University Library

A gateway to Melbourne's research publications

Minerva Access is the Institutional Repository of The University of Melbourne

Author/s:
Keogh, Kathleen Nora

Title:
Improvised coordination in agent organisations

Date:
2018

Persistent Link:
https://hdl.handle.net/11343/238701

Terms and Conditions:
Terms and Conditions: Copyright in works deposited in Minerva Access is retained by the copyright owner. The work may not be altered without permission from the copyright owner. Readers may only download, print and save electronic copies of whole works for their own personal non-commercial use. Any use that exceeds these limits requires permission from the copyright owner. Attribution is essential when quoting or paraphrasing from these works.

Improvised coordination in agent organisations

Kathleen Keogh

orcid.org/0000-0001-5798-3659

Doctor of Philosophy
July 2019

Melbourne School of Engineering

School of Computing and Information Systems

The University of Melbourne

Submitted in total fulfilment of the requirements of the degree of Doctor of Philosophy

Abstract

This thesis investigates coordination between intelligent software agents operating within agent organisations. Motivated by the prospect of agents working with humans in real world complex domains, the thesis focuses on flexible behaviour and improvisation in agent organisations. Methods used to design organisations of software agents are explored with particular consideration given to problem situations that cannot be defined with a detailed pre-scripted solution for coordinated action.

A conceptual model that describes the components that are needed in an agent based model in a multi-agent system is referred to in this thesis as a meta-model. A number of agent organisation-based meta-models and frameworks for coordination of agents have been proposed such as OperA, OMACS and SharedPlans. There is however, no specific meta-model or approach that addresses agent improvisation and unscripted coordination.

The reality of complex coordination in people's behaviour is analysed and used to develop requirements for agents' behaviour. A meta-model is proposed to include components to address these requirements. A process outlining how to design and implement such organisations is presented. The meta-model draws on features in existing models in the literature and describes components to guide agents to behave with flexibility at run time.

The thesis argues that coordinated agents benefit from an explicit representation of an organisational model and policies to guide agents' run time behaviour. Policies are proposed to maintain consistent knowledge and mutual plans between team members. Coordination is explicit and some flexibility is given to agents to improvise beyond the solution anticipated at design-time. Agents can mutually adjust individual plans to fit in with others so the multi-agent organisation is able to dynamically adapt to a changing environment.

The meta-model and design approach is successfully demonstrated and validated using an implementation of a simulation system. In this demonstration system, agents in multiple organisations collaborate and coordinate to resolve a problem within an artificial simulation world.

Declaration

This is to certify that:

(i) the thesis comprises only my original work towards the PhD except where indicated in the Preface,

(ii) due acknowledgement has been made in the text to all other material used,

(iii) the thesis is less than 100,000 words in length

Kathleen Keogh

Preface

During the time this thesis was developed, a number of public presentations and publications have been made which are based on the original work presented in this thesis. They are listed in this section for reference.

Early exploration of the problem was discussed at the First International Workshop on Agent Technology in Disaster Management at AAMAS 2006 and the Second International ISCRAM workshop on Information Systems for Crisis Response and Management in 2007. The motivation for knowledge cultivation between agents with reference to extending existing approaches using SharedPlans as discussed in Chapter 4 was presented at the Organised Adaptation in Multi-Agent Systems OAMAS08 workshop at AAMAS in 2008 and led to a publication in 2009. This analysis contributed to the development of the OJAzzIC model as discussed in Chapter 4. The OJAzzIC meta-model presented in Chapter 5 was first proposed at the 12th International COIN workshop held at AAMAS 2011. Further explanation of the use of social policies to guide the agents in an OJAzzIC organisation in Chapter 5, Section 5.6.2 was presented at the 16th International COIN workshop held at PRIMA in 2013 and published in 2014. The methodology outlined in Chapter 6 with an example of how OJAzzIC could be used to represent, design and implement a simulation system was described and presented at the COIN workshop held at PRIMA in 2014 followed by publication in 2015.

The following list includes all the individual publications related to this thesis:

- K. Keogh and E. A. Sonenberg (May 2006). "Agent teamwork and reorganization: exploring self-awareness in dynamic situations". In: *Proceedings of the First International Workshop on Agent Technology in Disaster Management, AAMAS, 2006*. Hakodate, Japan, pp. 56–63
- K. Keogh and E. A. Sonenberg (2007). "Designing self-aware agents for a disaster management scenario". In: *Proceedings of the Second International ISCRAM Workshop on Information Systems for Crisis Response and Management*
- K. Keogh, E. A. Sonenberg, and W. Smith (2009). "Coordination in Adaptive Organisations: Extending SharedPlans with Knowledge Cultivation". In:

Post conference proceedings- Organised Adaptation in Multi-Agent Systems (OAMAS08), LNAI 5368. Vol. 5368. Springer, pp. 90–107 *NOTE: This is a more detailed version of an earlier publication that was presented at Organised Adaptation in Multi-Agent Systems (OAMAS08) workshop Portugal, May 2008* K. Keogh, E. A. Sonenberg, and W. Smith (May 2008). "Coordination in Emergent Organisations: Extending SharedPlans with Knowledge Cultivation". In: *Proceedings of International workshop on Organised Adaptation in Multi-Agent Systems (OAMAS08)*

- K. Keogh and E. A. Sonenberg (2012). "Adaptive Coordination in Distributed and Dynamic Agent Organizations". In: *Coordination, Organizations, Institutions, and Norms in Agent System VII, LNAI 7254.* Ed. by S. Cranefield, M. Birna van Riemsdijk, J. Vázquez-Salceda, and P. Noriega. Vol. 7254. Heidelberg: Springer-Verlag, pp. 38–57 *NOTE: This is a more detailed elaboration on an earlier publication that was presented at COIN workshop, Taipei, May 2011*K. Keogh, E. A. Sonenberg, and W. Smith (May 2011). "Organizations with Improvised Coordination: OJAzzIC". in: *Proceedings of 12th International COIN Workshop, Internation joint conference of Autonomous Agents and Multi-agent Systems, AAMAS 2011.* Ed. by S Cranefield and P Noriega. Springer
- K. Keogh and E. A. Sonenberg (2014a). "Coordination Using Social Policies in Dynamic Agent Organizations". In: *Coordination, Organizations, Institutions, and Norms in Agent System IX, LNAI 8386.* Ed. by T. Balke, F. Dignum, M. B. Riemsdijk, and A. K. Chopra. Vol. 8386. Springer, pp. 1–20 *NOTE: This is a more detailed elaboration on an earlier publication that was presented at COIN workshop, PRIMA 2013 Dunedin, Dec 2013* K. Keogh and E. A. Sonenberg (Dec. 2013). "Coordination Using Social Policies in Dynamic Agent Organizations". In: *Proceedings of 16th International COIN Workshop, PRIMA 2013.* Ed. by T Balke and M B Riemsdijk
- K Keogh and E A Sonenberg (2015). "Designing for Planned Emergence in Multi-agent Systems". In: *Lecture Notes in Computer Science* 9372. Ed. by A. Ghose, N. Oren, P. Telang, and J. Thangarajah, pp. 97–113 *NOTE: This is a more detailed elaboration on an earlier publication that was presented at COIN, Coordination, Organisations, Institutions, and Norms in Agent Systems Workshop, 17th International Conference on Principles and Practice of Multi-Agent Systems, PRIMA December 2014. Gold Coast* K. Keogh and E. A. Sonenberg (Dec. 2014b). "Designing Organization-Aware Agents using OJAzzIC". in: *Proceedings of 16th International COIN Workshop, PRIMA 2014.* Ed. by A. Ghose, N. Oren, P. Telang, and J. Thangarajah

Acknowledgements

Thank you to my husband, Thomas for his unfailing support and encouragement without which this thesis would never have been completed. This thesis has been an intrusion and distraction from many other family and home responsibilities and Thomas has made sure that what had to be done at home was taken care of, over many years. I thank my children: Clare, Thomas, Patrick and Sarah for providing me with endless opportunities to keep some balance in my life, for their acceptance and understanding when I was distracted by my work and for their encouragement to finish.

