do not receive adequate treatment in a state of emergency.

Results of the case study in [9] propose to improve the current system for reporting CLRs as follows. NEMC must

consider mobile technologies, to accurately identify the location of caregivers. The new intelligent information system is a socio-technical system, i.e., a software intensive system that has defined operational processes followed by human operators and that operates within an organization [10]. The envisioned socio-technical system specifies roles for human agents in healthcare organizations, such as patients, nurses and doctors. The human agents receive support by software agents that we define [8] as an entity that performs a specific activity in an environment of which it is aware, and that can respond to changes.

The execution of rules determine the behaviour of a software agent [11]. The rules execute upon detection of changes in the environment, such as event occurrences. Events may also stem from other collaborating agents. The dynamic environment influences the possible reachable states of agents after executing activities. For instance, when the doctor for receiving CLR is unavailable, the socio-technical agent proactively identifies and suggests another appropriate caregiver according to availability, medical knowledge and speciality.

In our previous work [9], we evaluate the analysis- and design models for socio-technical systems by engaging domain experts, i.e., healthcare professionals. In this paper, we suggest heuristics for evaluating syntactical correctness and soundness of socio-technical agent-oriented behaviour models by mapping to a formalization that carries equivalent model properties.

III. BEHAVIOUR MODELLING

The analysis of socio-technical systems where humans receive support from intelligent software agents, must follow an appropriate methodology. Various Agent Oriented Software Engineering (AOSE) methodologies exist with a technical emphasis on designing systems consisting of software agents, e.g., Tropos [12], MaSE [13], or Prometheus [14]. In [8], the described Agent-Oriented Modelling (AOM) method is a socio-technical approach that includes features similar to those in mentioned AOSE methodologies while taking into account the combination of human- and man-made agents in the system design process. In this section, we present two AOM model types, i.e., the goal model and behaviour interface model, that capture important socio-technical behavioural features from the running case.

The goal model serves as a container for three main components: functional requirements commonly referred to as goals, roles, and non-functional requirements. The latter has two categories, quality goals for non-functions requirements related to software and emotional goals for those related to humans. Parallelograms represent goals, sticky men are roles, clouds are quality goals and hearts are emotional goals as depicted in Figure 1. Goal models serve as communication media between technical and non-technical stakeholders to establish a better understanding of the problem domain. The goal model starts with an overall objective of the socio-technical system that is known and clear to all stakeholders. Goals decompose into sub-goals where each sub-goal represents some aspect for achieving its parent goal [15]. Note that the lowest sub-goal must be atomic.

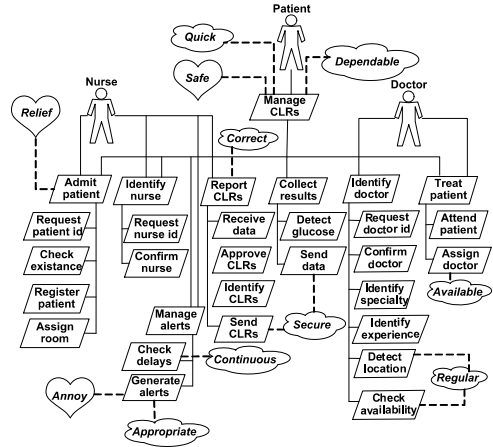


Fig. 1. Goal model of the socio-technical running healthcare case.

In the goal model of Figure 1, we first present the uppermost goal, viz., *Manage CLR's* with the attached role of *Patient* who is the focus of the analysis. The emotional goal *Safe*, and two quality goals *Quick* and *Dependable* are also attached to the main goal. The latter means to avoid service failures that are more frequent and more severe than acceptable [16]. The main goal *Manage CLR's* splits into seven sub-goals: *Admit patient*, *Identify nurse*, *Manage alerts*, *Report CLR's*, *Collect results*, *Identify doctor* and *Treat patient*. In addition to the role *Patient* who is the only role responsible for achieving the sub-goal *Collect results*, the role *Nurse* is responsible for the first four sub-goals and the role *Doctor* is responsible for the last two sub-goals. Each of the identified sub-goals has further refining third level sub-goals that are the lowest-level sub-goals for this running case. These sub-goals represent the activities of the socio-technical system. Parallel to the process of breaking down the sub-goals, we also identify suitable quality goals and emotional goals. For example, the sub-goal *Admit patient* in Figure 1 has the emotional goal *Relief*, meaning execution of the four activities of *Request patient ID*, *Check existence*, *Register patient* and *Assign room* targets at ensuring the patient feels a relief during the admission process.

Goal models focus on identifying functional and non-functional requirements of the whole socio-technical system rather than simple activities conducted by individual agents. During the design phase, behaviour models for individual agents facilitate the refinement of the goal model resulting from the analysis of the running case. A behaviour model in AOM has two parts: an agent behaviour model coupled with a behaviour interface model [8]. The former describes the rule-based behaviour of an agent, while the latter focuses on identifying the activities associated triggers, preconditions, and postconditions.

Table I presents the behavioural interfaces of four important activities in fulfilling the goal *Admit patient*. Each activity must have one trigger and at least one postcondition. Preconditions may either exist or not, depending on the nature of

TABLE I. BEHAVIOURAL INTERFACES OF ACTIVITIES FOR THE SUB-GOAL "ADMIT PATIENT"

| Activity | Trigger | Preconditions | Postconditions |
|--------------------|---------------------|-------------------|--|
| Request patient ID | New case detected | | Received Patient ID |
| Check existence | Patient ID received | ID found | Registered patient |
| Register patient | Patient ID received | ID not found | Updated patient DB Registered Patient |
| Assign room | Patient registered | Room is available | Assigned room |

the corresponding activity, e.g., the activity *Request patient ID* has only one trigger and one postcondition without any precondition. The execution of an activity is either triggered by the occurrence of an event, or by a pre-condition after the occurrence of the event. For example, the activity *Assign room* has three interfaces, *Patient registered* as a trigger, *Room is available* as precondition and *Assigned room* as postcondition. The given interface for the activity *Assign room* assumes room availability before patient registration. If the room is available after the registration of a patient then *Patient registered* becomes the precondition and *Room is available* as the trigger. In other words, the trigger and precondition may exchange their roles at runtime.

The behaviour interface models, designed by the domain experts, require a mapping to a formalization for an evaluation that ensures the models are sound before a technical implementation. The following section provides a step-by-step procedure for formalising agent-oriented behaviour models.

IV. MAPPING AGENT-ORIENTED BEHAVIOUR MODELS TO A FORMALIZATION

In order to formulate sound agent-oriented behaviour models for the socio-technical system, it is important to map AOM models to a formal and deterministic notation that allows for a strong evaluation. Consequently, we consider for Colored Petri Nets [17] (CPN) with mature tool support¹ as a mapping target. CPN is a graphical oriented language for design, specification, simulation and verification of systems. CPN has an intuitive, graphical representation that consists of a set of modules (pages), each containing a network of places, transitions and arcs. The modules interact with each other through a set of well-defined interfaces in a similar way as known from many modern programming languages. Places may hold multiple tokens that carry colour, i.e., attributes with values. Transitions fire when all input places hold the required sets of tokens and produce condition-adhering tokens into output places.

We next explain the mapping between AOM and CPN that Table II summarizes. Places and transitions are connected by directed arcs denoting the flow during the execution of activities and resources in AOM. Rectangles depict transitions that represent simple activities performed by agents. Ovals depict places that may either be attached with an outgoing arc to the transition or incoming arc from the transition. The former represents a trigger or precondition while the latter represents the postcondition of given activity in AOM. Double-boarded rectangles depict modules that represent goals in the socio-technical system that can further be broken-down into simpler sub-goals or activities. During the enactment of a CPN

model, flow of control passes to the sub-goals or activities (in the AOM equivalent) associated with a parent goal represented as module. This way, a CPN model represents a hierarchical structure of the goal model in AOM. When mapping agent-oriented behaviour models to CPN, behaviour interface models as shown in Table I represent each identified activity of the socio-technical system found in the goal model of Figure 1, i.e., identifying and deciding about the triggers, preconditions and postconditions of each activity.

TABLE II. NOTATIONS FOR MAPPING AOM TO CPN






| Notation | Name |
|---|-------------------------|
|  | Connecting Arc |
|  | Sub-Goal or Activity |
|  | Trigger or Precondition |
|  | Postcondition |
|  | Goal |

Table III describes a sample behaviour interface model for two consecutive activities. Activity 1 has Trigger 1, Precondition 1 and 2, and Postcondition 1. When mapping Activity 1 to CPN, it turns into a transition connected by arcs to four different places. Among them, three are incoming arcs from the three places representing Trigger 1, Precondition 1 and Precondition 2. The other connection to Activity 1 is an outgoing arc to a place representing Postcondition 1. Furthermore, Table III shows that Postcondition 1 triggers Activity 2 since its execution follows just after completion of Activity 1. Thus, making Postcondition 1 connected to Activity 2 by an outgoing arc and referred as a trigger named Trigger 2 by Activity 2. Following that, the outgoing arcs from Activity 2 connect to two places, namely, Postcondition 2 and Postcondition 3.

TABLE III. BEHAVIOUR INTERFACES FOR ACTIVITY 1 AND ACTIVITY 2

| Activity | Trigger | Preconditions | Postconditions |
|------------|-----------|----------------------------------|------------------------------------|
| Activity 1 | Trigger 1 | Precondition 1 Precondition 2 | Postcondition 1 |
| Activity 2 | Trigger 2 | | Postcondition 2 Postcondition 3 |

Figure 2 presents a CPN model of interconnected nodes representing triggers, preconditions and postconditions of Activity 1 and Activity 2 mapped from the behaviour interface model given in Table III. Data-flows are not captured in the sample CPN model of Figure 2. Heuristics for modelling data-flows are out of scope for this paper and left as future work.

The following section gives a full implementation of the running case's CPN model for studying the behaviour of socio-technical systems and evaluating soundness.

¹<http://cpntools.org/>

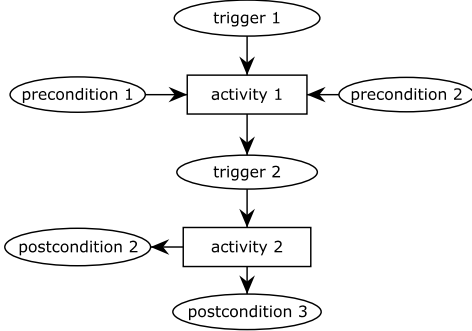


Fig. 2. A CPN model for sample behaviour interface model.

V. FORMALIZED CPN MODEL AND EVALUATION

Following the procedure for mapping agent-oriented behaviour models to CPN, Section V-A presents a formalised CPN model of the running case. In Section V-B, we simulate the CPN model for studying socio-technical behaviours by altering some factors such as availability of the doctors and the average time it takes for attending patients with CLRs.

A. CPN Model for the Running Case

A CPN model in Figure 4 is equivalent to the earlier presented agent-oriented goal model of the running case. The atomic activities from the goal model we map to the behaviour interface models. Due to page limitations, it is not possible to show all models in this paper. Instead, we refer the reader to the full version² in the footnote for the complete CPN model of the running case. Behaviour interface models consist of triggers, preconditions and postconditions for each activity depicted by the lowest level sub-goals in the goal model.

For representing the goal *Admit patient*, Figure 3 shows the equivalent refinement as a CPN module. The refinement consists of four transitions mapped from the activities in the behaviour interface model given in Table I. Each transition is connected to places by at least one incoming arc and one outgoing arc. The former describes a trigger, or precondition while the latter describes a postcondition identified by the help of AOM models in Section III. The CPN modules comply to the guidelines given in the previous section that each activity must have a trigger and at least one postcondition.

For the running case, the module simulation triggers when a new patient arrives at the hospital. The transition *request patient ID* fires and results in a new place *received ID*. This new place acts as a trigger to two possible transitions, namely, *check existence* and *register patient*. In addition, the execution of the former requires existence of a suitable token in *patient DB* place as a precondition. The place *patient DB* also serves as the postcondition together with the place *registered patient* for the mentioned two transitions.

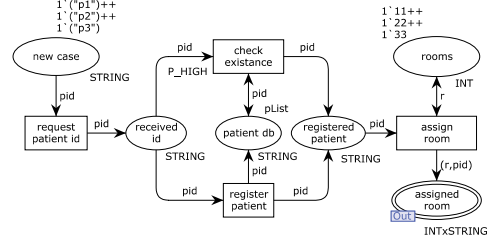


Fig. 3. A formalised CPN model for the "Admit patient" sub-goal.

We introduce the label *P_HIGH* to the transition *check existence* indicating firing priority. If *patient ID* is not found in the *patient DB*, the transition *check existence* never fires. Therefore, the alternative transition *register patient* fires by registering the patient into *patient DB* and having the output place *registered patient*. With the existence of available rooms, the *assign room* transition is triggered by the place *registered patient*. The execution of this module ends with the place *assigned room* that connects to remaining activities of the healthcare system represented in CPN. These activities are refinements of different modules corresponding to their parent goals such as *Collect Results*, *Identify Doctor* and so on. Figure 4 represents a higher-level CPN model of the running case with all the modules and relationships among them.

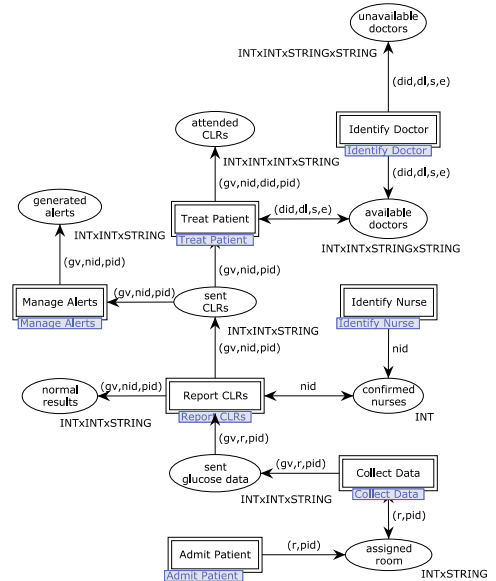


Fig. 4. A formalised CPN Model from AOM.

²https://www.dropbox.com/s/9efc9t9zn2oqtn/aom_cpn_model.cpn

B. Simulation and Results

The simulation of the complete CPN Model for the running case aims at identifying the optimal number of available doctors that attend CLRs. The simulation also assures a minimal number of generated alerts. According to the depiction of CPN sub-model Figure 5, the system generates an alert when there is no available doctor to attend CLRs. A firing of transition *assign doctor* requires a fulfilled precondition *available doctor*. Otherwise, the transition *check delays* fires, followed by transition *generate alert* that Figure 5 does not capture.

For one CPN simulation, three different patients generate a total of 100 CLRs in an interval of 10 time units. We record the number of available doctors, generated alerts, attended CLRs, and average time taken by doctors when attending CLRs. The amount of activities carried out determines the availability of doctors, e.g., for attending patients. Table IV summarises the results of the CPN simulation and assumes attending CLRs consumes between 0 and 10 time units. The results in Table V assume the time taken for attending CLRs consumes between 10 and 20 time units.

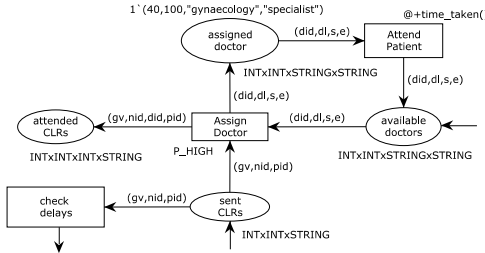


Fig. 5. A formalised CPN sub-model for generating alerts.

TABLE IV. RESULTS TABLE 1

| Available Doctors | Attended CLRs | Generated Alerts |
|-------------------|---------------|------------------|
| 1 | 32 | 64 |
| 2 | 72 | 28 |
| 3 | 100 | 0 |

Table IV and Table V deduce approximate numbers for respective doctors who attend CLRs and the number of generated alerts. In both tables, the trends of the results comply with real-life observations [9] where the number of generated alerts decreases with an increase in the number available doctors. With a fixed time interval for generating CLRs, the results in Table IV suggest a need for 3 available doctors to minimize the number of generated alerts, while the results in Table V show 6 available doctors to achieve the same results. The doubled number of available doctors in Table V complies with the doubled range in the time taken for attending CLRs. The summarised results in these two tables not only show the real-life coherent behaviour of the designed socio-technical system but also the correctness and soundness of the designed CPN model.

In the next section, we discuss related work for formalising and evaluating socio-technical agent-oriented behaviour

TABLE V. RESULTS TABLE 2

| Available Doctors | Attended CLRs | Generated Alerts |
|-------------------|---------------|------------------|
| 1 | 16 | 84 |
| 2 | 34 | 66 |
| 3 | 53 | 47 |
| 4 | 70 | 30 |
| 5 | 85 | 15 |
| 6 | 100 | 0 |

models.

VI. RELATED WORK

The trend towards using Petri Nets for modelling and analysing is gaining prominence. For example, the conceptual framework AgOS [18] allows for a high level representation of a multi-agent system environment using classical Petri Nets. The disadvantage of using classical Petri nets in AgOS is a decrease of expressiveness for large systems. As CPN allow for modelling hierarchies, the approach in our paper is more scalable. In [19], agents for the management of computing resources in clouds the authors formalise using Petri Nets. These examples have a technical focus for using Petri Nets in the design of multi-agent systems. Instead, the relatively new AOM focus is socio-technical in nature and thus, recognizes the interaction between people and technology in workplaces that other research work does not consider.

Automating a technical realisation of multi-agent system research in [20] presents. The authors show a domain engineering process for developing multi-agent system product lines including supporting agent variability and providing agent feature traceability resulting in reduced time-to-market and lower development costs. CPN Tools also offers an automatic translation to Java code that a programmer can implement to full completion.

For formalising agent models, other options exist too. A tool called Rodin [21] supports system formalization with Event-B that uses set theory and refinement through theorem proving to represent systems at different abstraction levels. The generated mathematical proof verifies the refinements. The Rodin tool integrates system modelling and proving of formalised systems.

The so-called PiVizTool [22] supports system design with π -calculus. The original purpose is to model and analyse Web-service choreographies and it is also a candidate for formalising aspects of AOM. However, as [23] discusses, π -calculus is differently to Petri Nets not a graphical notation that system modelling and analysis more challenging for laymen. Using mature CPN Tools is easier to accomplish and it suffices to understand the use of the integrated tools for simulation, performance testing and verification to generate quickly soundness checks for AOM.

VII. CONCLUSION

In this paper, we define heuristics for formalising and evaluating agent-oriented behaviour models of socio-technical systems. The aim is to ensure before an actual implementation that the models are sound and coherent. A running case from

the healthcare domain demonstrates the mapping from AOM to CPN. The running case focuses on the management of Critical Laboratory Results by utilising a minimum number of resources, including humans such as doctors and nurses.