Thank you to my primary supervisor Liz Sonenberg for her support, encouragement, advice and patience. A supervisor is so important, in so many ways toward the successful completion of a thesis and I particularly thank Liz for her careful questioning, timely prodding when needed and her patient understanding when life interfered in the progress of this thesis. I appreciate the academic guidance and mentoring provided by Liz throughout this PhD. Thank you also to my other supervisor, Wally Smith for his early support especially as my ideas developed in the exploration of the thesis topic.

I also wish to thank my advisory committee, Gil Tidhar and Rachelle Bosua for their support and willingness to discuss and review my work. I especially thank Gil for his interest in reviewing the thesis in its final stages.

I thank my honours supervisor many years ago, Gopal Gupta for planting a seed in my mind, telling me that I could write a PhD thesis one day.

I thank my parents, Anthony and Robin for bringing me up to believe I was capable of whatever I aimed to achieve.

Kathleen Keogh

Contents

Abstract iii

Declaration v

Preface vii

Contents xi

List of Figures xvii

Acronyms and definitions xx

1 Introduction 1

1.1 Overview 1

1.2 Motivation 2

1.2.1 Motivating Scenario 4

1.3 Research question 5

1.4 Scope and contribution 6

1.4.1 Scope 6

1.4.2 Contribution 8

1.5 Research objectives 8

1.6 Hypothesis 9

1.7 Overview of thesis 10

2 Research Approach 13

2.1 Overview 13

2.2 Design science perspective . . . 13
2.2.1 Identify problem and motivate . . . 14
2.2.2 Define objectives of a solution . . . 15
2.2.3 Design and development . . . 16
2.2.4 Demonstration . . . 16
2.2.5 Evaluation . . . 18
2.3 Simulation system . . . 20
2.3.1 Simulation World: Blocks World for Teams . . . 20
2.3.2 Problem scenario . . . 21
2.3.3 Requirements for agent behaviour . . . 24
2.4 Traceability of requirements to develop meta-model . . . 27

3 Background literature 29

3.1 Overview . . . 29
3.2 Dealing with complexity in human organisations . . . 30
3.3 Organisations, agents, meta-models, concepts . . . 35
3.3.1 Definitions and concepts . . . 36
3.4 Software Engineering methodologies for MAS design . . . 44
3.4.1 The design process . . . 45
3.4.2 O-MaSE . . . 50
3.4.3 GORMAS (Guidelines for Organisational Multi-agent Systems) . . . 51
3.5 Organisational meta-models in MAS . . . 52
3.5.1 Categorising organisational meta-models . . . 52
3.5.2 OMACS . . . 55
3.5.3 JaCaMo+ . . . 58
3.5.4 OperA+ . . . 59
3.5.5 SharedPlans . . . 61
3.5.6 Coordination in organisational meta-models . . . 63
3.5.7 Organisational adaptation in MAS . . . 68

3.5.8 Flexibility at the agent level: role and goal adoption 69
3.6 Supporting agent interaction 72
3.6.1 Social policies and social commitments 72
3.6.2 Models for knowledge sharing 76
3.7 Gaps and limitations in existing models 77
3.8 Conclusion 79

4 Toward a Grounded Model Design 81

4.1 Overview 81
4.2 Critical evaluation of existing meta-models with reference to requirements 83
4.2.1 OMACS meta-model 84
4.2.2 JaCaMo+ 86
4.2.3 OperA+ 88
4.2.4 SharedPlans 89
4.3 Case study - Ais Gill train accident 91
4.3.1 Overview of accident scenario 92
4.3.2 Development of situational knowledge regarding the incident . . 92
4.3.3 Coordinating interdependent dynamic goals 94
4.3.4 Evolution of the organisation structure 95
4.3.5 Development of plans in response to the incident 97
4.3.6 Multiple uncertain, partial and adaptive plans 98
4.4 Knowledge cultivation - extending SharedPlans 100
4.5 Representing the organisational knowledge at Ais Gill 104
4.6 Toward a flexible organisation meta-model 109
4.6.1 Building on OMACS meta-model 109
4.6.2 Meta-model design decisions 112
4.7 Conclusion 115

5 OJAzzIC Meta-Model for organisation and agents 119

5.1 Overview 119

5.2 OJAzzIC: Agents in Organisations Juggling Adaptation with Improvised Coordination . . . 120
5.3 Organisation meta-model . . . 124
5.3.1 Organisation meta-model structure . . . 124
5.3.2 Organisation run time execution model . . . 126
5.4 Agent meta-model . . . 128
5.4.1 Agent run time execution model . . . 129
5.5 Functional meta-model: goal trees and role model . . . 133
5.6 Coordination model . . . 137
5.6.1 Contracts . . . 139
5.6.2 Social policies and commitments . . . 140
5.6.3 Demonstration social policies for the OJAzzIC organisation . . . 143
5.7 Environment model . . . 148
5.8 Demonstrating the use of policies . . . 148
5.8.1 Demonstration use case description . . . 148
5.8.2 Using policies and commitments . . . 152
5.9 Evaluating OJAzzIC against requirements . . . 154
5.10 Simulation MAS Specification . . . 155
5.11 Comparing OJAzzIC to other models . . . 158
5.12 Conclusion . . . 160

6 OJAzzIC Design Process 163

6.1 Overview . . . 163
6.2 Adopting a design methodology . . . 163
6.2.1 Addressing requirements in the design process . . . 164
6.2.2 The design process . . . 164
6.3 Demonstrating the design process . . . 167
6.3.1 Define the functional goal model . . . 167
6.3.2 Define the organisation model . . . 170
6.3.3 Define the agent capabilities model . . . 172

6.3.4 Define the role model . . . 173
6.3.5 Define the coordination model: establish social policies . . . 174
6.4 Conclusion . . . 177

7 Implementation of a MAS organisation based on OJAzzIC 179
7.1 Overview . . . 179
7.2 Demonstration MAS system and scenarios . . . 180
7.2.1 Motivation for scenario testing . . . 180
7.2.2 Simulation system MAS implementation . . . 182
7.3 Scenario design settings . . . 183
7.3.1 Overview of different scenarios used . . . 183
7.3.2 Organisation configurations . . . 185
7.3.3 Task complexity during the rescue response scenario . . . 187
7.3.4 Policies . . . 191
7.3.5 Metrics . . . 191
7.3.6 Scenarios . . . 192
7.4 Rescue response system design . . . 195
7.4.1 Overview of implementation platform for agents . . . 195
7.4.2 Functional Goal model . . . 200
7.4.3 Organisation model . . . 201
7.4.4 Agent capabilities model . . . 203
7.4.5 Role model . . . 203
7.4.6 Social policies . . . 204
7.4.7 Policies addressing our flexibility requirements: initial observations 206
7.5 Results . . . 213
7.6 Discussion . . . 221

8 Conclusions 227
8.1 Overview . . . 227
8.2 Revisiting the research question . . . 227

8.3 Findings 229

8.4 Distinguishing OJAzzIC from other meta-models 232

8.5 Contributions 234

8.6 Future work 235

References **239**

A MAS system specification based on OJAzzIC **257**

List of Figures

2.1 Design science process model applied to our research 14

2.2 BW4T crisis maps 22

2.3 UML activity diagram for medic agent in stretcher rescue use case (OMG n.d.) 25

3.1 O-MaSE key tasks applied to produce design components 51

3.2 Positioning a selection of existing meta-models for coordination 54

3.3 Positioning a selection of existing meta-models regarding organisation structure 55

3.4 Modal Operators used in SharedPlans (Hunsberger 1999) 61

3.5 Coordination mechanisms in organisational meta-models 64

4.1 Abstractions for agent-goal relationships in OMACS 85

4.2 Agent User Story Requirements addressed by OMACS 86

4.3 Agent User Story Requirements addressed by JaCaMo+ 87

4.4 Agent User Story Requirements addressed by OperA+ 88

4.5 Agent User Story Requirements addressed by SharedPlans 90

4.6 Initial goal decomposition 94

4.7 Generic high level incident Plan 95

4.8 Development of organisation structures at Ais Gill 95

4.9 Potential recipe for transporting injured to hospital 98

4.10 Predicate definitions 102

4.11 Requirements drawn from analysis, related to agent stories (section 2.3.3) 116

5.1 Conceptual components in an OJAzzIC organisation 122
5.2 Life cycle of an organisation 123
5.3 The OJAzzIC organisation meta-model 124
5.4 Pseudocode for creating an adhocracy contract for coordination, *comments italicised* 128
5.5 Individual agent BDI deliberation cycle 130
5.6 Organisational agent BDI deliberation cycle 131
5.7 High level logic for organisation aware agent processing organisational attitudes (pseudocode) 132
5.8 Problem representation relating roles and tasks in an OJAzzIC organisation 134
5.9 Agent-Goal Abstraction in OJAzzIC 134
5.10 Example OJAzzIC Goal Tree 135
5.11 Predicates and definitions 144
5.12 Roles and Capabilities of agents in rescue scenario 149
5.13 Agent User Story Requirements addressed by OJAzzIC 154
5.14 Organisations and Agents in simulation system specification 156
5.15 Agent User Story Requirements addressed by each meta-model 159

6.1 O-MaSE key tasks applied to produce OJAzzIC components 165
6.2 Example objectives with details of required capabilities and sub-states . 169
6.3 Selected landmark objectives defined for rescue scenario, shown as GOAL statements 170

7.1 Organisation structure index 186
7.2 Tasks and their complexity 187
7.3 Social Policies implemented in demonstration MAS system 191
7.4 Scenario configurations 193
7.5 Partial GOAL code for organisational goal consideration by an agent . . 197
7.6 BDI interpreter cycle (A. S. Rao and Georgeff 1995) 199
7.7 A snapshot of goals for medic agent medic4 at a particular time during a run 200

7.8 Specification of OJAzzIC organisations for the rescue response system . 201

7.9 GOAL Code showing creation of an adhocracy with an officer delegated to carry stretcher 202

7.10 Agent capability specifications in the rescue response system 203

7.11 Role specifications in the rescue response system 204

7.12 GOAL code implementing policy for sharing beliefs of location of injured 205

7.13 GOAL code implementing policy to share objective reached beliefs . . . 205

7.14 Snapshot of goals adopted by medic3 during a run 207

7.15 Policy to find rescue stretcher partner based on agent capability not role, from Medic agent 208

7.16 Selection of messages and beliefs from medic3 210

7.17 UML sequence diagram (OMG n.d.) showing agent interactions demonstrating mutual adjustment of plans 211

7.18 Agent performance 214

7.19 Agent performance over time 215

7.20 Agent performance regarding number of rooms entered 216

7.21 Agent messages count for categories of organisational support 218

7.22 Agent Performance in 12 room stretcher rescues for Organisation index 6-9 220

Acronyms and Definitions

Acronym/Term	Definition
ABMS	Agent Based Modelling and Simulation (Adam and Gaudou 2016)
ACMAS	Agent Centred Multi-Agent System (Ferber, Gutknecht, and Michel 2004)
Adhocracy	An ad hoc, short term organisation that forms for decision making in a dynamic environment (Mintzberg 1979; D. Mendonca, Jefferson, and Harrald 2007)
BDI	Beliefs Desires Intentions (Bratman, Israel, and Pollack 1988)
BRAHMS	A modelling environment for work flow, interactions and behaviour of agents (Sierhuis, Clancey, and Hoof 2006; Sierhuis 2001)
DM	Disaster Management
EM	Emergency Management
FAML	A generic meta-model for MAS development (Beydoun, Low, Henderson-Sellers, Mouratidis, J.-J. Gomez-Sanz, Pavón, and Gonzalez-Perez 2009)
GORMAS	Guidelines for Organisational Multi-agent Systems (Argente, Botti, and Julian 2011; Esparcia, Argente, Vicente, and Botti 2014)
JaCaMo	Framework for developing MAS combining MOISE Organisation with Interaction, Environment and Agent dimensions (Boissier, J. Hübner, and Ricci 2016)
JaCaMo+	JaCaMo Framework combined with social commitments using artifacts (Baldoni, Baroglio, Capuzzimati, and Micalizio 2018a)
KB-ORG	A system for design of multi-agent systems (Sims, Corkill, and Lesser 2008)
KC	Knowledge Cultivation - processes to ensure knowledge is shared (Section 4.4, Page 100)
MOISE	Meta-Model for organisations involving missions allocated to groups of agents in order to address functional specification in schemes (Hannoun, Boissier, J. Sichman, and Sayettat 2000)
MOISE-*inst*	An extension to MOISE based on institutions (Gâteau, Khadraoui, Dubois, and Boissier 2005)
MOISE+	An extension to MOISE including structural, functional and deontic perspectives (J. Hübner, J. Sichman, and Boissier 2002; J.F. Hübner, J.S. Sichman, and Boissier 2007)

Acronym/Term	Definition
MAS	Multi-Agent System
Organisation	organisation of agents with capabilities, constraints, roles, groups and tasks (Ferber, Gutknecht, and Michel 2004)
Organisationally adept agent	an organisation aware agent capable of assessing how well it is performing in the organisation (Corkill, Durfee, Lesser, Zafar, and C. Zhang 2011)
Organisation aware agent	agent capable of organisational reasoning (with some awareness of the organisational structure and the roles and responsibilities of themselves as an agent in the organisation) (Van Riemsdijk, K. Hindriks, and C. Jonker 2009)
Organisation Meta-Model	A description of concepts, entities and relationships describing agents and groups and how they interact (Odell, Nodine, and Levy 2005)
OCMAS	Organisation Centred Multi-Agent System (Ferber, Gutknecht, and Michel 2004)
OJAzzIC	Organisations Juggling Adaption with Improvised Coordination (Section 4.4, Page 100)
OperA	A multi-agent organisation meta-model including interaction model, social model and organisation model (V. Dignum 2004)
OperA+	A context-aware extension to OperA (Jiang, V. Dignum, and Tan 2011)
OMACS	Organisation Model for Adaptive Computational Systems (S. A. DeLoach 2009)
O-MaSE	Organisation-based multi-agent software engineering methodology (S. A. DeLoach and Garcia-Ojeda 2010)
SharedPlans	Protocol for explicit creation of commitments and creating mutual shared intentions to ensure plan coordination between agents (B.J. Grosz and Sidner 1990)
RSMM	Reflexive Shared Mental Model of organisation structure (Section 4.4, Page 100)
SMM	Shared Mental Model, a shared mental representation of the situation (Smith and Dowell 2000).
TSMM	Task level Shared Mental Model (Section 4.4, Page 100)

Chapter 1

Introduction

1.1 Overview

Organisations of people can be faced with complex situations where a detailed planned response procedure does not exist, or where existing procedures cannot be performed exactly as planned. If these are critical situations, the ability to improvise appropriately and effectively can be the difference between survival and catastrophic systems failure (Trotter, Salmon, and Lenné 2013). This thesis investigates the possibility that complex artificial agents can similarly coordinate their behaviour with other agents and if necessary improvise to find a solution at run time.

Organisations of agents are examples of Multi-Agent Systems (MAS). A MAS involves a number of individual agents interacting as part of one system (Wooldridge 2002). In an organisation based MAS, similar to organisations in the real world, roles and associated responsibilities are defined. Individual agents enact roles within the organisation. Organisational objectives are expressed as organisational goals.

If artificial agents can improvise and coordinate, similar to human behaviour, there is potential for them to be teamed with people or to model people in simulation systems. This thesis is aimed at MAS developers who wish to develop Agent Based Modelling and Simulation (ABMS) systems in complex and dynamic domains such as disaster management. In particular, simulations where intelligent agents work together and communicate to coordinate their behaviour. Adam and Gaudou argue that ABMS would benefit from using complex agent models, particularly when modelling human behaviour (Adam and Gaudou 2016). In situations where agents are modelling human behaviour in responding to a complex situation, because of the dynamic nature of the problem, run time adaptation and improvisation based on a provided design is needed.

This thesis investigates adaptability, improvisation and coordination of agents in agent organisations. Consideration is given to situations when the available agents might not possess the exact capabilities to match the roles specified at design time. In dynamic situations, it is not appropriate to specify a detailed solution plan at design time because there are many possible contexts that could arise at run time. An approach that enables agents to improvise or adapt in order to find a run time solution, given the agents available at a particular time, offers more flexibility than a fixed design time solution. An organisation meta-model describes the conceptual components that define an organisation based MAS. Organisation meta-models are used to inform the design of a MAS for a particular situation. Currently, no organisation meta-models focus on addressing the requirements of improvisation and flexibility in agent behaviour at run time.