For capturing socio-technical requirements of the running case, the AOM approach is suitable for engaging technical and non-technical stakeholders from the healthcare domain. A goal model and behaviour interface model summarise the results of the AOM-based design. The former specifies in a hierarchically refining way, the objectives of a socio-technical system while the latter defines the triggers, preconditions and postconditions for each identified activity.

We give a set of mapping heuristics from AOM to CPN with the latter having the advantage of providing visual elements of places, transitions, modules, arcs that may carry condition statements with the required expressiveness to capture the properties of the equivalent AOM model. The advantage of this mapping is that the CPN model allows for tool supported simulation, performance testing and model-checking based verification that is currently not possible in AOM. Consequently, such a CPN-based evaluation gives indications about the soundness of the AOM models and coherent behaviour of the designed socio-technical system. The running case of the paper shows that the results of the CPN-model simulations yields results corresponding to empirical data collected from the healthcare domain.

As future work, we plan to develop tool-support for mapping from AOM to CPN that requires a detailed definition of the mapping rules beyond the heuristics given in this paper. Furthermore, this tool must also comprise mechanisms for a rapid system implementation, for example, by mapping automatically to programming code that reduces full development time to a minimum.

ACKNOWLEDGEMENT

This work is partially supported by the project SF0140013s10 "Model-based Creation and Management of Evolutionary Information Systems" (2010-2014) by the Estonian Ministry of Education and Research; The IT Academy Programme for Information and Communication Technology Research of Tallinn University of Technology (13-09-00-1); The Dawn Program of Shanghai Education Commission (11SG44) and The Research Fund for the Doctoral Program of Higher Education of China 20123120130001).

REFERENCES

- [1] I. Sommerville, D. Cliff, R. Calinescu, J. Keen, T. Kelly, M. Kwiatkowska, J. Mcdermid, and R. Paige, "Large-scale complex it systems," *Communications of the ACM*, vol. 55, no. 7, pp. 71–77, 2012.
- [2] P. Carayon, "Human factors of complex sociotechnical systems," *Applied ergonomics*, vol. 37, no. 4, pp. 525–535, 2006.
- [3] E. Trist, "Some social and psycho," *Human relations*, vol. 4, p. 3, 1951.
- [4] M. Mahunnah and K. Taveter, "A scalable multi-agent architecture in environments with limited connectivity: Case study on individualised care for healthy pregnancy," in *7th IEEE International Conference on Digital Ecosystems and Technologies (DEST)*. IEEE, 2013, pp. 84–89.
- [5] I. Shvartsman and K. Taveter, "From agent-oriented models to profile driven military training scenarios," in *Intelligent Distributed Computing VII*. Springer, 2014, pp. 317–322.
- [6] S. Pedell and L. Sterling, "Agent-based modelling for understanding sustainability," in *Agents in Principle, Agents in Practice*. Springer, 2011, pp. 398–409.
- [7] S. Pedell, T. Miller, L. Sterling, F. Vetere, and S. Howard, "Substantiating agent-based quality goals for understanding socio-technical systems," in *Advanced Agent Technology*. Springer, 2012, pp. 80–95.
- [8] L. Sterling and K. Taveter, *The art of agent-oriented modeling*. MIT Press, 2009.
- [9] M. Mahunnah, A. Koorts, and K. Taveter, "Towards distributed sociotechnical system for reporting critical laboratory results," in *The 6th International Conference on Health Informatics (HEALTHINF), Barcelona, Spain*. SciTePress-Science and Technology Publications, 2013.
- [10] E. Trist, "The evolution of socio-technical systems," *Occasional paper*, vol. 2, p. 1981, 1981.
- [11] S. A. DeLoach, "Modeling organizational rules in the multi-agent systems engineering methodology," in *Advances in Artificial Intelligence*. Springer, 2002, pp. 1–15.
- [12] P. Giorgini, J. Mylopoulos, and R. Sebastiani, "Goal-oriented requirements analysis and reasoning in the tropos methodology," *Engineering Applications of Artificial Intelligence*, vol. 18, no. 2, pp. 159–171, 2005.
- [13] S. A. DeLoach, "Analysis and design using mase and agentool," DTIC Document, Tech. Rep., 2001.
- [14] L. Padgham and M. Winikoff, "Prometheus: A methodology for developing intelligent agents," in *Agent-oriented software engineering III*. Springer, 2003, pp. 174–185.
- [15] J. Marshall, "Agent-based modelling of emotional goals in digital media design projects," *International Journal of People-Oriented Programming*, 2014.
- [16] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing," *Dependable and Secure Computing, IEEE Transactions on*, vol. 1, no. 1, pp. 11–33, 2004.
- [17] K. Jensen, L. Michael, K. L. Wells, K. Jensen, and L. M. Kristensen, "Coloured petri nets and cpn tools for modelling and validation of concurrent systems," in *International Journal on Software Tools for Technology Transfer*, 2007, p. 2007.
- [18] R. K. Chatterjee, A. Sarkar, and S. Bhattacharya, "Modeling and analysis of agent oriented system: Petri net based approach," in *11th International Conference on Software Engineering Research and Practice (SERP 11, WORLDCOMP 2011)*, vol. 1, 2011, pp. 17–23.
- [19] W. Iqbal and S. Yousaf, "Formal modeling of agent based cloud computing services using petri nets," *VFAST Transactions on Software Engineering*, vol. 1, no. 2, pp. 1–6, 2013.
- [20] I. Nunes, C. J. D. Lucena, D. Cowan, U. Kulesza, P. Alencar, and C. Nunes, "Developing multi-agent system product lines: from requirements to code," *International Journal of Agent-Oriented Software Engineering*, vol. 4, no. 4, pp. 353–389, 2011.
- [21] J.-R. Abrial, M. Butler, S. Hallerstede, T. S. Hoang, F. Mehta, and L. Voisin, "Rodin: an open toolset for modelling and reasoning in event-b," *International journal on software tools for technology transfer*, vol. 12, no. 6, pp. 447–466, 2010.
- [22] P. Papapanagiotou and J. D. Fleuriot, "A theorem proving framework for the formal verification of web services composition," *arXiv preprint arXiv:1108.2348*, 2011.
- [23] W. van der Aalst, "Pi calculus versus petri nets: Let us eat 'humble pie' rather than further inflate the 'pi hype'," 2003.

Appendix B

Cheah Wai Shiang, Bong Tien Onn, Fu Swee Tee, Muhammad Asyraf bin Khairuddin, and **Msury Mahunnah**. "Developing Agent-Oriented Video Surveillance System through Agent-Oriented Methodology (AOM)." CIT. Journal of Computing and Information Technology 24, no. 4 (2016): 349–368.

Developing Agent-Oriented Video Surveillance System through Agent-Oriented Methodology (AOM)

Cheah Wai Shiang¹, Bong Tien Onn¹, Fu Swee Tee², Muhammad Asyraf bin Khairuddin¹ and Msury Mahunnah³

¹Faculty of Computer Science & IT, Universiti Malaysia Sarawak, Sarawak, Malaysia

²Faculty of Engineering, Computing and Science, Swinburne University Sarawak, Sarawak, Malaysia

³Faculty of Information Technology, Tallinn Technology University, Tallinn, Estonia

Agent-oriented methodology (AOM) is a comprehensive and unified agent methodology for agent-oriented software development. Although AOM is claimed to be able to cope with a complex system development, it is still not yet determined up to what extent this may be true. Therefore, it is vital to conduct an investigation to validate this methodology. This paper presents the adoption of AOM in developing an agent-oriented video surveillance system (VSS). An intruder handling scenario is designed and implemented through AOM. AOM provides an alternative method to engineer a distributed security system in a systematic manner. It presents the security system at a holistic view; provides a better conceptualization of agent-oriented security system and supports rapid prototyping as well as simulation of video surveillance system.

ACM CCS (2012) Classification: Computing methodologies → Artificial intelligence → Distributed artificial intelligence → Intelligent agents

Software and its engineering → Software notations and tools → System description languages → System modeling languages

Keywords: agent-oriented software engineering, video surveillance

1. Introduction

Agent technology has been used in building various domain specific applications. The agent paradigm introduces a software entity (e.g. agent) that is autonomous, proactive and able to interact with other agents for task accomplishment [1]. This kind of software supports complex applications like ambient intelligence, e-business, peer-to-peer, bio-informatics, negotiation [2] which demand the software to be robust, effective [3], co-operative to wide envi-

ronments, customizable to support user needs, secure, and evolve over time to cope with changing requirements.

Until recently, the agent technology has been adopted in a range of areas including collaborative learning games [4], rural ICT [5], ubiquitous computing, e-commerce (business to business-B2B and business to client-B2C) [6], robotic [7], library [8], e-learning, manufacturing, logistic [9], environment, banking [10], construction [11], bioinformatics, accident management [12], power management [3], crisis management [13], sustainable software [14], mathematical model [15] and grid computing. For example, the agent technology is used to support the collaborative design environment among the project's participants in the construction application. It is used to facilitate decision support at various stages of construction project like engineering design, negotiation and so on. However, agent technology has not been widely adopted by the software community [16]. The reasons for the setbacks are the diversity of agent-oriented software engineering methodologies and the lack of maturity in some of the methodologies [17]. The agent methodologies are proposed to aid the agent developer with the introduction of technique, terminology, notation and guideline during the development of the agent system [18].

To date, about 30 agent-oriented methodologies have been introduced [19]. It is reported that some of the agent methodologies lack generality. They are focused on specific systems and agent architectures [20]. In addition, some

of the methodologies do not consist of sufficient detail to be of real use [21]. The variety of agent methodologies that have direct or indirect influences on the object-oriented methodologies can cause difficulty for the industrial developers in selecting the methodology [17]. This paper presents a detailed case study to validate agent-oriented methodology (AOM). Agent-oriented methodology (AOM) [22] is a comprehensive and unified agent methodology for agent-oriented software development. Although AOM is claimed to be able to cope with a complex system development, it is still not yet determined up to what extent this may be true. Therefore, it is vital to conduct an investigation to validate this methodology in order to promote the agent technology to a wider community.

Video surveillance is a complex system. The adoption of agent technology in video surveillance leads to several benefits. The agent paradigm supports decentralization, autonomy, fault tolerance and flexible [1], robustness, low coupling [23]. Hence, it is worth to research into the adoption of AOM in this domain.

The paper first covers the requirements of the video surveillance system (VSS) through HOMER, an elicitation method for AOM. Then, the conceptualized domain modelling of VSS is presented. This is followed by the elaboration of VSS design through platform independent design modelling. Finally, it demonstrates the transformation of the design models into JADE implementation. The main contribution of this paper is to validate the AOM through a case study of VSS development. This paper shows the feasibility and applicability of AOM to model a complex distributed system. In addition, the detailed modelling can be served as a guideline for developers in engineering a distributed security system, VSS.

Section 2 presents the survey on agent-based video surveillance systems. It covers the current practice in designing and developing agent-based surveillance systems and the background of agent-oriented methodology. An elaborated case study is presented in Section 3. An intruder handling scenario is highlighted and used for the rest of the discussion in this paper. Section 4 and Section 5 present the modelling process of intruder handling through AOM. Section 4 covers the requirement elicitation step in AOM. Meanwhile, Section 5 presents the details modelling of intruder han-

dling through conceptual domain modelling, platform independent design and modelling and platform specific design and modelling. Section 6 presents the implementation of intruder handling in JADE, a multi-agent platform for developing agents in Java. A runtime aspect of agent based intruder handling is presented in this section. The paper is concluded in Section 7.

2. Related Works

Agent technology is adopted in video surveillance [1], [24], [25]. A multi-agent framework is introduced for the detection of suspicious activities in crowded scenes in a distributed multi-camera closed circuit TV (CCTV) network environment [1]. In [24], [26], agent coordination protocol and scheme are introduced to support agents-controlled camera. In this case, the agent behaves like human camera operator to reason and communicate on surveillance tasks. A multi-agent architecture is introduced to overcome the limitations of the current surveillance systems [25]. It has been reported that the current surveillance solutions suffer from the lack of flexibility and scalability. An empirical study is conducted and it validates the flexibility and scalability of the agent technology in distributed video surveillance [25]. From the survey, a software engineering approach (e.g. agent architecture, requirement engineering study etc) is adopted when designing and developing multi-agent video surveillance system. Although agent methodologies are introduced, there is not much addressing in designing and developing an agent-based video surveillance system. In line with the work to adopt agent methodology in agent-based video surveillance [24], [27], [28], [29], we adopt AOM in designing and developing a video surveillance system. The AOM is able to provide an alternative technique in modelling an agent-based surveillance system.

[27] presents the modelling of a surveillance system using the agent methodology named Ignenias. [23] provides a detailed description of VigilAgent methodology, which has been applied to modelling and implements multisensory surveillance systems. Finally, [28] presents the modelling and development of a person-following mobile robot application using Prometheus.

AOM is a methodology to model a complex socio-technical system. Sterling and Taveter combined the ROADMAP and AOR methodology [22] to produce a set of systematic methods, vocabularies and notations for conceptualizing socio-technical systems. It is compliant to model-driven development approach. Generally, the modelling process consists of conceptual independent modelling (CIM), platform independent design (PIM), and platform specific design (PSM). Specifically, the modelling process involves the design of goals, roles, interactions and domain knowledge models. This is followed by deciding the agent types, identifying the agents' knowledge, formulating interactions between agents and determining agents' behaviour. In brief, each agent models are described as follows.

Goal model. Goal model describes the purpose of the system (e.g. system functionality) at a higher level of abstraction. The notion of goal provides an overview of the functionalities that should be achieved by an agent system. Goals can be divided into sub-goals. Achieving a goal consumes resources and a goal is related to a particular role which indicates the actor or agent that is involved in achieving the goal [30].

Role model. A role model describes the roles involved within an organization. A role model is represented as a role schema, which consists of the following elements: role name, role description, responsibilities, and constraints. In an organization, people are assigned roles and positions to perform specific tasks. A position is required to subsume common tasks and sub-tasks under it.

Domain Model. The domain model represents the information that is handled by the system as a set of domain entities and the relationships between them.

Behaviour model. A behaviour model outlines the actions, reactions and responses of different types of agents [22]. It enables both proactive and reactive behaviour to be modelled. An agent achieves a goal by performing activities. The sequence of activities is modelled by means of control flows. A rule is the basic behaviour modelling construct. These rules are triggered either at the start of activities or by conditions that have been fulfilled, or an action event caused by external agents. The execution of a particular activity is modelled by triggering a rule to update the agent's mental state and/or

to send messages or to perform an action. An agent is proactive if its mental state is capable of triggering an activity. An agent reacts due to the perception received through a communication action or physical action by a human agent. A communication action involves exchanging messages between agents. A physical action involves a direct command by a human, which normally occurs through a graphical user interface.

Interaction Model. An interaction model models the social influence between agents. In this model, interactions between agents are represented through message passing. The interaction model models the content and the order of the messages to be exchanged.

Scenario model. A scenario model consists of activities to describe how agents carry out certain roles to achieve a particular goal.

Knowledge model. A knowledge model represents the details of the information types that are required by agents to solve a particular problem. A knowledge model specifies informational object types, predicate types, relationships between objects, and private and shared objects.

3. Elaborated Case Study – Distributed Video Surveillance System – Intruder Handling Scenario

Residential home burglary is a serious social problem in Malaysia. Thieves like to break into houses when the house is unattended, especially during daytime. Thieves are intruders who break into the house through climbing over the wall, breaking the lock or door etc. In order to prevent thieves and home break-ins, CCTV is installed to monitor any intruders who are entering the house. The CCTV detects any unfamiliar faces or intruders, then sends an intrusion alert to the house owner as well as to the police station. The police officer will wait for the confirmation of the detected intruder from the house owner and further notify the on duty police patrol officer if needed. In addition, the police officer will contact the house owner for further action. In the following section, we present modelling of the elaborated case study through AOM. This involves elicitation requirement process, and agent modelling (CIM, PIM and PSM).

4. Requirement Elicitation of Agent-Oriented VSS

A Human-Oriented Method for Eliciting Requirement, HOMER [31] is used to elicit requirements for an agent system. HOMER is based on the organization metaphor in "hiring a staff" to collect and identify the requirements of a given problem. The elicitation questions are shown in Table 1. In this case, software engineers elicit and reason on various considerations in recruiting staffs to solve a problem. From the answers gathered, the discovered requirements can be easily translated into the goal model, role model, organization model and domain model based on the guidelines proposed by [32]. In other words, the questions described in HOMER have a direct realization in the ROADMAP goal model and role schema.

Table 1 shows the elicitation answers for intruder handling scenario. Several stakeholders are involved in working on intruder handling scenario. They are the security manager, security personnel, family members (e.g. house owner and family members), visitors and neighbors. Each stakeholder has their own role and responsibilities. For example, the security manager has the responsibilities to observe the changes of environment, alert authorized personnel or house owner upon an intruder detected, by sending a message, as well as by continuous monitoring and tracking the intruder. The information tabulated in Table 1 is used to furnish the agent modelling process as elaborated in the following section.

5. Agent Modelling

Within the AOM, three phases are involved in modelling a multi-agent system. They are conceptual independent modelling (CIM), platform independent design and modelling (PIM) and platform specific design and modelling (PSM). Section 5.1 presents the conceptual domain modelling of intruder handling. Section 5.2 presents the platform independent design and modelling of intruder handling, and finally Section 5.3 presents the JADE-based design and modelling of intruder handling. All the models are presented in the appendix A.

5.1 Conceptual Independent Modelling

The CIM level reflects the early requirements and analysis for an agent-oriented system. Requirements analysis is a common stage among various agent-oriented methodologies, which is used to model agent system at a higher level of abstraction as well as to understand and analyse the requirements for developing an agent system. It is intended to present an overview of the system and determine its functionalities. Ignoring it can lead to misunderstanding the system in design [33]. Furthermore, the analysis normally involves activities that provide the context in which the system is to be designed [33]. The conceptual domain modelling of intruder handling is presented at Stage I: modelling goal and role, Stage II: modelling role, Stage III: modelling interaction frame and Stage IV: modelling domain knowledge.

Stage I: modelling goal and role. Figure 1 shows the goal model to handle an intruder. The notion of goal provides an overview of the functionalities that should be achieved by an agent system. Goals can be divided into sub-goals. In addition to functional goals, there are quality goals that represent non-functional requirements for the system. Achieving a goal consumes resources and a goal is related to a particular role which indicates the actor or agent that is involved in achieving the goal [22]. A role is the position played by an individual in an organization. The agent is a software entity that is situated in an environment.

In Figure 1, quality goals are incorporated in the main goal, indicating that the response needs to be appropriate and timely. Three roles are required to achieve the overall goal: Security Manager, Intruder, and Evaluator. The overall goal has been decomposed into five sub-goals: "Detection", "Identify", "Respond", "Scheduling" and "Evaluate". Two quality goals, "Timely detection" and "Accurate identification" are added to the "Detection" and "Identify" sub-goals respectively. The "Respond" sub-goal in turn has been expanded into two sub-goals: "Greeting" and "Communication". The "Communication" sub-goal is further divided into eight sub-goals. To accomplish these sub-goals, additional roles such as police, visitor, security guard, volunteer person (RELA), insurance agent, family members, immediate neighbour and owner are involved.