This chapter motivates the research in section 1.2 and a research question is articulated in section 1.3. The scope of the project is outlined and the contributions in the thesis are described in section 1.4. The chapter concludes with a description of the research approach in section 1.5 and the hypothesis that is brought to the experimentation is presented in section 1.6. An overview of the thesis is provided in section 1.7.

1.2 Motivation

The inspiration for this work comes from domains where people are challenged by complexity and need ad hoc problem solving (W. A. Mendonca D. . W. 2007). Life provides us with many complex situations when it is not possible to plan ahead with a well defined 'script' for problem solving. When individuals work together on a problem that involves interdependence between tasks, dynamic coordination between participants is required. Creating artificial agents with capabilities to improvise, coordinate behaviour and share knowledge during complex problem solving motivates the work in this thesis.

A multi-agency Emergency Management (EM) situation is used as a motivating example. In EM, a number of different agencies are involved (e.g. fire, medical and law enforcement). Although these agencies may work somewhat independently, it is important that they coordinate where resources need to be shared and plans need to align. The agencies need to ensure that they do not interfere with each other. Situations may arise where using initiative or improvising is appropriate for an individual to assist another to complete a task (even if it is not directly part of that individual's predefined responsibility). In EM in practice, the development of a solution may change

as a situation unfolds even though high level protocols are well defined and roles are defined for a situation. Adaptability and improvisation are required to modify and develop plans to suit the changing circumstances. Detailed plans emerge and roles and responsibilities are adapted to fit the unfolding situation (Valentine and Edmondson 2015).

Coordination is a difficult task. When problems occur in large scale incident management, it is often traced back to poor coordination and limited transfer of information across a system. The following quote regarding the response to the Black Saturday bush fires in Victoria in February 2009 is telling:

> *"It is recognised that failure to adequately respond to large-scale disasters is often not a problem of insufficient resources or technology, but lack of coordination. Causes of the failure include lack of communication between responders, slow circulation of often out dated information, and insufficient dissemination of situational data and action plans."* A. Au (2011). "Analysis of command and control networks on Black Saturday". In: *Australian Journal of Emergency Management* 26.3

This thesis investigates constructs for the coordination of agents in a MAS that enable agents to behave with flexibility to solve a problem dynamically at run time. This coordination relies on agents have a mechanism for creating and maintaining consistent beliefs and plans. In this thesis, dynamic problem solving performed in situations motivating this work is referred to as 'improvised coordination'. A meta-model specifies all the entities required in a model and the relationships between these entities. A MAS designed based on a meta-model should include all essential components specified in the meta-model.

There are benefits for people and agents in establishing a better understanding of the complexities involved in improvised coordination. Research has shown that when agents and people work together, having Shared Mental Models (SMM) improve performance (Scheutz, S. A. DeLoach, and Adams 2017). In complex team work, challenges exist that are not currently addressed with existing frameworks (Amir, B. Grosz, K. Gajos, Swenson, and Sanders 2015). It is important to implement systems including mechanisms for SMMs and update processes to that agents can reason and ensure that these models are maintained and kept consistent within the team (Scheutz, S. A. DeLoach, and Adams 2017). People do fail to achieve well coordinated activity or clear agreed goals in complex situations such as health care teams where mutiple teams or individuals are involved (Amir, B. Grosz, K. Gajos, Swenson, and Sanders 2015). Additionally, in such settings, goals and plans change and need revision. Maintaining

a consistent coordinated plan between all members of the team is necessary. Agent based frameworks such as SharedPlans (B.J. Grosz and Sidner 1990; B. Grosz and Kraus 1999) have been shown to improve performance in MAS. This work has then been useful to offer insights into the team work and necessary coordination mechanisms that human teams can adopt (Amir, B. Grosz, K. Gajos, Swenson, and Sanders 2015).

MAS have been used successfully to model human behaviour and provide insights into the analysis of human behaviour (Adam, Danet, Thangarajah, and Dugdale 2016; Adam and Gaudou 2016). There are benefits to simulation and training within EM using software agents engaged as virtual humans within a simulation scenario (Adam, Danet, Thangarajah, and Dugdale 2016; Zachary, Weiland, D. Scolaro, J. Scolaro, and Santarelli 2002). Enabling software agents to behave in a way that is observed as human-like leads to the possibility of them behaving as virtual humans in simulations to aid in training for complex coordination and management tasks (Adam, Danet, Thangarajah, and Dugdale 2016). The cognitive load on incident commanders responding to complex critical incidents can be helped by adopting technological aids (e.g. (Ntuen, Balogun, Boyle, and Turner 2006)).

Looking to the future, people could be aided by software agents assisting with knowledge sharing and coordination so that distributed problem solving tasks are easier to manage (Mentler and Herczeg 2015; Ntuen, Balogun, Boyle, and Turner 2006; Schaafstal, Johnston, and Oser 2001). A comprehensive computational model for agents coordinating knowledge, beliefs and plans and engaging in dynamic problem solving is needed.

In the next sub-section, we introduce a scenario that embodies the agent behavioural requirements that we are motivated to address.

1.2.1 Motivating Scenario

This section introduces a scenario of a rescue situation which will be used as a running example throughout the thesis. The scenario involves multiple agencies involved in rescuing injured individuals from a disaster area (A. S. Jensen, Alderwereld, and V. Dignum 2013). The scenario begins following a disaster event and the disaster area is a football arena. An unknown number of individuals are injured. The rescue task involves searching for the injured and transporting them to hospital for further medical treatment. The two rescue agencies involved in this example are a medical agency (Medics) and a law enforcement agency (Officers). The responsibilities of the Medics at the rescue scene are to rescue injured parties and deliver them to an particular zone for ambulance pick up. The objectives of the Officers is to ensure safety and order at the

rescue scene. Medic agents can delegate or request that Officers clear away bystanders who are impeding rescue progress because they are in an area where a rescue must take place. There are also bystander agents in the scene from two opposing football teams. Officers separate these agents into different areas to resolve fights that break out.

The situation demands that the rescue agents behave with some flexibility and potentially improvise in order to rescue the injured in a timely way. For example, if a Medic agent is ready to rescue an injured party on a stretcher and another Medic agent is not available to carry the stretcher, a nearby Officer agent with stretcher carrying capabilities could help carry the stretcher, even though the agent is not officially assigned to fulfil the Medic role.

For the purpose of this research, *flexibility* is defined as agents having the ability at run time to select individual goals autonomously based on the context of the situation in order to achieve the outcome of reaching desired organisation goals. The goals that agents select are independently chosen by each agent at run time. There is not a detailed solution plan specified at design time that specifies which agents will complete which tasks. *Improvisation* is defined as the ability for agents to behave outside of a specified role and responsibility set defined at design time in order to achieve a solution.

This thesis investigates situations where agents may need to improvise to act outside of their 'role description' or adjust their individual plans to fit in with other agents. Such situations can arise when agents begin a joint task for which coordination is not completely specified at design time or when agents agree to adopt the joint task before completely agreeing on a detailed plan specifying individual roles. When a planned solution fails, one approach is to redesign the organisation roles so that a solution is possible. Another complementary, more dynamic, approach is to allow agents to improvise based on the existing design. The latter approach is taken here.

1.3 Research question

In light of the above, the following research question is adopted:

> What elements are needed in the design of organisations of software agents, so that agents coordinate the sharing of knowledge and plans, and can demonstrate run time improvisation based on the design?

This question can be further expressed as:

1. What information needs to be available to agents about the domain, the organisation of software agents, the intentions and actions of other agents?

2. What elements need to be specified in a MAS at design time and what can emerge at run time?
3. What needs to be specified so that agents can improvise to take on a new role that needs filling when they have the capability to do so?
4. What needs to be specified so that multiple agents can coordinate their behaviour to achieve a joint task or share a role that cannot otherwise be allocated?