Table 1. Elicitation questions and answers for intruder handling.

| From HOMER's question | Answer(s) |
|---|---|
| 1. If you were to hire more staff to handle your current problem, which positions would you need to fill? | <ol style="list-style-type: none"> 1. Security manager 2. Security personnel 3. Family members 4. Visitors 5. Neighbours |
| 2. For each position, we need to collect a "job description": (a) What is the purpose of this position? What aspects of the problem will this position solve or partially solve? | <ol style="list-style-type: none"> 1. Security manager <ul style="list-style-type: none"> • Observe the environment change. • Alert authorized personnel/person registered in the system of environment change. • Security manager will detect unauthorized person and send an alert message to the house owner, and other registered personnel in the system. • Send location of the threat to the security personnel for further action. 2. Security personnel <ul style="list-style-type: none"> • Respond immediately to home security threat. • Go to the location immediately after receiving alert message. 3. Family members, visitors, neighbours <ul style="list-style-type: none"> • Take immediate precaution and stay away from security threat location. |
| 2. For each position, we need to collect a "job description": (b) What tasks will commonly be required? | <ol style="list-style-type: none"> 1. Security manager <ul style="list-style-type: none"> • Send an alert message and location to security personnel, home owner, family members, visitors and neighbors. 2. Security personnel <ul style="list-style-type: none"> • Go to alert location and investigate the threat. 3. Family members, Visitors, Neighbors <ul style="list-style-type: none"> • Stay away from threat location. |
| 2(c). For each task above: i. What sub-tasks make up this task? | 1. Communication |
| 2(c). For each task above: ii. What constraints are there for this task? | Messages not delivered. |
| 2(d). Which system/people in the company does this person rely upon? | <ol style="list-style-type: none"> 1. Security manager – relies on intruder detection system(services) 2. Security personnel – rely on security manager 3. Family members, visitors, neighbors – rely on security manager |
| 2(e). Who else in the company relies upon this person? | None |
| 2(f). What knowledge does this person require to perform his tasks correctly? | 1. Security manager – list of personnel and personal information in the company |
| 2(g). What resources, existing and new, are required by this person in fulfilling his position? | 1. Image and contact number. |
| 3. What code of behaviour must be observed by all of your employees? (a) Are there other codes of behaviour for certain positions, and what are they? | None |
| 4. What other rules and regulations must your company adhere to? | None |

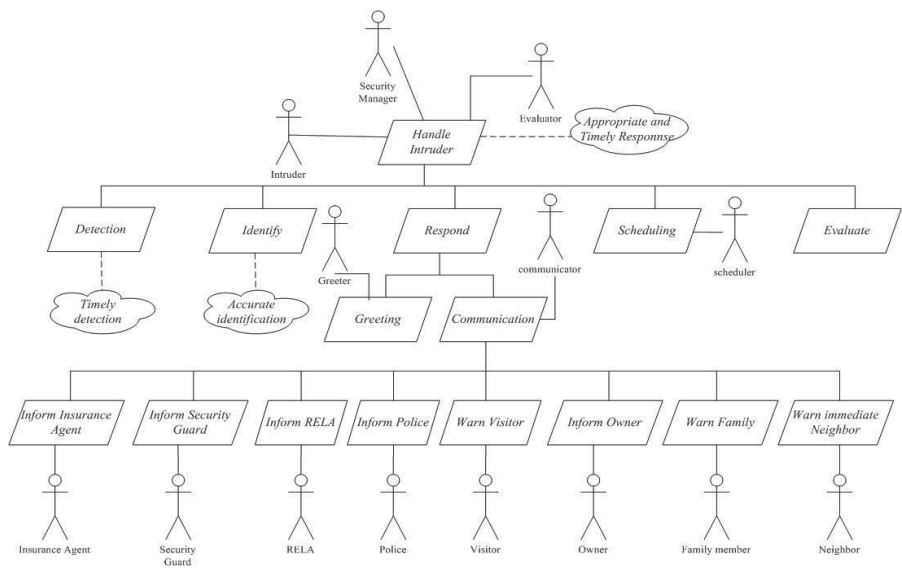


Figure 1. The goal model for intruder handling.

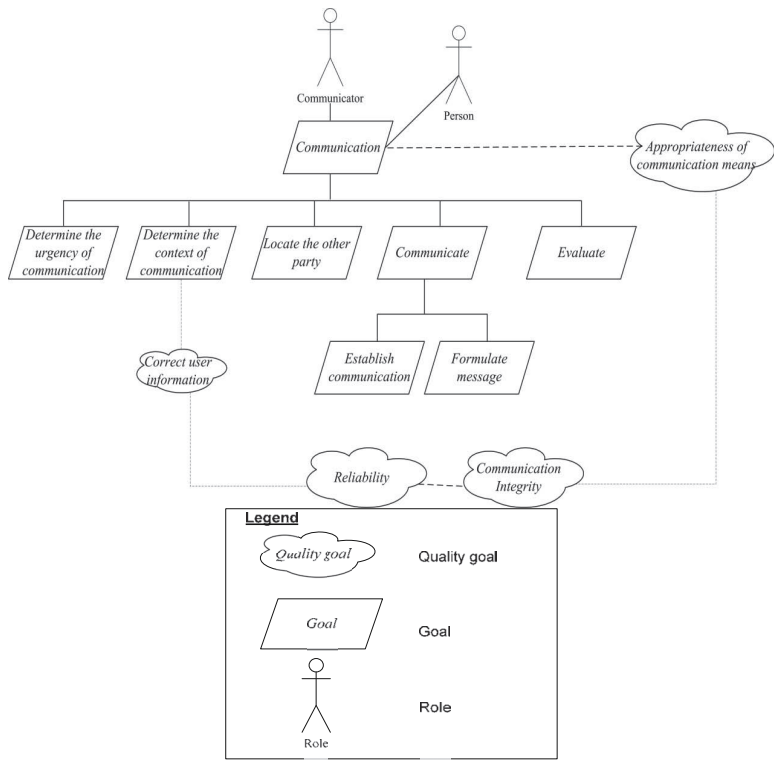


Figure 2. Goal model for communication.

Table 2. Role model for security manager.

| Role Name | Security Manager |
|------------------|---|
| Description | To identify and respond to an intruder detected in the house. |
| Responsibilities | Detect the occurrence of a person in the environment. Take an image of the person. Compare the image against the database of known people. Verify with the owner the identity of the person detected in the environment. Contact the police and send the capture image to them. Notify each visitor expected that day to stay away. Inform the owner that the police are on their way and the visitors have been warned not to enter the house. |
| Constraints | System needs to be provided with photo of the owner, family members, and visitors in advance. The camera needs to be installed in a place where a subject is easily detected and image can be captured clearly. The owner, family members, and visitors must have a communication device in order to receive messages. |

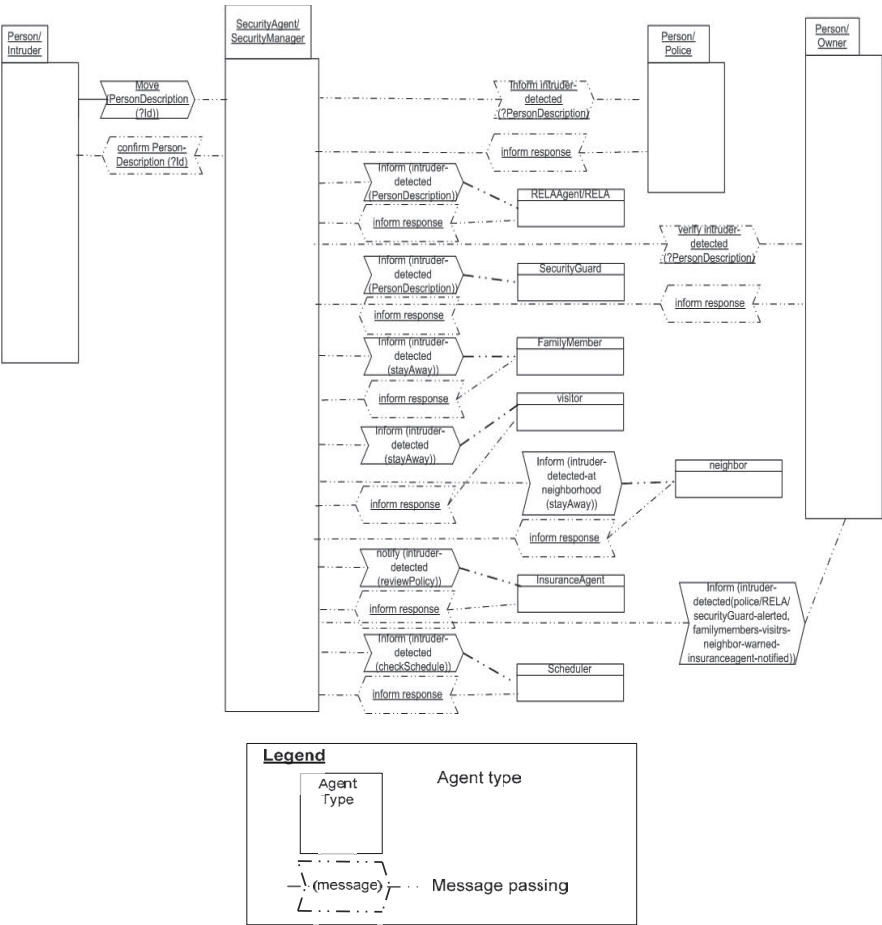


Figure 3. Interaction-frame diagram for intruder handling.

In general, the security manager will first start the face recognition service to monitor the changes of the environment, which includes extra object detected in the environment. Once the changes are identified, the security manager will act as a communicator to establish communication with other roles/persons as shown in Figure 2 with the intention to alert them. The "Person" modelled in this system refers to the visitor, owner, family members, police, security guard, RELA, neighbor, insurance agent and any other relevant personnel.

Stage II: modelling roles. Various roles are required in achieving the goals within the intruder handling. To exemplify it, we present a role model of security manager. Table 2 describes the role model for the security manager.

The model presents its responsibilities and constraints that may restrain the operations while serving these responsibilities.

Stage III: modelling interaction. Interaction frame diagram is used to generalize the types of action events that occur between agents and other types of agents. The interaction-frame diagram models the possible interactions between agents for both physical interactions and communication. The order in which the agents interact or the conditions under which one or another alternative interaction occur are not mentioned. Figure 3 presents the interaction-frame diagram for intruder handling.

After determining the agent types, we can now capture the interactions between agents of those types with the interaction model through an in-

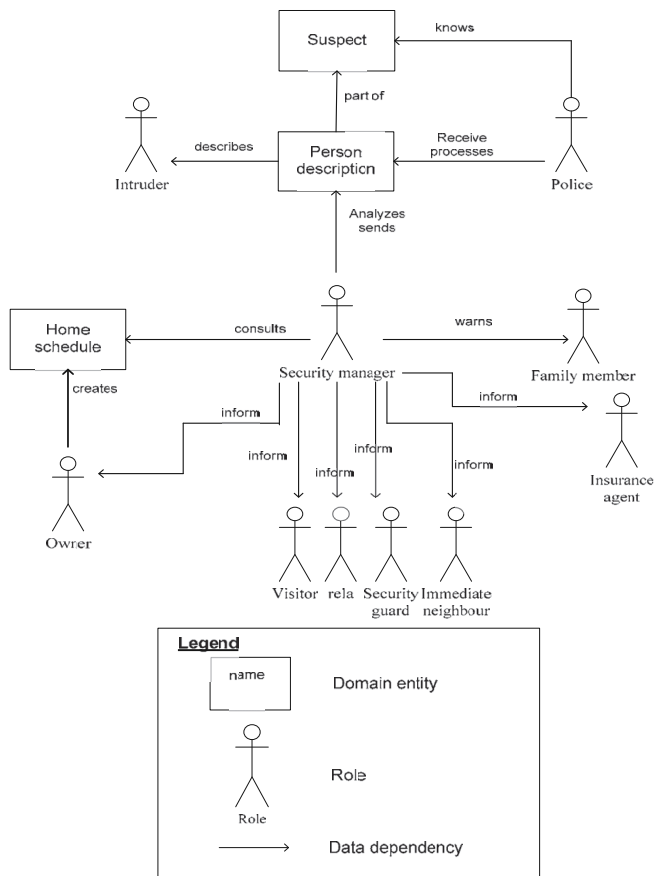


Figure 4. The domain model for intruder handling.

teraction-frame diagram. Interactions can be extracted from the responsibilities documented in the role schemas. Figure 3 consists of two interaction frames: between the agent of a person (intruder) and the security manager, and between the agent of the security manager and the persons (house owner, police, security guard, RELA, insurance agent, family members, immediate neighbor and visitor(s)). Messages in interaction frames have two modalities: "inform" and "confirm". A message of the "inform" modality serves to alert another agent about an incident. For example, when an unknown person is detected, the security manager will send a message to inform the police agent about that. The police agent will then confirm the message and respond to the security manager accordingly.

Stage IV: modelling domain knowledge. Figure 4 shows the domain model that describes

the relationship between different entities or roles when an event has transpired. The PersonDescription domain entity caches the visual information captured about the person detected by the security manager. The security manager compares and matches the captured image against his database. The person is regarded as an intruder if the system fails to identify him. In this case, the person's description will be forwarded to the police, who may be able to identify the concrete suspect. To identify those who are authorized to be in the house such as plumbers or electricians, the security manager will consult the house schedule in the HouseSchedule domain entity. HouseSchedule domain entity stores the start and end times of various activities that are taking place in the house, such as visits by friends and colleagues, family celebrations, and calls by service people. The schedule is created by the owner.

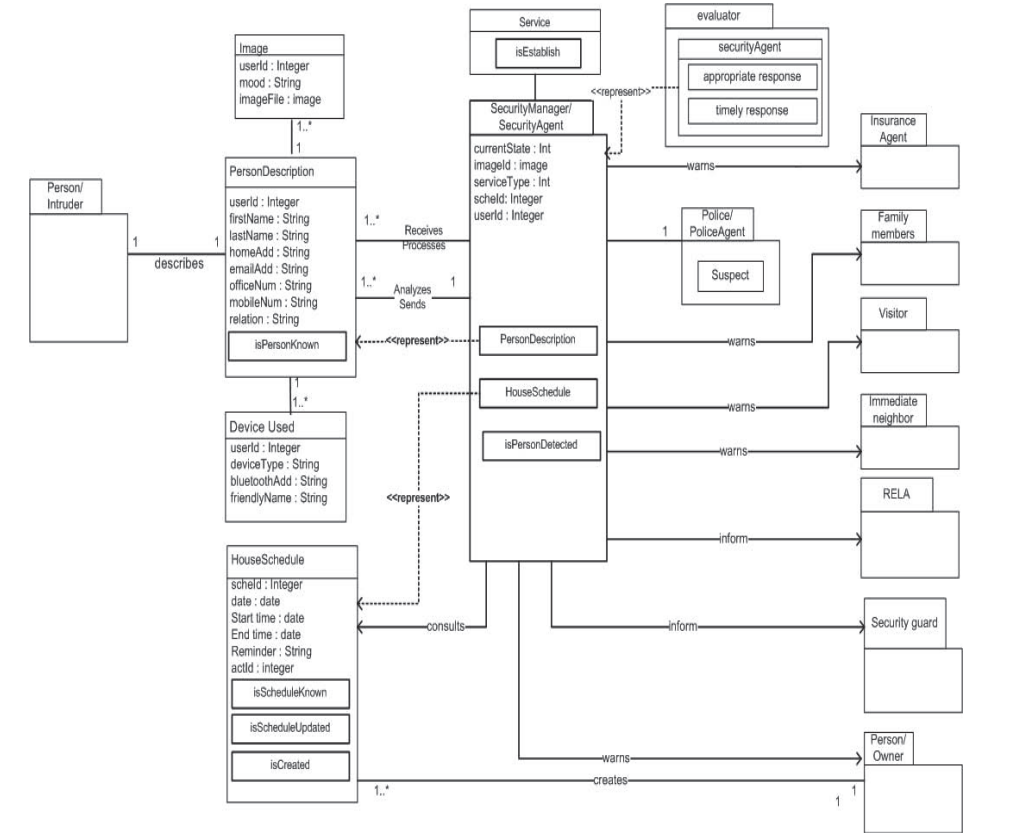


Figure 5. Knowledge model for intruder handling.

the intruder detection process, face recognition function is deployed to detect the changes of the environment, for instance, in the living room of a house. When a change in the environment is detected with the presence of an unidentified object or person, the securityAgent is notified and the communication service is established.

Stage VII: modelling interaction between agents. The sequence of interactions between various types of agents in handling intruder is modelled in Figure 6. The intruder-handling function is activated upon the detection of object changes or movement by the sensors. This will trigger the face recognition function to capture the image to determine the threat encountered. If the object is unidentifiable, the system will send an 'intruder' event notification to the SecurityManagerAgent. Then, the SecurityManagerAgent will in turn send an 'intruder detected' message to the policeAgent, RELA, SecurityGuard, the house owner and alert the visitors, neighbor, family members, in order to stay away from the threat. To ensure a quick response, the policeAgent will automatically assign any police within the vicinity to the scene to conduct further investigation. The RELA and SecurityGuard will also be sent to the location to assist in the investigation. During this incident, the policeAgent will commence communication with the house owner to update him.

Stage VIII: modelling scenario model. Stage VIII presents a more detailed list of actions to handle the scenario. This artifact will be used in modelling the agent's behaviour using agent-oriented relationship (AOR) behavioural diagram. A scenario model consists of a sequence of steps labelled with its type, name, role, description, and the data it accesses. Each step represents a goal, action, percept, or sub-scenario. A sample scenario for handling intruder is introduced in Table 3. The start of the step is shown in the main scenario model and the end of the step is the last step of sub-scenario. When an object/person enters the space within the camera's view, the detector will capture the image of the object/person (scenario 2) and then analyze or identify it using the face recognition program (scenarios 3, 4, and 5). It then compares the captured image against the database to identify the person. If the person is unrecognizable, the security manager will notify the house owner of the unknown person in the house. At the same time, the security manager will send a message

to the enforcement officer or agent to alert them (scenario 7), and warn the family members, visitors and immediate neighbor to stay away (scenario 8). The security manager will then update the house owner on the actions taken (scenario 9).

Stage VIII: modelling agent behaviour. Figure 7 shows a simplified version of the behavioural model for intruder handling. The outermost activity is started by rule R1, which is triggered by an action event labelled as *move (?PersonDescription)*. This modelling construct represents a physical movement of an object or person detected by the security agent in the form of event. Precision of the sensors is not of a concern at this stage of the system engineering process. Rule R1 also creates an instance of the *PersonDescription* object type within the security agent to store important data of the person. "Detect person" activity starts an "Identify intruder" sub-activity that triggers rule R2. This rule verifies the identity of the person through *isKnown(PersonDescription)* function. If the person is unidentifiable, the activity "Respond" is executed. The "Respond" activity consists of a series of sub-activities which include alerting the relevant parties. The activity types modelled in the behavioural model should correspond to the goals illustrated in the goal model in Figure 1. This implies that an activity should achieve the goal set at the preliminary design stage. For example, the "Respond" activity should effectuate the actions to achieve the "Respond" goal. R3 and R4 are reaction rules in Figure 7 that respond to the sub-activities by initiating communication through messages sent to the relevant recipients.

Progressing forward from conceptual models in PIM to more concrete models of PSM, next section will demonstrate transformation and derivation of the programming constructs based on the models presented in this section.

5.3. Platform Specific Modelling and Design (PSM) of Intruder Detection Scenario

The PSM layer is the lowest level of the system design, moving towards implementation. This level provides design details that "specify how the system is to be implemented in a specific platform, architecture, and tool or programming

Table 3. The scenario model for intruder handling.

| SCENARIO 1 | | | | | |
|--|-------------------|--|------------------------------------|------------------------|------------------|
| Goal | Handling Intruder | | | | |
| Initiator | Security Manager | | | | |
| Trigger | Intruder | | | | |
| Failure | Home break-in | | | | |
| DESCRIPTION | | | | | |
| Condition | Step | Activity | Agent types and roles | Resources | Quality goals |
| When an object/ person enters a space that is within the camera's view | 1 | Person walks into a room. The system detects that there are changes in the environment. (Scenario 2) | Detector/Person Detector/Person | Camera/ PC/internet | Timely detection |
| | 2 | The system identifies the person's presence in the environment. (Scenario 3, 4, or 5) | Identifier/ Detector/Person | | |
| | 3 | The system checks the house schedule. (Scenario 6) | scheduler | | |
| | 4 | Take an image of the person. | Security Manager | | |
| | 5 | Compare the image against the database of known persons. | Security Manager | | |
| | 6 | Inform the owner there is an unknown person in the environment. | | | |
| | 7 | Contact the law enforcement officer and send the image to them. (Scenario 7) | Communicator/ Person | | |
| | 8 | Send a stay away message to family members, visitors and immediate neighbour. (Scenario 8) | Communicator/ Person | | |
| | 9 | Inform the owner that the police and other security officers are on their way and that their family members, visitor and neighbour had been warned. (Scenario 9) | | | |

language" [22]. It facilitates conversion from the PIM models to derive the skeleton programs or program templates that reflect the structure of the system. The design model is transformed from the PIM level based on the guidelines proposed by the author [32].

Stage IX: JADE specific information model. Figure 8 shows the snippets of JADE specific model for PersonDescription information type that was modelled in Figure 5 as the knowledge model of intruder handling. As mentioned be-

fore, the agent types IntruderHandler, Police, Owner, Visitor, Family Members, Security guard, RELA, Neighbour, and Insurance agent are represented as the respective agent types of JADE. The shared object type IntruderDescription and the private object type Person are implemented as the corresponding Java object types. The predicate isKnown is converted to a method attached to the object type Person. The information type of PersonDescription is transformed into the JADE object class while the attributes of the information type are declared

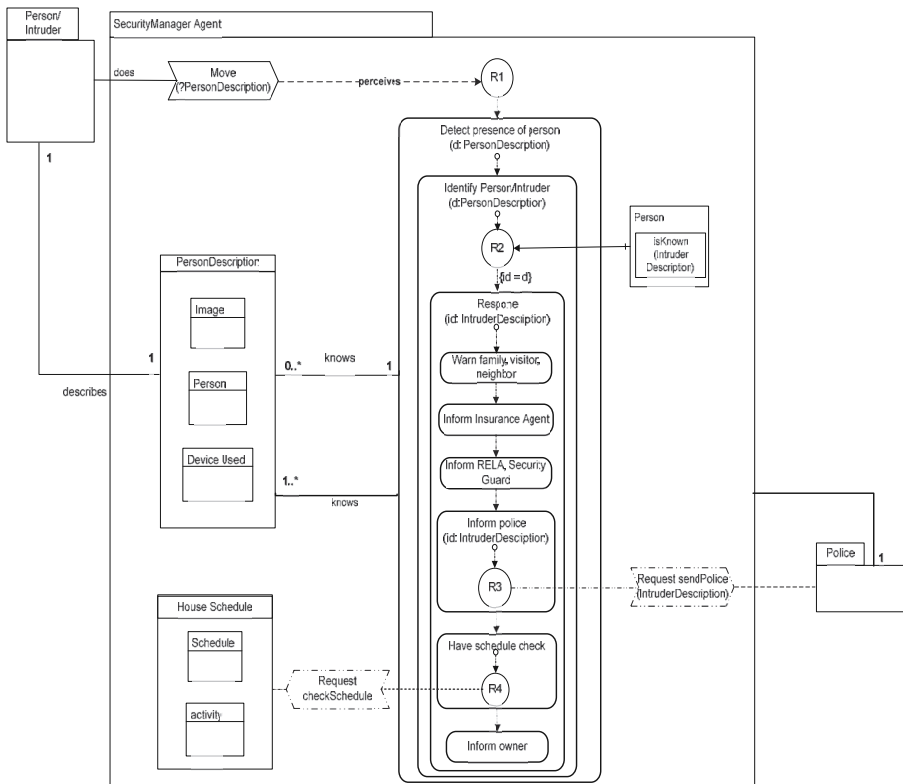


Figure 7. The behaviour model for the intruder-handling.

```

import jade.content.*;
public class PersonDescription implements Concept {
    private int UserId;
    private String firstName;
    private String lastName;
    private String homeAdd;
    public int getUserId() {
        return UserId;
    }
    public String getfirstName() {
        return firstName;
    }
    public String getlastName() {
        return lastName;
    }
    //...others information model
}

```

Figure 8. JADE specific information model for part of the information type of PersonDescription.

as String data type. The relationship of the PersonDescription linked to the image and device used will be implemented with JADE object class such as Image and DeviceUsed. Figure 9 shows part of the face recognition coding that captures image, analyzes it and informs the SecurityManagerAgent when an unknown person is detected. It forms part of the service model in JADE.

Stage XI: JADE specific interaction model.

Figure 10 shows a JADE specific interaction model, which is implemented as intruder handling ontology. It models the ontology that is shared among the agents during agent interaction by registering the type and properties of the entities. The ontology scheme consists of an action schema and an information type schema to specify the action type and concepts involved while handling intruder detection. The action schema models the CreatePerson action to create a new person object with detail parameters of information types as specified in the PersonDescription and the domain type. The object members that are set to mandatory indicate that all of the attributes must be set as compulsory during the agent communication. The JADE specific interaction model is a transformation of both the interaction model and knowledge model that are presented in Figures 8 and 9 respectively.

Stage XII: JADE specific behavioural model.

Figure 11 partially shows PSM behavioural model in respond to intruder detection. Once an intruder is detected by the face recognition service, the SecurityManagerAgent will send an informed message labeled as “Intruder Alert” to all agents registered on JADE Yellow Pages.

6. Implementation of Intruder Handling Scenario

We presented the modelling of intruder handling in the Section 5. This section presents the implementation of intruder handling in JADE, a multi-agent programming platform. Figure 12 shows part of the physical distribution and agents of the VSS. All the nodes are connected wirelessly with a static IP for face recognition service (192.168.1.50), intruder detection system (IDS) Main Server (192.168.1.5), Police Main Server (192.168.1.4) and dynamic IP for agents involved in the scenario.

A webcam is connected to the face recognition system. An IDS Main_Server will act as context tier that performs all the work of detecting changes in the environment and interpreting the data captured to determine the context information by securityAgent. When a person enters the vicinity under scrutiny, the image recognizer

```
//some opencv human detection code
void FrameGrabber_Standard(object sender, EventArgs e){
//Get the current frame form capture device
currentFrame = grabber.QueryFrame().Resize(320, 240, Emgu.CV.CvEnum.INTER.CV_INTER_CUBIC);
//Convert it to Grayscale
if (currentFrame != null){
gray_frame = currentFrame.Convert<Gray, Byte>();
//Face Detector
MCvAvgComp[][] facesDetected = gray_frame.DetectHaarCascade(Face, 1.2, 10,
Emgu.CV.CvEnum.HAAR_DETECTION_TYPE.DO_CANNY_PRUNING, new Size(50, 50));
//Action for each element detected...
//Draw the label for each face detected and recognized
//add unknown face
ActivateAgent();
//..other processing code
```

Figure 9. Service model for object detection in OpenCV and the sending of intruder detection event to SecurityAgent.

```

package ontologies;
import jade.content.onto.*;
import jade.content.schema.*;

public class IntruderOntology extends Ontology implements IntruderVocabulary {
    private static final long serialVersionUID = 1L;
    // ----->The name identifying this ontology
    public static final String ONTOLOGY_NAME = "Intruder-Ontology";
    // ----->The singleton instance of this ontology
    private static Ontology instance = new IntruderOntology();
    // -----> Method to access the singleton ontology object
    public static Ontology getInstance() { return instance; }
    // Private constructor
    private IntruderOntology() {
        super(ONTOLOGY_NAME, BasicOntology.getInstance());
    }
    try {
    // ----- Add Concepts
    // Person
    ConceptSchema cs = new ConceptSchema(PERSON);
    add(cs, UserProfile.class);
    cs.add(PERSON_ID, (PrimitiveSchema) getSchema(BasicOntology.INTEGER),
        ObjectSchema.MANDATORY);
    cs.add(PERSON_FIRST_NAME, (PrimitiveSchema) getSchema(BasicOntology.STRING),
        ObjectSchema.MANDATORY);
    // House Schedule
    add(cs = new ConceptSchema(HOUSE_SCHEDULE), HouseSchedule.class);
    cs.add(HOUSE_SCHEDULE_ID, (PrimitiveSchema) getSchema(BasicOntology.INTEGER),
    // ----- Add AgentActions
    catch (OntologyException oe) {
        }
    }
}

```

Figure 10. JADE specific information model for the intruder handling ontology.

(Web camera) detects the presence of a person and captures his face. The face recognition service extracts the data from the image and sends the contextual information pertaining to the person detected to the securityAgent. Once the person is identified as an intruder, the securityAgent sends a message to the ownerAgent to further validate the identity of the person. When the ownerAgent is unable to recognize the person, the securityAgent will notify the policeManagerAgent and securityGuardAgent or RELA of the intrusion. Thereafter, the securityAgent contacts the scheduled visitorAgent, family members, and neighbourAgent to warn them to stay away. Meanwhile, the policeManagerAgent assigns the nearest police officer to the

scene by comparing the distance in between the respective police officer with the scene. Figure 13 presents the communication between securityManagerAgent, visitorAgent, familyAgent and RelaOfficerAgent during the intruder handling. Meanwhile, Figure 14 presents the sample of communication between policeManagerAgent and respective policeAgent.

7. Conclusion

The main contribution of this paper is to introduce AOM for designing and developing an agent-oriented video surveillance system. It attempts to validate the feasibility of this metho-

dology in developing a complex socio-technical system in a qualitative manner. With AOM, we can engineer a distributed video surveillance system in a systematic manner. The comprehensive elaboration and decomposition of the models from different perspectives can serve as a guideline for novice developer during the development of multi-agent system. With AOM, there is a possibility to reuse the models in

maintaining the VSS system. In addition, those models can be reused in other similar kind of projects in which more investigation is needed in the near future. In addition, the integration of AOM models with other models is worth to explore. This is because some of the AOM models are lacking explicit expression. For example, the goal model has failed to model the goal dependency in details. On the other hand,

```

import jade.core.Agent;
import jade.core.behaviours.CyclicBehaviour;
import jade.domain.AMSService;
import jade.domain.FIPAAgentManagement.AMSAgentDescription;
import jade.domain.FIPAAgentManagement.SearchConstraints;
import jade.lang.acl.ACLMessage;
public class SecurityManagerAgent extends Agent{
private static final long serialVersionUID = 1L;
protected void setup() {
    AMSAgentDescription [] agents = null;
    try {
        SearchConstraints c = new SearchConstraints();
        c.setMaxResults( new Long(-1));
        agents = AMSService.search( this, new AMSAgentDescription (), c );
    }
    catch (Exception e) {
        System.out.println( "Problem searching AMS: " + e );
    }
    //once the intruder detection event is trigger, the securityAgent will send a messages to others
    ACLMessage msg = new ACLMessage(ACLMessage.INFORM);
    msg.setContent( "Intruder Alert" );

    for (int i=0; i<agents.length;i++)
        msg.addReceiver( agents[i].getName() );
    send(msg);
    addBehaviour(new CyclicBehaviour(this){
        private static final long serialVersionUID = 1L;
        public void action() {
            ACLMessage msg= receive();
            if (msg!=null)
                System.out.println( "== " + "-> " + msg.getContent() + " from "+ msg.getSender().getName() );
            msg=null;
        }
    });
}
}

```

Figure 11. JADE behaviour model for securityAgent upon the detected intruder.

the behaviour model has failed to present the overall capability of an individual agent. Based on our experience, presenting the overall goal and goal dependency and capabilities of the agent are needed to ease debugging and discussion with the stakeholders, in which more to explore in future. Also, more example can be explored like abnormal human behaviour as stated in [36].

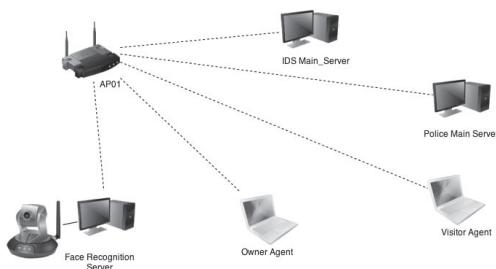


Figure 12. Simulation of intruder handling.

```
Agent container Container-4@192.168.1.50 is ready.
==> Intruder Alert from SecurityManager@192.168.1.5:1099/JADE
==> noted from Visitor@192.168.1.5:1099/JADE
==> Will stay away from the site. from Visitor@192.168.1.5:1099/JADE
==> noted from Family@192.168.1.5:1099/JADE
==> noted from RelA Officer-1@192.168.1.5:1099/JADE
==> Will took appropriate action from Family@192.168.1.5:1099/JADE
==> Will investigate immediately from RelA Officer-1@192.168.1.5:1099/JADE
```

Figure 13. Communication between securityManagerAgent, visitorAgent, familyAgent and RelAOfficerAgent.

```
Agent container Container-3@192.168.1.50 is ready.
Trying to assign police officer to MJC
Found the following police agents:
Police@192.168.1.4:1099/JADE
Police@192.168.1.4:1099/JADE
Intruder alert case at MJC was assign to Police@192.168.1.4:1099/JADE
Distance from location : 12
Intruder alert from Police_HQ@192.168.1.4:1099/JADE terminating.
```

Figure 14. Communication between policeManagerAgent with the respective policeAgent.

References

- [1] N. Ejaz *et al.*, "A Collaborative Multi-Agent Framework for Abnormal Activity Detection in Crowded Areas", in *Int. Journal of Innovative Computing, Information and Control*, vol. 8, no.6, 2012.
- [2] C. Wai Shiang *et al.*, "Software Agent Negotiation Development: An Experience Report", *Proc. of the Int. Conf. on Intelligent Systems Design and Applications*, pp. 881–886, 2006.
<http://dx.doi.org/10.1109/ISDA.2006.253728>
- [3] X. Liu and K. Wang, "Active Power Control Simulation Platform Research of Wind Farm Based on Multi-Agent" in *MATEC Web of Conferences*, vol. 22, 2015, EDP Sciences.
<http://dx.doi.org/10.1051/mateconf/20152202004>
- [4] C. Wai Shiang *et al.*, "Designing a Shared Single Display Education Application through Interactive Patterns", in *Journal of Software Engineering and Applications*, vol. 7, no. 13, pp. 1074, 2014.
<http://dx.doi.org/10.4236/jsea.2014.713095>
- [5] C. Wai Shiang *et al.*, "Long Lamai Community ICT4D E-Commerce System Modelling: An Agent-Oriented Role-Based Approach", in *The Electronic Journal of Information Systems in Developing Countries*, vol. 75, 2016.
- [6] D. Chen *et al.*, "An Agent-Based Model for Consumer-to-Business Electronic Commerce", in *Expert Systems With Applications*, vol. 34, no. 1, pp. 469–481, 2008.
<http://dx.doi.org/10.1016/j.eswa.2006.09.020>
- [7] A. J. C. Trappey *et al.*, "Development of an Intelligent Agent System for Collaborative Mold Production with RFID Technology", in *Robotics and Computer Integrated Manufacturing*, vol. 25, no. 1, pp. 42–56, 2009.
<http://dx.doi.org/10.1016/j.rcim.2007.06.002>
- [8] J. A. Sánchez *et al.*, "An Agent-Based Approach to the Construction of Floristic Digital Libraries", in *Proc. of the 3rd ACM Conf. on Digital Libraries*, New York, NY, USA, 1998.
<http://dx.doi.org/10.1145/276675.276699>
- [9] P. E. Dossou *et al.*, "The Use of Multi-Agent Systems for Improving a Logistic Platform in a GRAI Environment", *Highlights of Practical Applications of Agents, Multi-Agent Systems, and Sustainability-The PAAMS Collection*, Springer International Publishing, pp.126–135, 2015.
http://dx.doi.org/10.1007/978-3-319-19033-4_11
- [10] S. Gao and D. Xu, "Conceptual Modeling and Development of an Intelligent Agent-Assisted Decision Support System for Anti-Money Laundering", in *Expert Systems With Applications*, vol. 36, no.2, pp. 1493–1504, 2009.
<http://dx.doi.org/10.1016/j.eswa.2007.11.059>
- [11] Z. Ren and C. J. Anumba, "Multi-Agent Systems in Construction-State of the Art and Prospects", in *Automation in Construction*, vol. 13, no. 3, pp. 421–434, 2004.
<http://dx.doi.org/10.1016/j.autcon.2003.12.002>
- [12] R. Gowri *et al.*, "Development of Multi-Agent System for Fire Accident Detection Using Gaia Methodology", in *Int. Journal of Computer Science and Information Security*, vol. 8, no. 1, pp. 190–194, 2010.