The research objectives are answered in this thesis through the creation of a meta-model, and the articulation and demonstration of a process for how to apply this meta-model to design an organisation based MAS. The meta-model includes a specification of agents in an organisation, an organisation structure and a set of behavioural policies and guidelines. These specify how multiple organisations could be created to form a distributed, coordinated system of interconnected agents and organisations. A MAS for a simulated rescue scenario is designed and implemented based on the meta-model.

1.4 Scope and contribution

1.4.1 Scope

The thesis considers an individual agent to have individual mental attitudes such as beliefs about the world, goals that represent desired states of the world and intentions adopted to reach those goals, as well as plans (sets of actions) to enact to reach those intentions. The thesis builds on the traditions of the Beliefs, Desires and Intentions (BDI) architecture for individual agents (Bratman, Israel, and Pollack 1988; A. S. Rao and Georgeff 1991), extended to include group activity e.g. (G. Tidhar 1999). The thesis adopts an agent organisation as providing a structure for a group of agents who have shared mental attitudes associated with the organisation. The organisation is structured using roles defining responsibilities. Relationships exist between roles, and the organisation can define obligations that member agents must accept. An instance of an agent organisation includes a set of agents each possessing individual capabilities, beliefs and intentions. Agents fulfil responsibilities within the organisation by adopting intentions to reach goals.

The characteristics of problem requirements motivating this work include:

- Multiple groups work within a larger organisation and share at least one high level objective;
- Within the groups, individuals work with individual autonomy *along with* some level of group awareness and a desire to achieve organisation goals;

- Interaction, coordination and cooperation between individuals/groups is needed in order to achieve goals with interdependencies;
- Membership of groups is fluid - individuals may leave or be unavailable;
- Plan elaboration is required - partial high level, incomplete plans may be adopted, then details elaborated upon or revised as the situation evolves;
- Agents enacting roles are not fixed. Opportunistically, members need to improvise in order to achieve goals in a timely way, based on who is available;
- The problem is distributed across multiple locations and central coordination and control is not possible.

In order to address the research question in section 1.3 a number of existing meta-models for agent organisations are analysed with respect to their ability to address the requirements. The requirements are elicited from experiences of people in EM. A new meta-model is proposed that combines and builds on existing approaches.

The OJAzzIC (Organisations Juggling Adaptation with Improvised Coordination) meta-model proposed provides a conceptual framework that can be used to create MAS models. The OJAzzIC meta-model is used to specify and implement a MAS in a virtual EM situation addressing the complex coordination requirements.

It is important to acknowledge that there is already a large body of related work defining and specifying norms, obligations and sanctions for agents in the context of an institution of agents. It is not our intent to contribute to the use and specification of institutions or norms in MAS. Both institutions and organisations can be used to coordinate agents, however, the analysis in this thesis is limited to organisation based MAS with an interest in agents belonging to organisations. Within the organisation, agents can adopt organisation objectives and work cooperatively, with awareness of others within the organisation to achieve the organisation goals. Mechanisms that can be used to enable agents to do this in a flexible, coordinated way are of interest. An institution is different to an organisation as it focuses on regulations, rules and laws rather than relationships and interactions between agents. We focus on systems that allow agents to behave with some flexibility whilst providing guidance and high level plans. It is presumed that, within an organisation, agents all share at least one high level organisation goal, which is not necessarily the case in an institution. Since, the institution does not include agents (Fornara and Balke-Visser 2018) but provides a mechanism to regulate agent behaviour, concentrating on an organisation that may include agents as role playing actors or directly as member parts of the organisation is beneficial. For further discussion about institutions of agents, the reader is directed to the recent work of Fornara and Balke-Visser, who discuss the modelling of both

institutions and organisations in MAS for open distributed systems (Fornara and Balke-Visser 2018). For further background regarding norms, the reader is directed to (F. Dignum 1999).

1.4.2 Contribution

The major contribution from this thesis is an architectural structure (a meta-model specifying entities and their relationships) for the design of agent based organisations addressing the particular requirements of improvised coordination. The meta-model includes a run time model for agent reasoning, a design time meta-model for the organisation structure and a run time model for organisations' life cycle. These artifacts are demonstrated through development of a simulation and comparison to other models. A methodology for designing adaptable MAS using the proposed meta-model concepts is presented based on adapting an existing methodology. The thesis also contributes to the design of complex socio-technical systems, particularly involving coordination between agents in order to share actions and achieve joint outcomes. Policies are suggested that help an organisation to behave in a coordinated way. Some experimentation is described to explore the effectiveness of adopting organisational policies.

The work presented in this thesis has been particularly informed by OperA (V. Dignum and F. Dignum 2012) introduced by Virginia Dignum, OperA+ (Jiang, V. Dignum, and Tan 2011) an extension to OperA, OMACS (S. A. DeLoach 2009) proposed by Scott DeLoach and SharedPlans (B.J. Grosz and Sidner 1990) described by Barbara Grosz. The coordination mechanisms and features of other organisation meta-models and frameworks including KBORG (Sims, Corkill, and Lesser 2008), JaCaMo (Boissier, R. H. Bordini, J. Hübner, and Ricci 2014) and MOISE (Gâteau, Khadraoui, Dubois, and Boissier 2005) are also examined. In describing a methodology for designing organisation aware agents using OJAzzIC, an adaptation of O-MaSE (S. A. DeLoach and Garcia-Ojeda 2010) is used. O-MaSE was developed by Scott DeLoach and colleagues as a customisable process methodology for designing agent systems, using OMACS as an example meta-model. O-MaSE is adapted for use with OJAzzIC.

1.5 Research objectives

The purpose of this research is to identify the components necessary to include in a MAS to enable agents to exhibit human-like behaviour. In particular, to enable agents to behave with some agility with respect to roles, to improvise in order to solve a problem and to coordinate their behaviour. In Chapter 2, we describe the research approach

adopted in more detail. A description of the approaches used for development and evaluation of the meta-model is provided, including

- developing clear requirements based on human behaviour
- analysis of existing meta-models
- requirements traceability to validate the meta-model
- the specification and creation of a MAS used to demonstrate application of the meta-model.

1.6 Hypothesis

The following hypothesis is tested in this thesis.

Hypothesis: Using the OJAzzIC meta-model and design methodology to implement a MAS, agents can be given capabilities but not be provided with detailed operational plans at design time, and will demonstrate the desired run time flexibility and improvised coordination.

In particular, agents in a MAS built based on this meta-model will demonstrate these behaviours:

i Agents will be able to improvise at run time to adopt goals if they have capability even if they are not assigned to relevant roles

ii Agents will be able to demonstrate mutual adjustment of individual plans to fit in with others

iii Agents will behave autonomously but prioritise organisational goals

iv Agents will be able to coordinate joint activities at run time even if they are not already in an organisation with their collaborators

v Agents will be able to negotiate and create plans with other agents to coordinate joint action at run time

vi Agents will demonstrate appropriate knowledge sharing within the organisation in order to behave efficiently

The performance of the implemented MAS is compared with the desired agent behaviour.

In the next section, an overview of the thesis is provided.

1.7 Overview of thesis

Chapter 2 provides an overview of the research methodology adopted for this research. The requirements being addressed are described using a rescue scenario. This rescue scenario is referred to throughout the thesis. An outline mapping how the requirements for improvised coordination can be traced through to components of the meta-model is also provided. A simulation platform is presented as an executable demonstration system. The simulated rescue scenario is used to model our requirements. It is created to test agents' ability to behave with flexibility and demonstrate the required features of coordination and flexibility in terms of role improvisation.

In Chapter 3, major background literature relevant to this work is introduced. Issues are highlighted that contribute to coordination as an interesting and complex topic - including communication of knowledge, knowing 'who' to tell and knowing 'what' is relevant. Complex coordination is described from two perspectives - from a human, real world perspective and from an artificial agent perspective. Also, some approaches to the explicit governance of agents in organised groups are provided.

The following organisational models are introduced in Chapter 3 giving consideration to their application to coordination of agents in an emergent, complex and dynamic situation: OMACS (S. A. DeLoach 2009), JaCaMo+ (Baldoni, Baroglio, Capuzzimati, and Micalizio 2018a) and OperA+ (V. Dignum 2004; Jiang, V. Dignum, and Tan 2011). Knowledge sharing and Shared Mental Models within agent groups are also discussed, focusing on mechanisms for maintaining consistent knowledge within a team. The use of SharedPlans (B.J. Grosz and Sidner 1990) to establish dynamic commitments between agents regarding plans is described. The use of policies and social commitments (Carabelea and Boissier 2006; M. P. Singh 2012) as mechanisms for explicit interactions between agents is also introduced.

In Chapter 4, a more detailed analysis of the selected meta-models is presented with respect to the requirements introduced with a scenario early in the thesis. These meta-models are examined in terms of their ability to address the requirements of improvisation and unscripted coordination within an agent organisation. A previous case study is then introduced. This case study describes a particular incident in EM in the real world, a train accident in the English countryside and the subsequent EM response to this event.

The remainder of Chapter 4 uses this case study to demonstrate the complex requirements of improvised cooordination, in a situation where it is not possible to script a detailed solution in terms of fixed roles, responsibilities and plans beforehand. An

analysis of the requirements of the social interactions and relationships between people engaged in complex coordination tasks is performed. The analysis begins with an examination of the communications and coordination that occurred. This analysis highlights the need for knowledge cultivation as well as coordination of plans amongst responders. Knowledge cultivation refers to the process of creating the appropriate interactions between agents so that the relevant knowledge regarding the incident and the solution is appropriately shared. SharedPlans (B.J. Grosz and Sidner 1990) are examined in the context of dynamic coordination involving a changing situation. The literature review focus is narrowed to MASs structured as organisations and in particular an analysis of OMACS (S. A. DeLoach 2009) is provided. Gaps in selected existing approaches related to our requirements are identified and implications for the meta-model are highlighted.

The motivation for knowledge cultivation between agents with reference to extending existing approaches using SharedPlans as discussed in Chapter 4 was presented at the Organised Adaptation in Multi-Agent Systems OAMAS08 workshop at AAMAS in 2008 [1] and led to a publication in 2009 [2]. This analysis contributed to the development of the OJAzzIC meta-model as discussed in Chapter 4.

Chapter 5 then presents the major theoretical account of the thesis in the form of the OJAzzIC meta-model. The proposed meta-model is based on features drawn from and extending existing work. An architectural description of an OJAzzIC organisation structure is provided. The OJAzzIC meta-model and associated processes are described. The meta-model includes a consideration of knowledge sharing and mechanisms for ensuring that appropriate knowledge is distributed across the organisation. The run-time OJAzzIC meta-model uses smaller, overlapping organisations dynamically created within a distributed organisation. This provides a network of connected paths for communication and knowledge transfer. The chapter concludes with a traceability analysis of the proposed OJAzzIC meta-model with reference to the requirements and a specification of a simulation system to highlight the operational functionality required in an implementation based on OJAzzIC. The OJAzzIC model was first proposed at the 12th International COIN workshop held at AAMAS 2011 [3] and published in 2012 [4]. Further explanation of the use of social policies to guide the

[1] K. Keogh, E. A. Sonenberg, and W. Smith (May 2008). "Coordination in Emergent Organisations: Extending SharedPlans with Knowledge Cultivation". In: *Proceedings of International workshop on Organised Adaptation in Multi-Agent Systems (OAMAS08)*

[2] K. Keogh, E. A. Sonenberg, and W. Smith (2009). "Coordination in Adaptive Organisations: Extending SharedPlans with Knowledge Cultivation". In: *Post conference proceedings- Organised Adaptation in Multi-Agent Systems (OAMAS08), LNAI 5368*. Vol. 5368. Springer, pp. 90–107

[3] K. Keogh, E. A. Sonenberg, and W. Smith (May 2011). "Organizations with Improvised Coordination: OJAzzIC". in: *Proceedings of 12th International COIN Workshop, Internation joint conference of Autonomous Agents and Multi-agent Systems, AAMAS 2011*. Ed. by S Cranefield and P Noriega. Springer

[4] K. Keogh and E. A. Sonenberg (2012). "Adaptive Coordination in Distributed and Dynamic Agent

agents in an OJAzzIC organisation in Chapter 5, Section 5.6.2 was presented at the 16th International COIN workshop held at PRIMA in 2013 [5] and published in 2014 [6].

In chapter 6, a methodology for the design of an OJAzzIC organisation MAS is described and demonstrated using the rescue scenario introduced earlier in Chapters 1 and 2. The methodology outlined in Chapter 6 with an example of how OJAzzIC could be used to represent, design and implement a simulation system was described and presented at the COIN workshop held at PRIMA in 2014 [7] followed by publication in 2015 [8].

The proposed meta-model, OJAzzIC is demonstrated in chapter 7 by using the meta-model to design and create a MAS to engage with a simulation using a scenario with requirements for improvised coordination. Agents in the MAS work together to resolve a rescue scenario. The run time behaviour of agents demonstrates improvisation and flexibility enabled by the constructs adopted from the meta-model.

Chapter 8 concludes with some discussion about the use of MAS to address the improvised coordination problem. The benefits of OJAzzIC as a meta-model during specification, design and implementation are highlighted. The limitations of this research are identified and further work is proposed to establish if OJAzzIC could be applied to different problems and different domains.

This chapter has introduced some motivating requirements addressed by this thesis and outlined the scope, contribution and research objectives. In the next chapter, the design science research approach and methodology used for this research is explained in more detail. The next chapter also introduces the simulation system scenario and platform used to demonstrate the high level requirements for flexibility and improvisation in agent behaviour. An outline of how the agent behavioural requirements and meta-model requirements can be traced through the thesis is also provided.

Organizations". In: *Coordination, Organizations, Institutions, and Norms in Agent System VII, LNAI 7254*. Ed. by S. Cranefield, M. Birna van Riemsdijk, J. Vázquez-Salceda, and P. Noriega. Vol. 7254. Heidelberg: Springer-Verlag, pp. 38–57

[5] K. Keogh and E. A. Sonenberg (Dec. 2013). "Coordination Using Social Policies in Dynamic Agent Organizations". In: *Proceedings of 16th International COIN Workshop, PRIMA 2013*. Ed. by T Balke and M B Riemsdijk

[6] K. Keogh and E. A. Sonenberg (2014a). "Coordination Using Social Policies in Dynamic Agent Organizations". In: *Coordination, Organizations, Institutions, and Norms in Agent System IX, LNAI 8386*. Ed. by T. Balke, F. Dignum, M. B. Riemsdijk, and A. K. Chopra. Vol. 8386. Springer, pp. 1–20

[7] K. Keogh and E. A. Sonenberg (Dec. 2014b). "Designing Organization-Aware Agents using OJAzzIC". in: *Proceedings of 16th International COIN Workshop, PRIMA 2014*. Ed. by A. Ghose, N. Oren, P. Telang, and J. Thangarajah

[8] K Keogh and E A Sonenberg (2015). "Designing for Planned Emergence in Multi-agent Systems". In: *Lecture Notes in Computer Science* 9372. Ed. by A. Ghose, N. Oren, P. Telang, and J. Thangarajah, pp. 97–113

Chapter 2

Research Approach

2.1 Overview

This chapter begins by providing an outline and description of the research approach used to conduct this research. The design science research approach is explained and the outcomes of each major process are described in section 2.2. The simulation platform used for experimentation, Blocks World for Teams (BW4T) (Johnson, C. Jonker, Reimsdijk, P. J Feltovich, and J. M. Bradshaw 2009) is also described in section 2.3. The chapter concludes with a clear specification of the requirements that are being addressed in this research and an overview of how these requirements can be traced through elements of the thesis in section 2.4.

2.2 Design science perspective

Our approach follows a design science research approach (Hevner 2007; A. Hevner, Salvatore, Jinsoo, and Sudha 2004; Peffers, Tuunanen, Rothenberger, and Chatterjee 2007). The design science research approach is a problem solving approach to research with roots in engineering and involves a systematic process progressing through stages in the research project. The first stage involves identifying the problem and defining solution objectives. The next stage involves specifying the artifacts to be designed or created in the process of addressing the research problem. A design cycle of creation and evaluation follows in which the artifacts and design processes are grounded with reference to existing knowledge bases. This grounding establishes a rigour in terms of referencing relevant existing work and establishing the contribution of the current research. There is also reference to relevant domain knowledge to specify the requirements to be addressed and identify processes for evaluation and testing of the success of

the artifacts created (Hevner 2007). The outcome is a design artifact that is developed and refined through the cycles of development and evaluation.