- [13] R. A. Gonzalez, "Developing a Multi-Agent System of a Crisis Response Organization", in *Business Process Management Journal*, vol. 16, no. 5, pp. 847–870, 2009.
<http://dx.doi.org/10.1108/14637151011076502>
- [14] L. Chee Wyai *et al.*, "Engineering Sustainable Software: A Case Study from Offline Computer Support Collaborative Annotation System", in *IEEE 9th Malaysian Software Engineering Conference (MySEC)*, 2015, pp. 272–277.
<http://dx.doi.org/10.1109/MySEC.2015.7475232>
- [15] C. Wai Shiang *et al.*, "Agent Oriented Requirement Engineering for Lake Mathematical Modelling: Preliminary Study", *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, vol. 8, no. 2, pp. 5–10, 2016.
- [16] A. Oluyomi *et al.*, "Description Templates for Agent-Oriented Patterns", in *The Journal of Systems & Software*, vol. 81, no. 1, pp. 20–36, 2008.
<http://dx.doi.org/10.1016/j.jss.2007.06.020>
- [17] A. Sturm and O. Shehory, "Agent-Oriented Software Engineering: Revisiting the State of the Art", in *Agent-Oriented Software Engineering: Reflections on Architectures, Methodologies, Languages and Frameworks*, pp. 13–26, 2014.
http://dx.doi.org/10.1007/978-3-642-54432-3_2
- [18] Q. N. N. Tran and G. Low, "MOBMAS: A Methodology for Ontology-Based Multi-Agent Systems Development", in *Information and Software Technology*, vol. 50, no. 7–8, pp. 697–722, 2008.
<http://dx.doi.org/10.1016/j.infsof.2007.07.005>
- [19] P. Koutsabasis and J. Darzentas, "Methodologies for Agent Systems Development: Underlying Assumptions and Implications for Design", in *AI & Society*, vol. 23, no. 3, pp. 379–407, 2009.
<http://dx.doi.org/10.1007/s00146-007-0110-9>
- [20] F. Zambonelli *et al.*, "Organisational Rules as an Abstraction for the Analysis and Design of Multi-Agent Systems", in *Int. Journal of Software Engineering and Knowledge Engineering*, vol. 11, no. 3, pp. 303–328, 2001.
<http://dx.doi.org/10.1142/S0218194001000505>
- [21] K. H. Dam and M. Winikoff, "Comparing Agent-Oriented Methodologies", in P. Giorgini, B. Henderson-Sellers and M. Winikoff (Ed.), *Agent-Oriented Information Systems*, Springer LNAI 3030, pp. 78–93, 2004.
- [22] L. Sterling and K. Taveter, "The Art of Agent Oriented Modelling", Cambridge, MA: MIT Press, 2009.
- [23] J. M. Gascueña *et al.*, "Knowledge Modelling through Computational Agents: Application to Surveillance Systems", in *Expert Systems, The Journal of Knowledge Engineering*, vol. 28, no. 4, 2011.
<http://dx.doi.org/10.1111/j.1468-0394.2011.00609.x>
- [24] J. Garcia *et al.*, "Agent-Based Coordination of Cameras", in *Int. Journal of Computer Science and applications*, vol. 2, no. 1, pp. 33–37, 2005.
- [25] D. Vallejo *et al.*, "A Multi-Agent Architecture for Supporting Distributed Normality-Based Intelligent Surveillance", in *Engineering Applications of Artificial Intelligence*, vol. 24, no. 2, pp. 325–340, 2011.
<http://dx.doi.org/10.1016/j.engappai.2010.11.005>
- [26] F. Castanedo *et al.*, "Data Fusion to Improve Trajectory Tracking in a Cooperative Surveillance Multi-Agent Architecture", in *Information Fusion*, vol. 11, pp. 243–255, 2010.
<http://dx.doi.org/10.1016/j.inffus.2009.09.002>
- [27] J. Pavón *et al.*, "Development of Intelligent Multisensor Surveillance Systems with Agents", in *Robotics and Autonomous Systems*, vol. 55, no. 12, pp. 892–903, 2007.
<http://dx.doi.org/10.1016/j.robot.2007.07.009>
- [28] J. M. Gascueña *et al.*, "Model-to-Model and Model-to-Text: Looking for the Automation of VigilAgent", in *Expert Systems: The Journal of Knowledge Engineering*, vol. 31, no. 3, pp. 199–212, 2014.
<http://dx.doi.org/10.1111/exsy.12023>
- [29] J. M. Gascueña and A. Fernández-Caballero, "On the Use of Agent Technology in Intelligent, Multisensory and Distributed Surveillance", in *The Knowledge Engineering Review*, vol. 26, no. 2, pp. 191–208, 2007.
<http://dx.doi.org/10.1017/S0269888911000026>
- [30] T. Juan *et al.*, "ROADMAP: Extending the Gaia Methodology for Complex Open Systems", *Proc. of the 1st Int. Joint Conf. on Autonomous Agents and Multiagent Systems: part 1*, pp. 3–10, 2002.
<http://dx.doi.org/10.1145/544741.544744>
- [31] D. Wilmann and L. Sterling, "Guiding Agent-Oriented Requirements Elicitation: HOMER", *Proc. of the 5th Int. Conf. on Quality Software*, 419–424, 2005.
<http://dx.doi.org/10.1109/QSIC.2005.34>
- [32] C. Wai Shiang *et al.*, "Task Knowledge Patterns reuse in Multi-Agent System Development", *Proc. of the 13th Int. Conf. on Principles and Practice of Multi-Agent Systems*, Kolkata, India, 2010.
http://dx.doi.org/10.1007/978-3-642-25920-3_33
- [33] P. Bresciani *et al.*, "Tropos: An Agent-Oriented Software Development Methodology", in *Autonomous Agents and Multi-Agent Systems*, vol. 8, no. 3, pp. 203–236, 2004.
- [34] J. M. Gascueña and A. Fernández-Caballero, "Agent-Oriented Modeling and Development of a Person-Following Mobile Robot", in *Expert Systems with Applications*, vol. 38, no. 4, pp. 4280–4290, 2011.
<http://dx.doi.org/10.1016/j.eswa.2010.09.096>

- [35] V. Marik and D. McFarlane, "Industrial Adoption of Agent-Based Technologies", in *IEEE Intelligent Systems*, pp. 27–35, 2005.
<http://dx.doi.org/10.1109/MIS.2005.11>
- [36] A. Morozov *et al.*, "Intelligent Visual Surveillance Logic Programming: Implementation Issues", in *Workshop on Implementation of Constraint and Logic Programming Systems and Logic-based Methods in Programming Environments*, pp. 31, 2014.

Received: December 2015

Revised: November 2016

Accepted: November 2016

Appendix C

Msury Mahunnah, Kuldar Taveter, and Raimundas Matulevicius, "An Empirical Evaluation of the Requirements Engineering Tool for Sociotechnical Systems.," in 26th IEEE International Requirements Engineering Conference Workshops (REW). IEEE, 2018.

An Empirical Evaluation of the Requirements Engineering Tool for Socio-Technical Systems

Msurry Mahunnah
Department of Software Science
Tallinn University of Technology
Tallinn, Estonia
msurry.mahunnah@ttu.ee

Kuldar Taveter
Department of Software Science
Tallinn University of Technology
Tallinn, Estonia
kuldar.taveter@ttu.ee

Raimundas Matulevičius
Institute of Computer Science
University of Tartu
Tartu, Estonia
raimundas.matulevicius@ut.ee

Abstract—One of the major problems of requirements engineering is the lack of sufficient empirical evidence that evaluates the benefits of modelling tools for Model-Driven Engineering (MDE). In this paper, we report on the results of empirical study that compares the modelling effort and effectiveness of the novel software tool for modelling requirements of sociotechnical systems against modelling on paper. We have asked 8 persons who received 2 different treatments – modelling on software against modelling on paper to create 2 requirements models – goal and domain models – for 2 different case studies. The study finds that modelling effort with a software tool nearly equals to modelling effort on paper while modelling effectiveness with a tool is higher than modelling effectiveness on paper. The major limitation of this study is the use of students as participants and the use of small sample size. In the future work, we will conduct another empirical study with a large sample size of professionals that aims to increase the confidence in the results obtained from this empirical study.

Index Terms—Requirements, sociotechnical system, model-driven engineering, tools.

I. INTRODUCTION

A sociotechnical system is a software intensive system that has defined operational processes followed by human operators and which operates within an organization [1]. For example, if a computer game does not feel fun, we will not play it; if an ecommerce website does not feel trustworthy (irrespective of the actual security) we will not purchase from it; and if a social networking application does not feel engaging we will not use it [2]. Therefore, a social aspect of the system plays an important role to complement a technical aspect of system and form a sociotechnical system [1]. These aspects can be elaborated into different levels and perspectives. In [3], Whitworth suggest four levels that describe sociotechnical systems: physical, information, personal, and group level. In [4], da Conceição, *et al.* distinguish seven abstraction levels of sociotechnical systems in the maritime domain: natural environment, and reactive, automated reactive, proactive, planning scheduling, planning strategic, and political-economic levels. In addition to the abstraction levels, [5] proposes six perspectives of sociotechnical systems, which are orthogonal to the abstraction levels: goals, people, technologies, physical infrastructure, cultural assumptions, and processes, and working practices.

To cope with engineering sociotechnical systems considering the multitude of diverse abstraction levels and

perspectives, Baxter and Sommerville [6] emphasize that appropriate models and abstractions should be used for representing sociotechnical considerations. Consequently, in [7], the Agent-Oriented Modelling (AOM) methodology is proposed and elaborated in [8]–[10]. It utilizes goal, role, organization and domain models during requirements engineering phase which is supported by the novel Agent-Oriented Modelling for Sociotechnical System (AOM4STS)¹ software tool. The AOM4STS software tool aims to reduce the effort and increase the effectiveness of the current practice of applying the AOM methodology by modelling on paper. However, the gap exists in the empirical evidence that compares the effort and effectiveness between modelling with the AOM4STS tool against modelling on paper.

In this paper, we fill the identified gap by answering the following research question: To what extent does the novel AOM4STS software tool improve the process of requirements modelling on paper? To establish complexity-reducing separation of concerns, we deduce the following sub-questions: (i) To what extent is the modelling effort with the AOM4STS software tool different from modelling on paper? (ii) To what extent is the modelling effectiveness with the AOM4STS software tool different from modelling on paper?

To find answers to the research questions, we carried out modelling experiments following the guidelines of experimentation in software engineering [11]. Our goal is to empirically compare the effort and effectiveness of modelling process using the AOM4STS tool against modelling on paper.

The rest of this paper is structured as follows. Section II presents related work. Section III provides an overview of the AOM methodology and briefly describes models for requirements engineering of sociotechnical systems. Section IV presents key features of the AOM4STS software tool. Section V describes an experiment conducted for empirical evaluation of the tool. Section VI summarizes the results of the experiment. Finally, conclusions are drawn in Section VII.

II. RELATED WORK

The AOM methodology [7] stems from the Model-Driven Engineering (MDE) [12] paradigm that focuses on the systematic use of models as primary engineering artefacts throughout the system engineering lifecycle. Among the key benefits of

¹ goo.gl/Wr6q57

MDE paradigm are effective expression of domain concepts [13], decreasing system development time (effort), and improving system quality [14].

Despite the benefits of the MDE paradigm, various studies show that a domain-specific MDE language is not enough for industry-wide adoption and a tool supporting such language increases the complexity of the development process instead of diminishing it [15]. Elsewhere, Whittle et al. [16] interviewed 39 practitioners on tool-related issues affecting the adoption of MDE. The results of this study indicate that the complexity of the modelling tools is among the major issues hindering practical application of MDE. Moreover, the study [16] suggests the need for developing new software modelling tools that focus on early design stages, support creativity in modelling, and match the way people think rather than the other way round. Another study involved 15 MDE experts in a thought experiment to identify the biggest problems with current MDE technologies [17]. The results of this study found that steep learning curves and arduous user interfaces are among significant usability challenges to industry-wide adoption of MDE tools.

Considering the benefits of MDE languages and the challenges of using MDE tools, Gorschek et al. [18] conducted a survey with 3785 developers to find out the extent to which design models are used before actual coding. The results of this study found that design models are not used very extensively in industry. Moreover, in such companies where they are used, the notations are often *not* UML notations, the use of design models is informal and without tool support. Instead of relying on tools, the models are usually drawn on whiteboard or paper.

The findings from this review of related work point to the need of conducting research studies on MDE software tools to empirically compare claimed benefits of a modelling tool against modelling on a whiteboard or paper.

III. AGENT-ORIENTED METHODOLOGY FOR ENGINEERING SOCIOTECHNICAL SYSTEMS

The AOM methodology proposed in [7] and elaborated in [8]–[10] is centered on the notions of agent, goal, role, and domain entity. A sociotechnical system (STS) is defined in AOM as a system consisting of diverse active components – both human and man-made (software and robots) – that collaborate in designing and sustaining the sociotechnical system. We term such active components as agents, which form a distributed system. AOM is an approach for engineering complex sociotechnical systems where a problem domain is conceptualized in terms of the goals to be achieved by the system, the roles required for achieving them, and the domain entities embodying the required knowledge.

Agent-oriented models for problem domain analysis, which are used for representing the requirements, help to improve communication between information technology (IT) and non-IT experts during the requirements elicitation phase in the development. These models provide a high-level description of the system and use graphical notations to enable project stakeholders to obtain a common understanding about the system requirements. Table I outlines the objective of each agent-oriented model for problem domain analysis.

TABLE I. OVERVIEW OF MODELS FOR PROBLEM DOMAIN ANALYSIS

| ID | Model Name | Objective |
|----|------------------------|--|
| 1 | Goal Modelling | To represent functional and non-functional requirements of the system as goals and quality goals, respectively, roles required for achieving the goals, and relationships among all of them. |
| 2 | Role Modelling | To list responsibilities and constraints of each role in the system. |
| 3 | Organization Modelling | To show the types of relationships that exist between the roles of the system. |
| 4 | Domain Modelling | To represent the knowledge represented within the system by capturing the types of domain entities (knowledge items) and the relationships between the roles and domain entities. |

The scope of this paper includes goal and domain modelling. Role and organization modelling are not considered during the empirical study reported in Section V.

IV. AOM4STS TOOL SUPPORT

The AOM4STS tool [19] supports the AOM methodology which involves incremental refinement of models in an iterative manner. Therefore, consistency checking becomes a necessary feature of the AOM4STS tool to ensure that the modelling artefacts represented in Table I remain consistent with each other. The following subsections A to B briefly describe the two key features of the AOM4STS tool. Due to space limitation, other features of AOM4STS tool are not describes in this paper.

A. Information propagation

According to Table I, models in the AOM methodology are divided horizontally along three abstraction layers and vertically along three viewpoint perspectives. Considering this, during the modelling process, the AOM4STS tool propagates information vertically across abstraction layers and horizontally across viewpoint perspectives.

In the vertical information propagation, models for problem domain analysis act as input for platform-independent design models while platform-independent design models act as input for platform-specific design models and prototypes.

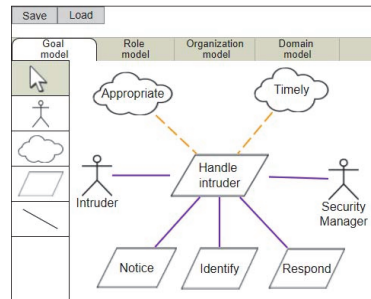


Fig. 1. Screenshot of the AOM4STS software tool

In the horizontal information propagation, AOM4STS propagates information across models of different viewpoint perspectives but within the same abstraction layer. The prob-

lem domain analysis layer contains four different models as outlined in Table I. The information in these models is propagated horizontally across the three viewpoint perspectives. For example, all the roles identified during goal modelling are horizontally propagated to role models, organization model, and domain model. Fig. 1 depicts a screenshot of the AOM4STS tool that describes the goal model of the Intruder Detection System [20].

B. Consistency checking

The AOM4STS tool continuously performs consistency checking to prevent certain errors from being made in the first place. The errors checked against are definition errors, simple typing errors, and violations of scope. The principle of detecting definition errors is that it is only possible to create a reference to an entity after the entity has been defined. For example, it is only possible to create a reference to a role in a domain model after that role has been defined in a goal model. Moreover, when the user deletes an entity, the tool deletes all references to the deleted entity. The effect is different when modelling on whiteboard or paper.

The principle of detecting simple errors allows users to create only syntactically correct connections between component types. The tool prevents all syntactically wrong connections and generates the corresponding error messages in the bottom frame of the tool containing user activity logs. For example, according to the AOM methodology, it is syntactically wrong to create a connection between a role and quality goal in the goal model.

Lastly, in preventing violations of scope constraints, the tool allows an analyst to neither increase nor decrease the scope of the project identified during the earlier modelling stages. In other words, once the goal modeller has defined the scope of the project, the role modeller, organization modeller, and domain modeller can only refine the requirements but not increase or decrease the scope of the project. This makes the AOM4STS tool suitable for an iterative (agile) modelling process that supports the AOM methodology [21].

V. EMPIRICAL EVALUATION

The AOM4STS tool, which has been presented in Section IV, is an online diagramming software tool that supports the methodology of requirements engineering for sociotechnical systems described in Section III. In this section, we present an empirical study for evaluating requirements modelling for a sociotechnical system with the help of the AOM4STS tool in comparison with modelling the requirements for the same sociotechnical system using pen and paper. The design of the experiments follows the guidelines by Wohlin et al. [11] on how to set up and document empirical studies in software engineering.

A. Experimental Design

This section describes the plan for the experiment that was followed during the empirical study.

1) Goal of the study

The goal of the empirical study was to compare software-based processes of modelling requirements for sociotechnical

system against paper-based processes of modelling the same requirements to find out if the benefits expected from using the AOM4STS tool were present when used by novice users of the AOM methodology and the AOM4STS tool in a realistic environment. Hence, the independent variables of this experiment were the modelling approaches that we wanted to compare: Modelling on Paper (MoP) and Modelling on Software (MoS). The former allows subjects to use pen and paper to create the requirements models while the latter allows subjects to use the AOM4STS tool for the same purpose.

The evaluation of a modelling approach can be characterized by two dependent variables: (1) the effort during modelling; and (2) the effectiveness of the modelling process.

With the objective of evaluating possible benefits of using the AOM4STS tool for modelling requirements for sociotechnical systems, in comparison with the use of pen and paper, we defined the following two research questions.

RQ1: To what extent is the modelling effort with the AOM4STS software tool different from modelling on paper?

RQ2: To what extent is the modelling effectiveness with the AOM4STS software tool different from modelling on paper?

2) Experimental Design

The experiment was run in a lecture room, as a blocked subject-object experiment [11] whereby a set of objects were assigned to a set of subjects in a random way. The two objects – requirements specifications of two sociotechnical systems – were assigned to each participant (subject).

3) Subjects

The participants of the experiment were 8 post-graduate students (MSc and PhD) taking the requirements engineering course. Among various sub-topics of this course are goal-oriented approaches and agent-oriented methodologies for requirements engineering. These participants of this study were not students taught by experimenters. Furthermore, they were using only paper-based requirements modelling in their requirements engineering course.

4) Objects

The objects of this study were two small sociotechnical systems – a Meeting Scheduler System (MES)² [22] and a Personalized Emergency System (PES)³ [23]. The former is a computer-based service that supports setting up meetings while the latter is a system that supports a person, generally an older person, to remain living at home longer.

5) Data collection

The experiment was conducted for 3 hours in two consecutive days – 90 minutes on day 1 and another 90 minutes on day 2. In this experimental design, each subject performed the experiment tasks with both objects and with both treatments. This means that on day 1, half of the subjects were given the PES object and the remaining half of the subjects were given the MSS object. Moreover, half of those who received the PES object conducted the modelling on paper and the other half with the AOM4STS tool. Similarly, half of those subjects who received the MSS object conducted the modelling on paper and the other half with the software tool. On day 2, each subject

² <https://goo.gl/AVrDzx>

³ <https://goo.gl/wAMxnE>

changed the object and treatment. This experimental design mitigates learning effects between the two objects and between the two treatments [11].

6) Data Analysis

For the comparison of the MoP and MoS treatments, we (1) collected data through questionnaires, (2) applied measures of central tendency – mean, median and mode [24] – to compare the impact of the collected data on the treatments, and (3) grouped the results based on the collected data to answer the two research questions RQ1 and RQ2.

To evaluate the two treatments, the questions from q4 to q10 from Table II were repeated for each treatment used during the experiment. Similarly, the answers to the questions from postq2 to postq5 from Table II, which mostly focused on the evaluation of the AOM4STS tool, were compared with respect to the value 3, which is the neutral answer according to the Likert scale used in the study. The answers to the questions from q1 to q3 from Table III, which captured the relative time spent on reading the tutorial and understanding the case study, and the actual time spent on modelling requirements for the system in %, were multiplied with the overall time used by the subject in the corresponding experiment to obtain time measurements that could be compared between the two treatments.

B. Execution

As for the task of the experiment, the subjects had to create two requirements models – goal model and domain model – for two case studies, one of which had to be modelled on paper and another one with the AOM4STS tool. Each case study had to be modelled with as much of detail as possible, with the given treatment, and by following the step-by-step description of the requirements for each case study. Before the beginning of the modelling task, each subject had to fill in a pre-questionnaire. After completion of the modelling task for each case study, a subject had to fill in a questionnaire for the corresponding case study and treatment used. Finally, each subject had to fill in a post-questionnaire. A collection of questions for each type of questionnaire is provided in Table II.

The pre-questionnaire aimed to assess the knowledge of the subjects with respect to computing studies, requirements engineering, and agent-oriented requirements modelling.

The questions from preq4 to preq6 as presented in Table II aimed to assess the knowledge of the AOM methodology acquired after completion of the tutorial, and therefore measured the adequateness of the tutorial given before the modelling experiment. The questionnaire associated with each treatment the questions from q4 to q10 as is described in Table II, which evaluated the adequateness of the objects of the case study and collected perceptions by the subjects based on the specific treatment applied. Finally, the questions in the post-questionnaire from postq2 to postq5, as listed in the bottom of Table II, collected data about the effectiveness of the requirements modelling with the AOM4STS tool as compared with modelling the requirements on paper.

TABLE II. A SET OF THE QUESTIONS IN THE QUESTIONNAIRES

| 1 – Strongly disagree 2 – Disagree 3 – Neutral 4 – Agree 5 – Strongly agree | |
|---|--|
| preq4 | Basic principles of the AOM modelling methodology are clear |
| preq5 | The visual notations of the AOM methodology are clear |
| preq6 | Basic knowledge of using the AOM4STS tool has been acquired |
| q4 | The description of the case study was clear to me |
| q5 | I had no difficulties in modelling the goal model |
| q6 | I had no difficulties in modelling the domain model |
| q7 | I had enough time for accomplishing the modelling task |
| q8 | Goal decomposition was very useful in this task |
| q9 | The concepts of the AOM methodology were detailed enough to model the requirements of the system |
| q10 | The effort of modelling seems too high for an efficient use of the methodology in practice |
| postq2 | The propagation of roles created in the goal model into the domain model is helpful for the modeller |
| postq3 | The propagation of changes made to the roles in the goal model into the domain model helps to reduce the modelling effort |
| postq4 | The modelling software supports creation of syntactically correct models by preventing and reporting syntactically wrong connections |
| postq5 | The use of coloured connections in the creation of the models by the modelling software helps to improve the readability of the resulting models |

Furthermore, the overall time needed for completing the experimental task was recorded before filling in the corresponding questionnaire. The participants were also asked to keep track of the time in fractions (in %) spent on various activities. An indicative period of 1 hour was given to the subjects as a suggestion for performing the experimental modelling task on each day of the experiment, but subjects were free to take the time they required for completing the experimental task. The questions that were asked on the time spent on activities in each experiment are presented in Table III.

TABLE III. QUESTIONS ON THE TIME SPENT ON THE ACTIVITIES IN EACH EXPERIMENT

| Question label | Question description |
|----------------|--|
| duration | Time used for the task, in minutes |
| q1 | Reading the description of the AOM methodology in % |
| q2 | Reading and understanding the description of the case study in % |
| q3 | Modelling the case study in % |

VI. RESULTS ANALYSIS AND INTERPRETATION

This section presents the results of the empirical study by considering the measures of central tendency of the data collected from the subjects through questionnaires and provides the interpretation of the results that yields answers to the research questions RQ1 and RQ2.

A. Adequateness of the experimental settings

Before analysing the main factors of the empirical study, we analysed if the settings for the experiment were adequate. The pre-questionnaire asked about the subject's experience in the fields of computing, requirements analysis, and agent-

oriented requirements modelling to measure the influence of these co-factors on the study.

1) Adequateness of the subjects

Although all the subjects were postgraduate students in the requirements engineering course, they had different experiences in computing. The results of the collected data⁴ show that half of the subjects had little knowledge in computing, whereby 38% had experience in computing obtained through research projects, and 12% had experience in computing obtained through working as a computing professional in IT companies.

All the subjects were registered for the requirements engineering course to either acquire new knowledge or improve their knowledge in requirements engineering. The results of the collected data show that 75% of the subjects had little experience in requirements analysis while the remaining 25% had research experience in requirements analysis.

Before the subjects started to participate in the experiment, we gave a short tutorial about agent-oriented requirements modelling. After the tutorial, we did a short demonstration on agent-oriented goal modelling and domain knowledge modelling by using the AOM4STS tool. To be able to measure the effectiveness of the tutorial and demonstration for the subjects, we measured the prior experience of the subjects in agent-oriented requirements modelling. The results of the collected data shows that 75% of the subjects did not have any experience in agent-oriented requirements modelling while 25% had little experience in agent-oriented requirements modelling.

The results in Table IV provide a summary of the adequateness of the settings for the experiment after completing the tutorial and demonstration of the AOM4STS tool.

TABLE IV. RESULTS OF THE MEDIANS OF THE ADEQUATENESS OF THE SUBJECTS

| Ref. | Question | Median |
|-------|---|--------|
| preq4 | Basic principles of the AOM modelling methodology are clear | 5 |
| preq5 | The visual notations of the AOM methodology are clear | 5 |
| preq6 | Basic knowledge of using the AOM4STS tool has been acquired | 5 |

The results presented in Table IV show that the subjects acquired adequate understanding of the AOM methodology and the AOM4STS modelling tool for participating in the modelling experiment to give undistorted feedback.

2) Adequateness of the case studies

Adequateness of the objects used in the experiment was evaluated by the questions q4 and q7, which were answered by the subjects after completion of the modelling task independently of the treatment used. The questions and results are presented in Table V.

On the one hand, for the question q4, the median value for the PES case study was 5, while the median value for the MES case study was 4. This means that the subjects considered that the descriptions of both case studies were nearly equally clear.

TABLE V. RESULTS OF THE MEDIANS FOR THE ADEQUATENESS OF THE OBJECTS

| Ref. | Question | Median (PES) | Median (MES) |
|------|--|--------------|--------------|
| q4 | The description of the case study was clear to me | 5 | 4 |
| q7 | I had enough time for accomplishing the modelling task | 4 | 4 |

Although the description of the PES case study was considered clearer compared to the description of the MES case study, we believe the objects were adequate to provide unbiased results because both results were above the median value 3. On the other hand, for the question q7, the median value for both the PES and MES case studies was 4. This means that the subjects agreed that they had enough time for accomplishing the modelling task. The value 4 for each case study reduces the possibility of having biased results with respect to the time allocated for the experiment. Moreover, since the result for the question q4 was above the neutral value, which is 3 in the 1...5 Likert scale, the subjects did not experience time pressure when performing the modelling tasks. Consequently, the time allocated for the experiment did not have any influence on the results of the experiment. Therefore, we can claim that in overall the settings for the experiment were adequate.

B. Main factor: results and interpretation

In this section, we provide the results for the main factor of the experiment – the approach used – and compare the two treatments.

1) Evaluation of modelling effort

In this sub-section, we provide an answer to the research question RQ1 addressing the modelling effort, which was stated in Section V.A.1, based on the mean values represented in Fig. 2 and variance values shown in Fig. 3.

The question q0 records the overall time used by a subject for modelling a case study. The mean for modelling on paper (30) was nearly the same as the mean for modelling with the tool (29.6). However, the variance for modelling on paper (5.8) is noticeably higher than that for modelling with the tool (3.5).

The question q10 records modelling effort perceived by the subjects. The mean value of the modelling effort perceived by the subjects for modelling on paper and modelling with the tool were both close to 3 and their variances close to 0.7. Therefore, the subjects perceived the modelling effort on paper to be the same as the modelling effort with the tool.

The question q1 records the time used by the subjects for reading and understanding the modelling methodology. The mean time used by the subjects for reading and understanding the modelling methodology was slightly higher for subjects who conducted modelling with the tool (5.5) compared to those who modelled on paper (4.1). The variance of the time used by the subjects for reading and understanding the methodology was noticeably higher for subjects who conducted modelling with the tool (2.6) compared to those who modelled on paper (1.3).

⁴ <https://goo.gl/qatU6s>

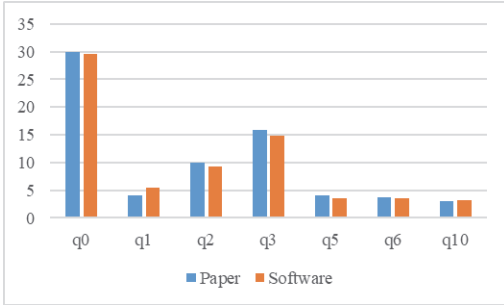


Fig. 2. Mean values for comparing modelling effort of the two treatments

Moreover, the question q2 records the time used by the subjects for reading and understanding the description of the case study. The mean time used by the subjects for reading and understanding the case study was slightly lower for subjects who conducted modelling with the tool (9.3) compared to those who modelled on paper (10.1). The variance of the time used by the subjects for reading and understanding the case study was noticeably lower for subjects who conducted modelling with the tool (3.1) compared to those who modelled on paper. Furthermore, the question q3 records the time consumed by the subjects for conducting the actual modelling using the two treatments. The mean time used by the subjects for conducting the actual modelling with the tool (14.8) was slightly lower than that for those who conducted the actual modelling on paper (15.8). The variance of the time used by the subjects for conducting the actual modelling with the tool was noticeably lower (3.6) than that for conducting the actual modelling on paper (6.7).

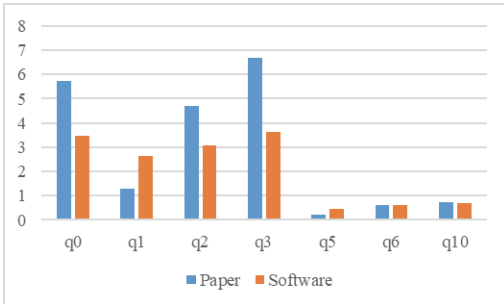


Fig. 3. Variance values for comparing modelling effort of the two treatments

The question q5 records the difficulty perceived by the subjects during goal modelling while q6 records the difficulty perceived by the subjects during domain knowledge modelling. For q5, the mean value of the difficulty perceived during goal modelling when modelling with the tool (3.6) is slightly lower than that for modelling on paper (4.1). However, the variance of the difficulty perceived by subjects during goal modelling

when modelling with the tool (0.5) is noticeably higher than that for modelling on paper (0.2). For q6, the mean value of the difficulty perceived during domain modelling when modelling with the tool is slightly lower (3.5) than that for modelling on paper (3.8) but their variances are the same (0.6).

Considering all the collected data for q0 to q6 and q10, we must answer the research question RQ1 as follows: the modelling effort on paper is nearly the same as the modelling effort with the AOM4STS software tool. However, the variance values for comparing the modelling efforts of the two treatments are considerably different. In the reported study the higher variance values for the modelling effort on paper dominate as compared with the modelling effort with the tool. An explanation for this is that the tool imposes more constraints on the requirements modelling activities. The higher variances of the time for the questions q1 and q5 when using the tool require further research.

2) Evaluation of modelling effectiveness

In this sub-section, we provide an answer to the research question RQ1 addressing the modelling effectiveness, which was stated in Section V.A.1, based on the mean values represented in Fig. 4 (a) and variance values shown in Fig. 4 (b).

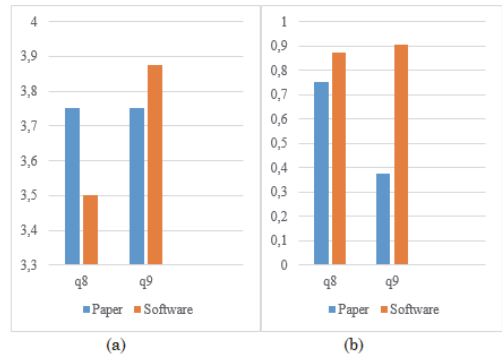


Fig. 4. (a) Mean and (b) Variance for comparing modelling effectiveness of the two treatments

The question q8 records the usefulness of goal decomposition during goal modelling. The mean value for the usefulness of goal decomposition on paper (3.75) as perceived by the subjects was slightly higher than that of modelling with the tool (3.5) with the variance of 0.75 when modelling on paper and 0.88 when modelling with the tool.

The question q9 records the utility of the concepts of goal modelling and domain knowledge modelling perceived by the subjects for requirements modelling. The mean value of the subjects who conducted modelling with the tool was 3.88 while the same value for those who conducted modelling on paper was 3.75. The variance of the subjects who conducted modelling with the tool was 0.91 while the same value for those who conducted modelling on paper was 0.38.

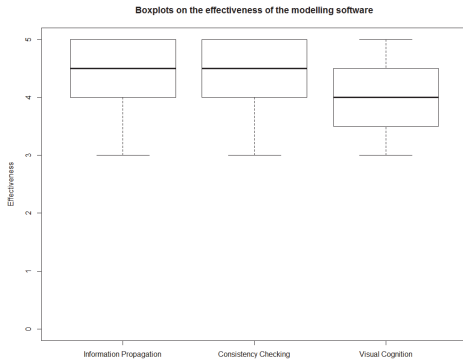


Fig. 5. Boxplot on the effectiveness of the modelling tool with respect to information propagation, consistency checking, and visual cognition

Furthermore, the subjects agreed on the effectiveness of the key features of the AOM4STS modelling tool that are not present in paper-based modelling with the median values 4.5 for information propagation, 4.5 – for consistency checking, and 4 – for visual cognition. The distribution of these results is depicted by the boxplot presented in Fig. 5. Moreover, the results in the boxplot clearly show that none of the subjects disagrees or strongly disagrees with the effectiveness of the AOM4STS modelling tool with respect to information propagation, consistency checking, and visual cognition.

Considering all the collected data from q8 and q9 and the postquestionnaire (information propagation, consistency checking, and visual cognition), we must answer the research question RQ2 as follows: the effectiveness of modelling requirements with the modelling tool was higher than the effectiveness of modelling requirements on paper except for goal decomposition which was slightly more effective when modelled on paper compared to modelling with the tool.

C. Validity Evaluation

In this experiment, there is one major threat to the internal validity [11]. This empirical study was not conducted by professionals in the industrial environment. According to [25], empirical evaluation by professionals in the real environment embraces all of the complexities of human practice in real organisations, gives stronger internal validity, and assures a more rigorous assessment of the effectiveness of the artefact. However, the research results by [26], [27] show that professionals and students perform similarly in empirical evaluations of software engineering artefacts, especially when they apply a new approach for the first time.

Concerning the external validity [11], it is highly probable that similar results will be obtained when running this experiment in a similar way with other subjects because the subjects of this experiment decided to register for the course of agent-oriented modelling of sociotechnical systems based on their interest in advanced software engineering and convenience. However, all the resources used in this experiment are publicly

available in the experiment package⁵ to encourage repetition of the study.

The threat to conclusion validity [11] relates to the sample size during the empirical study which involved modelling of 8 real world sociotechnical systems. According to [28], a large sample size helps to statistically observe nearly any legitimate differences between experimental conditions. Moreover, a large sample size improves the quality of research contributions. However, the systematic review of 1,700 software engineering papers published from 2001 to 2011 [29] on controlled experiments of software engineering tools with human participants reports on a large range of participants from 1 to 2,600 (the latter was a field deployment) with a median of 10 participants. Therefore, the sample size during this empirical study is very close to the median sample size of similar empirical studies.

The construct validity [11] includes two major threats. The first threat to the construct validity is that the used metrics may not be appropriate ones for evaluating the effectiveness of the modelling guidelines. For example, is “the comparison between the number of entities produced by the AOM approach and the number of the resulting entities of CPN in CPN Tools” an appropriate metric for evaluating the effectiveness of the modelling guidelines? The second threat to the construct validity is that the experiment was conducted as a part of the course, where the students were graded. This implies that the students may bias their data, as they believe that it will give them better grades. However, in the beginning of the course it was emphasised that the grade did not depend on the actual data. The grade was instead based on the completeness of the requirements, proper delivery, and the understanding of the topics expressed in the reports that were handed in by students at the end of the course.

VII. CONCLUSIONS

We conducted an empirical study with the objective of evaluating the effort and effectiveness of modelling requirements by goal models and domain models using pen and paper in comparison with the use of the modelling tool developed for engineering requirements for sociotechnical systems. The study involved experimental tasks of modelling requirements for sociotechnical systems and was conducted by postgraduate students registered for the requirements engineering course at the University of Tartu. The assessment results of experimental settings show that a short tutorial about goal modelling and domain knowledge modelling and a brief demonstration of the newly developed modelling tool were adequate. That is, these measures provided subjects with sufficient knowledge to perform adequately the modelling tasks.

The answer to the first research question lets us to conclude that the modelling effort on paper is nearly the same as the modelling effort with the AOM4STS software tool. However, the higher variance values for the modelling effort on paper dominate as compared with the modelling effort with the tool. An explanation for this is that the tool imposes more constraints on the requirements modelling activities. As the answer

⁵ <https://goo.gl/eVMe2B>

to the second research question, we can also conclude based on the results of the empirical study that the effectiveness of modelling requirements with the modelling tool was higher than the effectiveness of modelling requirements on paper by considering information propagation, consistency checking, and visual cognition. However, goal decomposition activity was slightly more effective when modelled on paper compared to modelling with the tool.