Using the context of emergency management in the real world as a motivator and drawing requirements from this, we ensure our design work is relevant. The outcome, the OJAzzIC meta-model is drawn from the literature and expertise in the area of multi-agent systems and in particular organisation centred multi-agent systems models.

In each of the following sub-sections, the key elements of each major process in design science research are described with reference to how they are achieved in this thesis. As shown in figure 2.1, adapted from the design science research method described by Peffers (Peffers, Tuunanen, Rothenberger, and Chatterjee 2007), we adopt a problem centred approach to motivate our work. We define objectives then create a subsequent design and development based on gathered requirements.

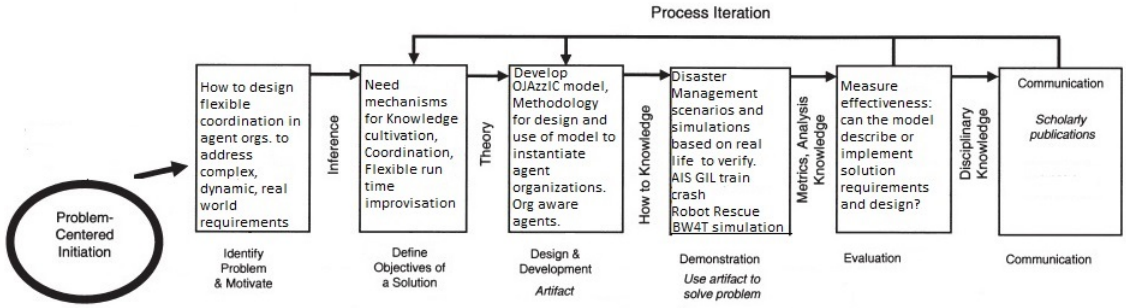


Figure 2.1: Design science process model applied to our research

2.2.1 Identify problem and motivate

As introduced in Chapter 1, there are complexities in distributed problem solving that require emergent coordination between parties to achieve a solution. These complexities arise in particular domains and problem types. As discussed in Chapter 3, currently, many MAS organisational models have been designed based on a top down approach. Solutions are created based on a problem hierarchy and organisational design specified at design time, with limited scope for agents to behave outside these specifications at run time. In dynamic situations, a predetermined organisational design may not fit and a solution is only possible by applying flexibility when adopting the original design. We identify in particular the need to have an organisational meta-model that facilitates agents improvisation at run time. There are no organisation based MAS meta-models that directly address our requirements.

2.2.2 Define objectives of a solution

The research objectives for this thesis included the creation of design artifacts. The main artifact produced is an organisation meta-model for MAS and a method for how to use this to design and implement an organisational based MAS. In the design science approach, design artifacts are built and evaluated in a design cycle. Design science research produces artifacts in the form of constructs, models, methods or instantiations. In this thesis, a number of additional artifacts are developed in order to support the creation of the main design artifact. The artifacts produced as part of the research process include:

- a clear set of requirements to be addressed (grounded in reality, based on analysis of human performance in complex dynamic domains) (Chapter 4). These requirements are used in the modeling process to create the meta-model;
- a comparative predictive analysis of existing meta-models' performance based on requirements (Chapter 4);
- a conceptual analysis of existing models and features offered in such models as applied to real case studies (Chapter 4);
- an organisation meta-model for MAS that describes abstractions that support the requirements of flexibility and improvisation in agent run time behaviour. This meta-model includes an organisational structure, a problem decomposition (functional) approach, a social dimension defining agent interaction including a mechanism for agent coordination and an ability to specify how agents can be flexible in using an organisational design at run time (Chapter 5);
- a design methodology describing the process of using the meta-model to design a MAS system, including design considerations regarding organisation policies to guide coordination. This also includes a description of the agent deliberation process in selecting actions and a process for the creation of ad hoc organisations (Chapters 5 and 6);
- an application of the meta-model to create a demonstration multi-agent based system addressing required features of coordination and flexibility in terms of role improvisation (Chapter 7).

Success in reaching these objectives is measurable by demonstrating success in creating the artifacts, demonstrating the meta-model in paper based analyses and evaluating the meta-model by using it to implement a run time system.

2.2.3 Design and development

The main design artifact created in the process of this research, the OJAzzIC meta-model for an organisation of agents, is presented in Chapter 5. It includes an organisation structure and a set of behavioural constraints and guidelines for how such structures could be created to form a coordinated system of interconnected organisations. The meta-model describes the organisation using roles, capabilities, requirements and a goal-task model as well as supporting policies for the creation of contracts between agents. Additionally, a framework is described for the use of the OJAzzIC meta-model to create overlapping adhocracies in an emergent system. This framework includes social policies and a description of an adapted BDI agent deliberation cycle so that organisational agents consider organisational objectives as well as individual objectives.

In the process of designing the OJAzzIC meta-model, an analysis of the requirements of the social interactions and relationships between people engaged in complex coordination tasks is performed. The analysis begins with a detailed study of a train accident in the English countryside and the subsequent communications and coordination that occurred. The analysis of communication and coordination highlights the need for knowledge cultivation as well as coordination of plans amongst responders. This is presented in Chapter 4. The outcome of this analysis was used as input to a build phase in constructing the main artifact, the meta-model. The organisation meta-model includes components for information sharing and coordination based on the existing explicit coordination commitment protocols presented in SharedPlans (B. Grosz and Kraus 1999; B. Grosz and Hunsberger 2006), extended to explicit coordination of knowledge.

In the process of evaluating the usefulness of the proposed OJAzzIC meta-model, the conceptual analysis forms part of the evidence that the features within the meta-model address the requirements. This conceptual analysis is presented in Chapters 4, 5 and 6.

2.2.4 Demonstration

In the process of describing and demonstrating the use of the OJAzzIC, further artifacts were created. An agent based simulation model was implemented based on the OJAzzIC meta-model. A descriptive methodology outlining the considerations to be aware of when designing a system for flexibility in agent behaviour is described in Chapter 6 and evaluated in Chapter 7.

An agent based simulation system is a simplification of a system and the model designer chooses which aspects of the real system to include. Given the complex-

ity involved in human decision-making, this choice is a major challenge (Balke and Gilbert 2014). To demonstrate that the meta-model addresses our knowledge cultivation requirements, we use scenarios that require coordination through social interactions emerging within an organisation.

A requirement of our agent organisation model is that agents are able to appropriately coordinate with each other and interact with other agents to share relevant knowledge and information regarding the situation. In order to demonstrate that the OJAzzIC meta-model addresses this requirement, in Chapter 4 we use a scenario based on a case study to illustrate how policies can ensure the creation of social and information contracts that achieve knowledge cultivation and coordination obligations.

In the next cycle toward a more detailed specification of the agent organisation meta-model, we examine the requirements drawn from the case study and identify implications for the design of the meta-model. Based on this analysis, the organisational structure is developed building on the OMACS (S. A. DeLoach 2009; S. DeLoach and M. Miller 2010) structure to decouple the problem definition in terms of goals and tasks from the organisational hierarchy of roles. Informed by contracts used in OperA (V. Dignum 2004), the OJAzzIC meta-model adopts contracts between agents within an organisation that specify and agree on roles. The meta-model is further specified and refined by defining social policies as the meso-layer of specification. The social policies enable the specification of guidelines that can be specified at design time to influence run time behaviour, particularly with reference to agent flexibility.

We use a scenario from the emergency mangement domain to demonstrate our use of policies in OJAzzIC to ensure coordination and knowledge sharing within and between organisations. We describe a solution based on the OJAzzIC meta-model. We create multiple organisations, each working autonomously on individual tasks but acting in a context of awareness of others in the network. The scenario is set during the incident management phase immediately following an event such as an explosion in a built up area. The system organisation has the goal to resolve the disaster - rescue the injured, give first aid, and transport the injured to hospital. The main service response agencies are allocated sub goals in the high level plan, based on their service. For example the goals of the ambulance agency are to provide immediate medical care and casualty transport to hospital. The goals of the fire agency are to manage hazards, protect the site, and extinguish fires. We demonstrate using a walk-through of part of the unfolding scenario to illustrate the coordination management and how OJAzzIC policies can be used to ensure appropriate knowledge sharing and cultivation of relationships between organisations.