The answers to the research questions and particularly the answer to the second research question allow us to conclude that the support by modelling tools is essential for engineering requirements for sociotechnical systems because for such systems requirements should be modelled at different abstraction levels and from different perspectives that should be consistent with each other. In the future work, we will conduct another empirical study with a large sample size of professionals that aims to increase the confidence in the results obtained from this empirical study.

Acknowledgment. This research is partly supported by the Estonian Research Council (grant IUT20-55).

REFERENCES

- [1] R. Lock and I. Sommerville, "Modelling and analysis of socio-technical system of systems," in *Engineering of Complex Computer Systems (ICECCS), 2010 15th IEEE International Conference on*, 2010, pp. 224–232.
- [2] J. Marshall, "Agent-based modelling of emotional goals in digital media design projects," in *Innovative Methods, User-Friendly Tools, Coding, and Design Approaches in People-Oriented Programming*, IGI Global, 2018, pp. 262–284.
- [3] B. Whitworth, "A brief introduction to sociotechnical systems," in *Encyclopedia of Information Science and Technology, Second Edition*, IGI Global, 2009, pp. 394–400.
- [4] V. P. da Conceição, J. Dahlman, and A. Navarro, "What is maritime navigation? Unfolding the complexity of a Sociotechnical System," in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 2017, vol. 61, pp. 267–271.
- [5] M. C. Davis, R. Challenger, D. N. Jayewardene, and C. W. Clegg, "Advancing socio-technical systems thinking: A call for bravery," *Appl. Ergon.*, vol. 45, no. 2, pp. 171–180, 2014.
- [6] G. Baxter and I. Sommerville, "Socio-technical systems: From design methods to systems engineering," *Interact. Comput.*, vol. 23, no. 1, pp. 4–17, 2011.
- [7] L. S. Sterling and K. Taveter, *The art of agent-oriented modeling*. MIT Press, 2009.
- [8] T. Miller, B. Lu, L. Sterling, G. Beydoun, and K. Taveter, "Requirements elicitation and specification using the agent paradigm: the case study of an aircraft turnaround simulator," *IEEE Trans. Softw. Eng.*, vol. 40, no. 10, pp. 1007–1024, 2014.
- [9] A. Norta, M. Mahunah, T. Tenso, K. Taveter, and N. C. Narendra, "An agent-oriented method for designing large socio-technical service-ecosystems," in *Services (SERVICES), 2014 IEEE World Congress on*, 2014, pp. 242–249.
- [10] K. Taveter and A. Norta, "Agile Software Engineering Methodology for Information Systems' Integration Projects," in *International Conference on Future Data and Security Engineering*, 2017, pp. 215–230.
- [11] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in software engineering*. Springer Science & Business Media, 2012.
- [12] M. Brambilla, J. Cabot, and M. Wimmer, "Model-driven software engineering in practice," *Synth. Lect. Softw. Eng.*, vol. 1, no. 1, pp. 1–182, 2012.
- [13] J. Whittle, J. Hutchinson, and M. Rouncefield, "The state of practice in model-driven engineering," *IEEE Softw.*, vol. 31, no. 3, pp. 79–85, 2014.
- [14] P. Mohagheghi, W. Gilani, A. Stefanescu, and M. A. Fernandez, "An empirical study of the state of the practice and acceptance of model-driven engineering in four industrial cases," *Empir. Softw. Eng.*, vol. 18, no. 1, pp. 89–116, 2013.
- [15] F. D. Giraldo, S. España, and Ó. Pastor, "Evidences of the mismatch between industry and academy on modelling language quality evaluation," *ArXiv Prepr. ArXiv160602025*, 2016.
- [16] J. Whittle, J. Hutchinson, M. Rouncefield, H. akan Burden, and R. Heldal, "A taxonomy of tool-related issues affecting the adoption of model-driven engineering," *Softw. Syst. Model.*, vol. 16, no. 2, pp. 313–331, 2017.
- [17] G. Mussbacher *et al.*, "The relevance of model-driven engineering thirty years from now," in *International Conference on Model Driven Engineering Languages and Systems*, 2014, pp. 183–200.
- [18] T. Gorschek, E. Tempero, and L. Angelis, "On the use of software design models in software development practice: An empirical investigation," *J. Syst. Softw.*, vol. 95, pp. 176–193, 2014.
- [19] A. Sapožnikov, "Design and Development of a Graphical Tool for Agent-Oriented Modelling. MSc thesis (in Estonian)." Tallinn University of Technology, Faculty of Information Technology, Department of Informatics, 2016.
- [20] L. Sterling and K. Taveter, "Building agent-based appliances with complementary methodologies," in *Proceedings of the 2006 conference on Knowledge-Based Software Engineering: Proceedings of the Seventh Joint Conference on Knowledge-Based Software Engineering*, 2006, pp. 223–232.
- [21] T. Tenso, A. H. Norta, H. Rootsi, K. Taveter, and I. Vorontsova, "Enhancing requirements engineering in agile methodologies by agent-oriented goal models: Two empirical case studies," in *2017 IEEE 25th International Requirements Engineering Conference Workshops (REW)*, 2017, pp. 268–275.
- [22] E. S. Yu, "Towards modelling and reasoning support for early-phase requirements engineering," in *Requirements Engineering, 1997., Proceedings of the Third IEEE International Symposium on*, 1997, pp. 226–235.
- [23] T. Miller, S. Pedell, A. A. Lopez-Lorca, A. Mendoza, L. Sterling, and A. Keirnan, "Emotion-led modelling for people-oriented requirements engineering: The case study of emergency systems," *J. Syst. Softw.*, vol. 105, pp. 54–71, 2015.
- [24] E. R. Babbie, *The basics of social research*. Cengage Learning, 2013.
- [25] J. Venable, J. Pries-Heje, and R. Baskerville, "FEDS: a framework for evaluation in design science research," *Eur. J. Inf. Syst.*, vol. 25, no. 1, pp. 77–89, 2016.
- [26] D. Falessi *et al.*, "Empirical software engineering experts on the use of students and professionals in experiments," *Empir. Softw. Eng.*, pp. 1–38, 2017.
- [27] I. Salman, A. T. Misirli, and N. Juristo, "Are students representatives of professionals in software engineering experiments?," in *Proceedings of the 37th International Conference on Software Engineering-Volume 1*, 2015, pp. 666–676.
- [28] M. Kaptein and J. Robertson, "Rethinking statistical analysis methods for CHI," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2012, pp. 1105–1114.
- [29] A. J. Ko, T. D. Latoza, and M. M. Burnett, "A practical guide to controlled experiments of software engineering tools with human participants," *Empir. Softw. Eng.*, vol. 20, no. 1, pp. 110–141, 2015.

Appendix D

Msury Mahunnah, Kuldar Taveter, Cheah Wai Shiang, and Sim Wai Yee, “An Empirical Evaluation of Guidelines for Prototyping Sociotechnical Systems in JADE Framework,” in 3rd IEEE International Symposium on Agents, Multi-Agent Systems and Robotics. IEEE, 2018.

Empirical Evaluation of Guidelines for Prototyping Sociotechnical Systems in JADE Framework

Msury Mahunnah¹, Kuldar Taveter¹, Cheah Wai Shiang² and Sim Yee Wai³

¹Department of Software Science, Tallinn University of Technology, Tallinn, Estonia.

²Faculty of Computer Science & IT Universiti Malaysia Sarawak, Sarawak, Malaysia

³School of Computing, University College of Technology Sarawak, Sarawak, Malaysia

msury.mahunnah@ttu.ee, kuldar.taveter@ttu.ee, c.waishiang@gmail.com, yee.wai.sim@ucts.edu.my

Abstract—One of the major problems of conceptual modelling is the lack of sufficient empirical evidence that evaluates the effectiveness of conceptual models in the development of sociotechnical systems. In this paper, we report the results of empirical study that investigates the effectiveness of adapting guidelines for prototyping conceptual models of sociotechnical systems in Java Agent Development (JADE) framework. These guidelines are divided into the JADE knowledge, -interaction and -behaviour prototyping guidelines. The empirical study involves developing 16 different sociotechnical prototypes with the JADE framework. Among them, 8 sociotechnical systems were developed using the JADE prototyping guidelines together with JADE website resources, while the remaining 8 sociotechnical systems were developed using JADE website resources alone. The evaluation results of the JADE prototyping guidelines ascertain that they are more effective for the development of agent knowledge of sociotechnical systems in JADE framework than the current practice of using JADE website resources alone. Moreover, the evaluation results of the JADE prototyping guidelines find out that conceptual objects of sociotechnical systems are necessary building blocks in developing JADE ontology. In the future work, we will conduct another empirical study with a large sample size of professionals that aims to increase the confidence in the results obtained from this empirical study.

Keywords—*empirical evaluation, agent-oriented modelling, sociotechnical systems*

I. INTRODUCTION

A sociotechnical system is a software intensive system that has defined operational processes followed by human operators that operates within an organization [1]. Recent developments in the software engineering paradigm have led to a renewed interest in system development methodologies that cover higher contexts and leads to the design of sociotechnical systems [2]. The adaption of agent -oriented software engineering (AOSE) methodologies shows promising results in conceptual requirements elicitation and design modelling of sociotechnical systems [3]. However, for sociotechnical systems to evolve and extend its reach, [4] suggests the need to extend conceptualization of what constitutes a system; apply our thinking to a much wider range of complex problems and global challenges; and engage in more predictive work. To extend conceptualization of what constitutes a system, [5]–[7] consider human, software and devices as necessary building blocks of the sociotechnical systems. Specifically, [6] focuses on improving workplace safety from a sociotechnical perspective, [7] focuses on security requirements engineering of sociotechnical systems and [5] suggests a holistic modelling approach that focuses on requirements elicitation and design of sociotechnical systems using agent-oriented modelling.

Despite of the obvious benefits, agent technology has not been widely adopted by the software community [8]. The reasons for the setbacks are the diversity of agent -oriented software engineering methodologies and the lack of maturity in some of the methodologies [9]. To improve the maturity of agent technology, [10] extends agent modelling approach [5] by proposing guidelines for the development of sociotechnical systems in Java Agent Development (JADE) framework [11].

The available research literature [12] shows that the JADE framework [11] is the most popular agent platform. JADE framework has been purely designed in Java and supports different kinds of web-based application systems and is compliant with the specifications by the Foundation for Intelligent Physical Agents (FIPA) [13]. However, the gap exists on the empirical evaluation of JADE development guidelines [10] in order to assess their advantages and disadvantages, to ensure their applicability in different contexts, their ease of use, and other issues such as required skills. Therefore, this paper reports the empirical study that aims to investigate the effectiveness of JADE guidelines [10] for the development of sociotechnical systems in JADE framework [11].

The structure of this chapter is as follows. Section II presents related work. Section III provides the research methods that describes the scope of the empirical study, the plan for conducting this experiment and the execution of the empirical study. Section IV provides the analysis of collected, describes the experiment results and discuss the experiment results. Section V concludes the paper and describes the future work.

II. RELATED WORK

The presence of software tools promotes the usage of agent-oriented software engineering for complex system development [14]. According to [15], only a few tools support the full development process of agent-oriented systems. The following paragraph provides an overview of some of these tools.

In the software tool described by [15], designers define the phases and activities of the development of agent-oriented systems and identify the relationships between modelling activities. Another relevant tool – AgentTool Process Editor (APE) – is a software tool implemented as an Eclipse plug-in that supports the design, validation and management of agent-oriented systems according to the O-MaSE methodology [16]. Furthermore, Freitas et al. [17] introduce a tool that enables the transformation of conceptual models into the implementations of agent-oriented system. Also, Yu et al. [18] describe an Integrated Development Environment (IDE) for modelling a system behaviour based

on the Goal Net model. Finally, Manzoor and Zafar [19] describe a Multi-Agent Modelling Toolkit (MAMT) that uses Agent Unified Modelling Language (AUML) models to support designers during rapid development of complex systems. Despite the existence of various conceptual modelling tools for conceptual requirements elicitation, design and development of agent systems, the existing literature indicates that JADE is the most popular agent platform, purely designed in Java and supports different kinds of systems operating in the web [12]. Moreover, [20] suggests the need for further research on the software tools to support the synergy between the agent technology and sociotechnical systems. Therefore, this paper focuses on the development guidelines [10] that employ JADE framework¹ for prototyping conceptual design models of sociotechnical produced by agent-oriented modelling approach [5].

However, the agent-oriented modelling approach [5] stems from the paradigm of Model-Driven Engineering (MDE) [21] that focuses on the systematic use of models as primary engineering artefacts throughout the system engineering lifecycle. Among the key benefits of the MDE paradigm are effective expression of domain concepts [22], decreasing system development time and improving system quality [23]. Despite the benefits of the MDE paradigm, various studies show that a domain-specific MDE language is not enough for industry-wide adoption and a tool supporting such language increases the complexity of the development process instead of diminishing it [24]. Elsewhere, Whittle et al. [25] interviewed 39 practitioners on tool-related issues affecting the adoption of MDE. The results of this study indicate that the complexity of the modelling tools is among the major issues hindering practical application of MDE. Another study involved 15 MDE experts in a thought experiment to identify the biggest problems with current MDE technologies [26]. The results of this study found that steep learning curves and arduous user interfaces are among significant usability challenges to industry-wide adoption of MDE tools.

The findings from this review of related work point to the need of conducting research studies on MDE methods to empirically compare claimed benefits of a modelling method. Furthermore, the summary of papers presented in the workshop on the experiences and empirical studies in software modelling [27] suggests the need to conduct more empirical studies on the evaluation of modelling methods, languages and tools in order to assess their advantages and disadvantages, to ensure their applicability in different contexts, their ease of use, and other issues such as required skills and costs. The papers overviewed by [27] include a study that assessed the frequency of empirical evaluation in software modelling research [28] by reviewing 266 papers. The study found that 195 (73%) of the publications did not report about any empirical evaluation. This finding clearly indicates the need for more empirical studies in software modelling research.

The following Section III and Section IV adapt the principles of experimentation in software engineering [29] to report the empirical study that investigates the effectiveness of the guidelines proposed in [10] for the development of sociotechnical systems the JADE Framework.

III. THE RESEARCH METHODS

This Section describes the scope, plan and execution of the empirical study.

A. Experiment Scoping

This Section describes the scope of the empirical study.

1) Goal definition

The objective of this empirical study is to determine the effectiveness of the JADE guidelines proposed in [10] for the development of sociotechnical systems in the JADE Framework.

For determining the effectiveness of the JADE guidelines proposed in [10], it is important to experiment and compare the results of developing sociotechnical design models on the JADE framework by using the JADE guidelines against the results of developing sociotechnical design models on the JADE framework without using the JADE guidelines.

2) Object of study

The object of the study are the JADE guidelines put forward in [10] that assist the development on the JADE framework of interaction, knowledge and behaviour design models of sociotechnical systems produced by the agent-oriented modelling approach [5].

3) Perspective

The perspective taken is that by the researchers and designers willing to make use of the JADE guidelines for developing sociotechnical systems from knowledge, interaction and behaviour design models produced by the agent-oriented modelling approach [5]. This also includes people who would like to adopt the JADE guidelines in industry or conduct further research on the JADE guidelines.

4) Quality focus

The first effect studied in this experiment is the relation between the numbers of design features in JADE prototypes produced by using the JADE guidelines against the number of design features in JADE prototypes produced without using the JADE guidelines. These design features include the numbers of types of agents, interactions, conceptual objects, and behaviours.

The second effect studied in this experiment is the study of the development of JADE ontology in prototypes produced without using the JADE guidelines against the development of the JADE ontology in prototypes produced by using the JADE guidelines.

5) Context

The experiment was run in the context of agent-oriented development for sociotechnical systems. The experiment was conducted within the course of agent oriented modelling and multi-agent systems given at the Department of Software Science of Tallinn University of Technology in Estonia.

6) Summary of Scoping

Analyse the outcome of developing sociotechnical systems on the JADE Framework for evaluation with respect of using the JADE guidelines from the perspective of researchers and system designers in the context of agent-oriented development of sociotechnical systems.

¹ <http://jade.tilab.com/>

B. Experiment Planning

This Section describes the plan for conducting this experiment.

1) Context Selection

The context of the experiment is the course on agent-oriented modelling of sociotechnical systems given at the university. The participants in the experiment are MSc and PhD students. Moreover, this experiment is specific because it is focused on agent-oriented modelling for sociotechnical systems in an educational environment. Later, this Section discusses the threats to the validity of the experiment and elaborates the ability to generalise the research findings from this specific context. This experiment addresses a real problem – the effectiveness of the JADE guidelines for developing on the JADE framework sociotechnical design models produced by the agent-oriented modelling approach [5].

The usage of the course on agent-oriented modelling of sociotechnical systems as an experimental context provides other researchers with an opportunity to replicate the experiment. Furthermore, it means that there is no need to spend much effort in setting up the repeated experiment in terms of defining the experiment and creating the environment for running the experiment.

2) Research Question

An important aspect of any experiment is to know and clearly state the research question to be answered by the experiment. This experiment aims to provide an answer to the following research question:

What is the effectiveness of the JADE guidelines in the prototyping of sociotechnical design models using JADE framework?

Before giving the answer to the research question, it is necessary to compare the key development features of the sociotechnical systems. The agents in the sociotechnical system use ontology to facilitate sharing of conceptual objects (knowledge items) through interactions [3]. These agents have different behaviours, which can either be implemented as one-shot or cyclic behaviour in JADE framework [11]. The former executes once and dies while the latter executes periodically.

Therefore, for guiding data collection and analysis, the research question is elaborated into more detailed research sub-questions and themes as is shown in TABLE I.

TABLE I. RESEARCH SUB-QUESTIONS AND THEMES.

| Research Sub-Question | Theme |
|--|-------------------------|
| Q1. How many agent types were developed? | multi-agent development |
| Q2. How many interaction types were developed? | interaction development |
| Q3. How many conceptual object types were developed? | knowledge development |
| Q4. How many behaviour types were developed? | behaviour development |
| Q5. Was ontology correctly developed and used by the agents? | knowledge development |

3) Selection of Subjects

The subjects were chosen based on convenience and interest, but not as a random sample in the sense that the subjects were students registered in 2012 and 2015 for the elective course on agent-oriented modelling of sociotechnical systems offered to MSc and PhD students of the School of Information Technology of Tallinn University of Technology. In this study, the students of 2012 are referred to as group 1 and the students of 2015 are referred to as group 2.

4) Variable Selection

The independent variables are the JADE guidelines presented in [10]. The dependent variables are the prototypes² developed without using the JADE guidelines by the subjects of group 1 and prototypes³ developed with the use of JADE guidelines by the subjects of group 2.

5) Experiment Design

The case studies were not assigned randomly to the subjects but the subjects chose their own case studies. The subjects of Group 1 were instructed to conduct requirements and design modelling for the selected case studies. Then, they were informed to use the resources from <http://jade.tilab.com/> and all available materials from the Internet to develop JADE prototypes based on the created design models. The subjects of Group 2 were instructed to use the same resources as Group 1 and the JADE guidelines presented in [10].

Furthermore, it would have been preferable to have a balanced dataset, but the experimental study was based on a course for which the subjects had registered. Therefore, it was impossible to influence the backgrounds of the subjects and this way balance the dataset.

6) Validity Evaluation

In this Section, we present the two major threats to the validity of this research work. First, this empirical study was not conducted by professionals in the industrial environment. Instead, it was conducted by Master's and PhD students. According to [30], empirical evaluation by professionals in the real environment embraces all of the complexities of human practice in real organisations, gives stronger internal validity, and assures a more rigorous assessment of the effectiveness of the artefact. This encourages the need for conducting an empirical evaluation by professionals in their professional work environment. However, the research results by [31], [32] on the use of students and professionals as subjects in software engineering experiments show that professionals and students perform similarly in empirical evaluations of software engineering artefacts, especially when they apply a new approach for the first time.

The second threat to the validity relates to the sample size during the empirical study which involved 8 participants, while each of them conducted modelling of two case studies. Consequently, there are 16 outcomes of the modelling experiment in total. According to [33], a large sample size helps to statistically observe nearly any legitimate differences between experimental conditions. Consequently, a large sample size improves the quality of research contributions. However, the systematic review of 1,700 software engineering papers published from 2001 to 2011

² <https://goo.gl/UA6bjj>

³ <https://goo.gl/2nRfPM>

[34] on controlled experiments of software engineering tools with human participants reports on a large range of participants from 1 to 2,600 (the latter was a field deployment) with a median of 10 participants. Therefore, the sample sizes during this empirical study is very close to the median sample size of similar empirical studies.

C. Experiment Operation

This Section describes the execution of the study.

1) Preparation

The subjects of this experiment were not aware of what aspects were going to be evaluated. They were only told that the researchers wanted to study the outcome of the course on agent-oriented modelling of sociotechnical systems with respect to the usage of the JADE Framework. They were, however, not aware of the actual research questions to be answered by the results of the experiment. The subjects, from their point of view, did not primarily participate in an experiment but were just taking a course. All students were guaranteed anonymity. The materials of the experiment were prepared in advance. The course itself was based on the textbook [5] about agent-oriented modelling of sociotechnical systems and the information about the course was provided on the course website⁴.

2) Execution

The experiment was executed over a period of 16 weeks, during which subjects participated in the JADE workshops, giving presentations and writing the reports. The data for the experiment was primarily collected from the source code of the developed JADE prototypes and from the reports submitted by the students at the end of the study.

As was stated before, the experiment was run within the course on agent-oriented modelling of sociotechnical systems and was conducted in the university environment. The design of the experiment was in line with the course objectives and therefore did not affect the study plan.

3) Data Validation

Data was collected from 13 projects in Group 1 and 13 projects in Group 2. Each group consisted of two to five students and focused on one case study. After the development experiment, the source code of the developed prototypes and reports were collected for analysis and interpretation. Some students found more convenient to do the projects individually. Their data was removed and regarded as invalid. Some student groups decided to use other agent development platforms, such as Jason⁵. Their data was also removed and regarded as invalid because that data was irrelevant for the given experiment. Some subjects did not follow the guidelines and created incomplete sets of design models for sociotechnical systems. Their data was also removed and regarded as invalid to avoid biased results.

After the removal of the invalid projects, a total of 16 projects remained and were considered in this empirical study: 8 projects in Group 1 and 8 projects in Group 2.

IV. RESULTS AND DISCUSSION

In this Section, we analyse the data collected from developing JADE prototypes by using mean, median, and mode. This analysis aims to compare what was achieved by

the subjects in Group 1 and Group 2 independently of their choices of projects.

According to the results represented in TABLE II., the median and mean for the numbers of types of agents, conceptual objects and behaviours implemented in the prototypes of Group 1 is nearly the same as those implemented in the prototypes of Group 2 with an exception of interactions. According to the collected data, project with the identifier 1 in Group 1 has a very large number of implemented interaction types (17) compared to the average number of implemented interaction types in Group 1 (8.875). Also, the same project identified by 1 has the highest number of agent types (12) compared to the average number of agent types implemented in Group 1 (3.875). Although these results do not say much about the usage of the JADE guidelines, they show that an increase in the number of agent types in the system results in the increase in the number of types of interaction between the agents.

TABLE II. MEDIAN AND MEAN COMPARISON OF PROTOTYPES IMPLEMENTED IN GROUP 1 AND GROUP 2.

| ENTITY TYPE | MEDIAN | | MEAN | |
|----------------------|---------|---------|---------|---------|
| | GROUP 1 | GROUP 2 | GROUP 1 | GROUP 2 |
| AGENT | 3 | 3,5 | 3,875 | 4 |
| INTERACTION | 9 | 6 | 8,875 | 5,75 |
| CONCEPTUAL OBJECT | 0,5 | 2 | 1,75 | 2,5 |
| BEHAVIOUR | 5 | 6 | 4,875 | 5,875 |

Fig. 1 presents the mode comparison results of the key components of sociotechnical systems in prototypes developed in Group 1 and Group 2. The results show that majority of the projects in both groups implemented simple attributes rather than types of conceptual objects. However, the results in Fig. 2 show that more projects in Group 2 (63%) implemented conceptual object types than in Group 1 (50%).

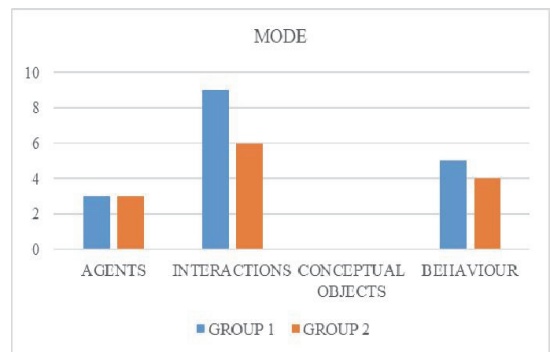


Fig. 1. Mode comparison of prototypes implemented in Group 1 and Group 2.

These results indicate that the JADE guidelines together with the JADE website resources are more effective for the development of conceptual objects of sociotechnical systems

⁴ http://maurus.ttu.ee/sts/?page_id=36

⁵ <http://jason.sourceforge.net/wp/>

⁶ <https://goo.gl/MMMCXM>

on the JADE Framework compared to using just the resources of the JADE website for the same purpose.

Moreover, the results in Fig. 2 show that 50% of the prototypes developed in Group 2 implemented the ontology and the implemented agents share knowledge among them through the ontology while in Group 1 none of the prototypes implemented the ontology. Again, these results indicate that the JADE guidelines together with the JADE website resources are more effective for the development of ontologies of sociotechnical systems on the JADE framework than just using the JADE website resources for the same purpose. In summary, the JADE guidelines together with the JADE website resources are more effective for the development on the JADE framework of agent knowledge for sociotechnical systems than using only the JADE website resources for the same purpose.

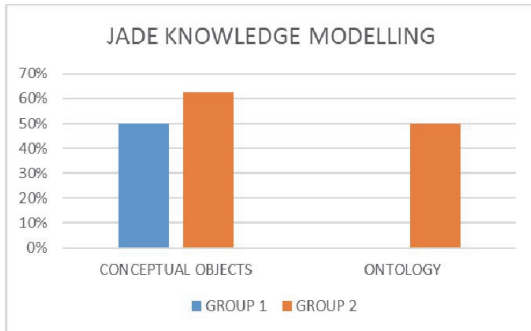


Fig. 2. Types of conceptual objects and ontology used in prototypes of Group 1 and Group 2.

TABLE III. refines the comparison between the number of conceptual object types and ontologies implemented in the prototypes developed by the subjects of Group 2. The results clearly show that all the prototypes that developed the ontology also managed to develop conceptual object types but not all the prototypes that implemented the conceptual object types managed to implement the ontology. These results indicate that conceptual object types may be necessary components for the development of an agent ontology on the JADE Framework.

TABLE III. CONCEPTUAL OBJECT TYPES AND ONTOLOGIES DEVELOPED IN GROUP 2.

| PROJECT ID | CONCEPTUAL OBJECTS | ONTOLOGY |
|------------|--------------------|----------|
| 1 | 4 | Yes |
| 2 | 2 | Yes |
| 3 | 0 | No |
| 4 | 0 | No |
| 5 | 6 | Yes |
| 6 | 0 | No |
| 7 | 2 | Yes |
| 8 | 6 | No |

V. CONCLUSIONS AND FUTURE WORK

This paper reports the results of the empirical study that investigates the effectiveness of the proposed JADE prototyping guidelines by developing 16 different real life sociotechnical prototypes with the JADE framework. Among them, 8 sociotechnical systems were developed using the JADE prototyping guidelines together with JADE website resources, while the remaining 8 sociotechnical systems were developed using JADE website resources alone.

On the one hand, the results of the empirical study ascertain that JADE prototyping guidelines together with JADE website resources are more effective for the development of agent knowledge for sociotechnical systems in JADE framework than using JADE website resources alone. Furthermore, the results find out that conceptual objects are necessary building blocks in developing JADE ontology for sociotechnical systems. On the other hand, the results of this empirical study do not show a substantial difference between the utility of JADE prototyping guidelines together with JADE website resources and the utility of JADE website resources alone in the development of agents, interactions and behaviours in JADE framework. Therefore, these results conclude that the proposed JADE guidelines provides effective development of agent knowledge and development of JADE ontology for the sociotechnical systems in JADE framework. In the future work, we will conduct another empirical study with a large sample size of professionals that aims to increase the confidence in the results obtained from this empirical study.

REFERENCES

- [1] R. Lock and I. Sommerville, "Modelling and analysis of socio-technical system of systems," in *Engineering of Complex Computer Systems (ICECCS), 2010 15th IEEE International Conference on*, 2010, pp. 224–232.
- [2] G. Baxter and I. Sommerville, "Socio-technical systems: From design methods to systems engineering," *Interact. Comput.*, vol. 23, no. 1, pp. 4–17, 2011.
- [3] K. H. van Dam, I. Nikolic, and Z. Lukszo, *Agent-based modelling of socio-technical systems*, vol. 9. Springer Science & Business Media, 2012.
- [4] M. C. Davis, R. Challenger, D. N. Jayewardene, and C. W. Clegg, "Advancing socio-technical systems thinking: A call for bravery," *Appl. Ergon.*, vol. 45, no. 2, pp. 171–180, 2014.
- [5] L. S. Sterling and K. Taveter, *The art of agent-oriented modeling*. MIT Press, 2009.
- [6] P. Carayon, P. Hancock, N. Leveson, I. Noy, L. Sznclwar, and G. Van Hootegem, "Advancing a sociotechnical systems approach to workplace safety—developing the conceptual framework," *Ergonomics*, vol. 58, no. 4, pp. 548–564, 2015.
- [7] F. Dalpiaz, E. Paja, and P. Giorgini, *Security requirements engineering: designing secure socio-technical systems*. MIT Press, 2016.
- [8] A. Oluyomi, S. Karunasekera, and L. Sterling, "Description templates for agent-oriented patterns," *J. Syst. Softw.*, vol. 81, no. 1, pp. 20–36, 2008.

- [9] A. Sturm and O. Shehory, "Agent-oriented software engineering: revisiting the state of the art," in *Agent-Oriented Software Engineering*, 2014, pp. 13–26.
- [10] C. W. Shiang, B. T. Onn, F. S. Tee, M. A. bin Khairuddin, and M. Mahunnah, "Developing Agent-Oriented Video Surveillance System through Agent-Oriented Methodology (AOM)," *CIT J. Comput. Inf. Technol.*, vol. 24, no. 4, pp. 349–368, 2016.
- [11] F. L. Bellifemine, G. Caire, and D. Greenwood, *Developing multi-agent systems with JADE*, vol. 7. John Wiley & Sons, 2007.
- [12] K. Kravari and N. Bassiliades, "A survey of agent platforms," *J. Artif. Soc. Soc. Simul.*, vol. 18, no. 1, p. 11, 2015.
- [13] F. Bellifemine, A. Poggi, and G. Rimassa, "Developing multi-agent systems with a FIPA-compliant agent framework," *Softw.-Pract. Exp.*, vol. 31, no. 2, pp. 103–128, 2001.
- [14] J. L. Posadas, J. L. Poza, J. E. Simó, G. Benet, and F. Blanes, "Agent-based distributed architecture for mobile robot control," *Eng. Appl. Artif. Intell.*, vol. 21, no. 6, pp. 805–823, 2008.
- [15] R. Fuentes-Fernández, I. García-Magariño, A. M. Gómez-Rodríguez, and J. C. González-Moreno, "A technique for defining agent-oriented engineering processes with tool support," *Eng. Appl. Artif. Intell.*, vol. 23, no. 3, pp. 432–444, 2010.
- [16] J. C. Garcia-Ojeda, S. A. DeLoach, and others, "agentTool process editor: supporting the design of tailored agent-based processes," in *Proceedings of the 2009 ACM symposium on Applied Computing*, 2009, pp. 707–714.
- [17] A. Freitas, L. Hilgert, S. Marczak, F. Meneguzzi, R. H. Bordini, and R. Vieira, "A multi-agent systems engineering tool based on ontologies," in *34th International Conference on Conceptual Modeling, Stockholm, Sweden, ser. Lecture Notes in Computer Science*. Springer, 2015.
- [18] H. Yu, Z. Shen, and C. Miao, "Intelligent software agent design tool using goal net methodology," in *Proceedings of the 2007 IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, 2007, pp. 43–46.
- [19] U. Manzoor and B. Zafar, "Multi-Agent Modeling Toolkit–MAMT," *Simul. Model. Pract. Theory*, vol. 49, pp. 215–227, 2014.
- [20] R. M. Crowder, M. A. Robinson, H. P. Hughes, and Y.-W. Sim, "The development of an agent-based modeling framework for simulating engineering team work," *IEEE Trans. Syst. Man Cybern.-Part Syst. Hum.*, vol. 42, no. 6, pp. 1425–1439, 2012.
- [21] M. Brambilla, J. Cabot, and M. Wimmer, "Model-driven software engineering in practice," *Synth. Lect. Softw. Eng.*, vol. 1, no. 1, pp. 1–182, 2012.
- [22] J. Whittle, J. Hutchinson, and M. Rouncefield, "The state of practice in model-driven engineering," *IEEE Softw.*, vol. 31, no. 3, pp. 79–85, 2014.
- [23] P. Mohagheghi, W. Gilani, A. Stefanescu, and M. A. Fernandez, "An empirical study of the state of the practice and acceptance of model-driven engineering in four industrial cases," *Empir. Softw. Eng.*, vol. 18, no. 1, pp. 89–116, 2013.
- [24] F. D. Giraldo, S. España, and Ó. Pastor, "Evidences of the mismatch between industry and academy on modelling language quality evaluation," *ArXiv Prepr. ArXiv160602025*, 2016.
- [25] J. Whittle, J. Hutchinson, M. Rouncefield, H. Burden, and R. Heldal, "A taxonomy of tool-related issues affecting the adoption of model-driven engineering," *Softw. Syst. Model.*, vol. 16, no. 2, pp. 313–331, 2017.
- [26] G. Mussbacher *et al.*, "The relevance of model-driven engineering thirty years from now," in *International Conference on Model Driven Engineering Languages and Systems*, 2014, pp. 183–200.
- [27] M. R. Chaudron, M. Genero, S. Abrahão, P. Mohagheghi, and L. Pareto, "Summary of the first international workshop on experiences and empirical studies in software modelling," in *International Conference on Model Driven Engineering Languages and Systems*, 2011, pp. 119–122.
- [28] J. C. Carver, E. Syriani, and J. Gray, "Assessing the Frequency of Empirical Evaluation in Software Modeling Research," *EESSMod*, vol. 785, 2011.
- [29] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in software engineering*. Springer Science & Business Media, 2012.
- [30] J. Venable, J. Pries-Heje, and R. Baskerville, "FEDS: a framework for evaluation in design science research," *Eur. J. Inf. Syst.*, vol. 25, no. 1, pp. 77–89, 2016.
- [31] D. Falessi *et al.*, "Empirical software engineering experts on the use of students and professionals in experiments," *Empir. Softw. Eng.*, pp. 1–38, 2017.
- [32] I. Salman, A. T. Misirli, and N. Juristo, "Are students representatives of professionals in software engineering experiments?," in *Proceedings of the 37th International Conference on Software Engineering-Volume 1*, 2015, pp. 666–676.
- [33] M. Kaptein and J. Robertson, "Rethinking statistical analysis methods for CHI," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2012, pp. 1105–1114.
- [34] A. J. Ko, T. D. Latoza, and M. M. Burnett, "A practical guide to controlled experiments of software engineering tools with human participants," *Empir. Softw. Eng.*, vol. 20, no. 1, pp. 110–141, 2015.

Curriculum Vitae

Contact Details

Tallinn University of Technology
Department of Software Science
Akadeemia Tee 15A
12618, Tallinn, Estonia

Phone: (372) 585 212 13
Email: msurym@gmail.com
Homepage: mahunnah.wordpress.com/

Personal Details

Born in August 1983.
Tanzanian Citizen.

Education

| | |
|-------------|--|
| 2011 – | Ph.D. studies in Computer Science Tallinn University of Technology, Estonia |
| 2008 – 2009 | M.Sc. in Computing (Information Technology) Dublin Institute of Technology, Ireland |
| 2004 – 2007 | B.Sc. in Computer Science University of Dar es Salaam, Tanzania |

Employment

| | |
|--------|---|
| 2012 – | Assistant Lecturer Tallinn University of Technology, Estonia |
| 2008 – | Assistant Lecturer The Institute of Finance Management, Tanzania |

Elulookirjeldus

Kontaktandmed

Tallinna Tehnikaülikool
Tarkvarateaduse Instituut
Akadeemia Tee 15A
12618, Tallinn, Eesti

Telefon: (372) 585 212 13
E-post: msurym@gmail.com
Kodulehekülg: mahunnah.wordpress.com/

Isikuandmed

Sündinud augustis 1983
Tansaania kodanik

Hariduskäik

| | |
|-------------|--|
| 2011 – | Doktorantuur, info- ja kommunikatsioonitehnoloogia Tallinna Tehnikaülikool, Eesti |
| 2008 – 2009 | M.Sc. in Computing (Information Technology) Dublin Institute of Technology, Iirimaa |
| 2004 – 2007 | B.Sc. in Computer Science University of Dar es Salaam, Tansaania |

Teenistuskäik

| | |
|--------|---|
| 2012 – | Assistent Tallinna Tehnikaülikool, Eesti |
| 2008 – | Assistent The Institute of Finance Management, Tansaania |