2.2.5 Evaluation

Our meta-model and the process methodology for its use are evaluated by applying them to describe and implement a system embodying characteristics of the improvised coordination requirements. The system developed based on the meta-model and subsequent experimentation used to evaluate the artifacts are described in detail in Chapter 7. Using OJAzzIC to design and implement a MAS for a simulation system, we demonstrate the methodology for how requirements should be considered during design process. The methodology for using OJAzzIC is outlined in Chapter 6 and applied in Chapter 7. The process of using the model to implement a run time system provides an opportunity to clarify specifications and requirements as well as ensure the design is indeed robust enough to encode and address our requirements.

We focus on the requirements of agent flexibility at run time and improvised coordinated behaviour. We test the capability of agents to improvise regarding role adoption at run time. In particular, we evaluate if the meta-model design sufficiently specifies tasks, capabilities, roles and policies at design time to result in a MAS that is able to improvise at run time, without detailed design time solution scripts. Of course, a solution is possible by 'hardwiring' behaviour at design time with detailed synchronised plans. We are interested in the design time specification of high level plans and policies combined with run time commitments between agents that facilitate agents to coordinate and adopt shared plans.

The experimentation tests agents' ability to improvise at run time beyond roles specified in a top down design. The adopted simulation scenario involves agents adopting goals based on matching capabilities rather than being allocated into a role. Agents are engaged to coordinate on a complex task that requires multiple agents to work together. Using blocks world for teams (BW4T) (described in section 2.3.1), we implement a scenario involving agents controlling robots in a simulated search and rescue scenario.

Experimental design One of the important dimensions that could be used to examine agent based simulation systems is the social dimension. The social dimension refers to the ability of the agent architecture to provide agents with capabilities to reason about social relationships and represent social concepts such as we-intentionality and theory of mind (Balke and Gilbert 2014). Another dimension is the agents' ability to reason about social norms and the architecture's ability to allow the emergence of social norms (Balke and Gilbert 2014). In order to test the ability for an agent organisational architecture to address requirements, we can attempt to use it to specify these social

requirements.

The simulation environment tests two particular situations. The first situation tests role improvisation. This involves a scenario where an agent has the capability to complete a required objective, but is not directly allocated this objective based on role allocations. The agent can however improvise and adopt this objective when the need arises. The need may arise when another agent with the required capability is not available. We establish policies at design time to specify the tasks that may allow improvisation around how they are adopted and by which agents.

The second situation involves a joint rescue task that requires two agents coordinate at run time to achieve one task simultaneously (role sharing). The agents in this situation both create run time commitments to each other to adopt the task simultaneously and then work together to achieve that task.

To investigate the ability for agents implemented using the OJAzzIC meta-model to appropriately share information, coordinate and improvise, we measure their success in completing rescues using a number of different scenarios with varying levels of complexity and levels of organisational support for agents, including a situation where agents have no organisational support at all. Within our trials, we experiment with varying the number and type of organisational policies provided at design time and measure the impact of this on success. The policies tested include obligations for sharing of beliefs, intentions and actions with other agents in an organisation. We also enable some flexibility with a policy to allow an agent involved in a collaborative joint task to handover its partner's task to another agent nearby.

In the remainder of this chapter, a problem scenario and simulation platform is introduced in section 2.3.2. The scenario presented builds on the motivating EM scenario presented in the previous chapter in section 1.2.1. This will be used as a running scenario throughout the thesis. A simulation platform used for experimentation, Blocks World for Teams (BW4T) (Johnson, C. Jonker, Reimsdijk, P. J Feltovich, and J. M. Bradshaw 2009) is described. A clear statement of the high level requirements for agent behaviour that are addressed in this thesis is provided. The chapter then concludes with an explicit discussion explaining traceability of the requirements through to the development of the meta-model. An outline is provided to highlight how the requirements can be traced through into the design, use and validation of the meta-model.

2.3 Simulation system

In this section, the simulation system used to demonstrate the application of the meta-model is described.

2.3.1 Simulation World: Blocks World for Teams

In order to evaluate our meta-model, we chose to implement a simulated rescue scenario using Blocks World for Teams (BW4T) (Johnson, C. Jonker, Reimsdijk, P. J Feltovich, and J. M. Bradshaw 2009). BW4T is a testbed for coordination and provides us with a controllable and repeatable scenario simulation that is easily configured with complex coordination requirements. It is possible to design scenarios that involve interdependence between tasks and require coordination between agents. This provides us with a situation where we can test information sharing and task coordination between agents within organisations.

Previously, BW4T has been used to conduct experiments within agent teams regarding agent interaction, coordination and shared mental models (Harbers, Van Reimsdijk, and C. Jonker 2012; Harbers, J. M. Bradshaw, Johnson, P. Feltovich, Bosch, and J.-J. Meyer 2011; A. S. Jensen, Alderwereld, and V. Dignum 2013; Johnson, C. Jonker, Reimsdijk, P. J Feltovich, and J. M. Bradshaw 2009; S. Li, W. Sun, and T. Miller 2015; R. Singh, E. A. Sonenberg, and T. Miller 2016).

BW4T has been developed as a simulation environment that interfaces to both agents and humans. The agent interface allows for implementing agents using Java or the GOAL programming language (Hendriks and Dix 2014). The simulation environment is composed of a number of rooms and hallways connecting the rooms. In each room, there may be any number of coloured blocks or no blocks. There may be a number of robots who are in the environment. Robots can be controlled by humans or agents. In our experimentation, we use agents only to control the robots. Robots can perceive the location and colours of blocks after entering a room. Initially, no agents have awareness of where the coloured blocks are located. Only one agent is allowed in a room at any one time. The aim of the basic simulation is for robots to collectively retrieve blocks of required colours in a specified sequence and deliver them to the 'drop-zone' area. If blocks are delivered in the incorrect sequence, they are removed from the simulation. Success may be measured in terms of whether the robots complete the task by delivering the blocks in the correct sequence, the time taken to do this and the number of rooms entered by the robots.

Jensen and colleagues extended the basic BW4T simulation environment to rep-

resent an emergency response scenario, where red blocks represent injured people to be rescued by delivery to the dropzone (representing the ambulance) and the robots were in teams representing agencies - medical agents and officer agents (A. S. Jensen, Alderwereld, and V. Dignum 2013). Cooperation and delegation opportunities arise in this simple scenario and Jensen and colleagues use this to examine relevant dimensions of organisations. We used an extension of the simulation system developed by Jensen to develop a testbed simulation system for our meta-model. These simulations are described in section 2.3.2.

2.3.2 Problem scenario

In this section, we provide more detail to the scenario introduced in section 1.2.1.

The simulated rescue scenario tests the ability of agents to work together in a coordinated way and demonstrate flexibility in agent behaviour and will be referred to throughout the thesis.

The rescue responder simulation demands flexible, coordinated agent behaviour. Situations arise when agents may need to adjust their individual plans to fit in with other agents. This can occur when

- agents begin a joint task for which coordination is not completely specified at design time, or
- when an agent agrees to a plan and then is not available when needed or
- when agents agree to adopt tasks to achieve a goal before completely agreeing on a detailed plan specifying individual roles to be enacted.

The simulation system and use case adopted is based on an emergency rescue scenario. We use the blocks world for teams (BW4T) environment with a number of rooms (Johnson, C. Jonker, Reimsdijk, P. J Feltovich, and J. M. Bradshaw 2009). This environment has been used by others to test teamwork between agents and is described in section 2.3.1. We adapt the use of BW4T to represent a rescue responder task as described by Jensen (A. S. Jensen, Alderwereld, and V. Dignum 2013). In this situation, the drop zone represents an ambulance pickup zone and coloured blocks represent injured people that need to be rescued. The injured are distributed across multiple rooms. As agents control the robots in our experimentation, we refer to the robots as agents. The rescue process involves an agent picking up the coloured block, taking it to the dropzone and putting it down in the dropzone. Agents are unaware of where the injured are located until rooms are searched. Only one agent is allowed in a room at any one time.

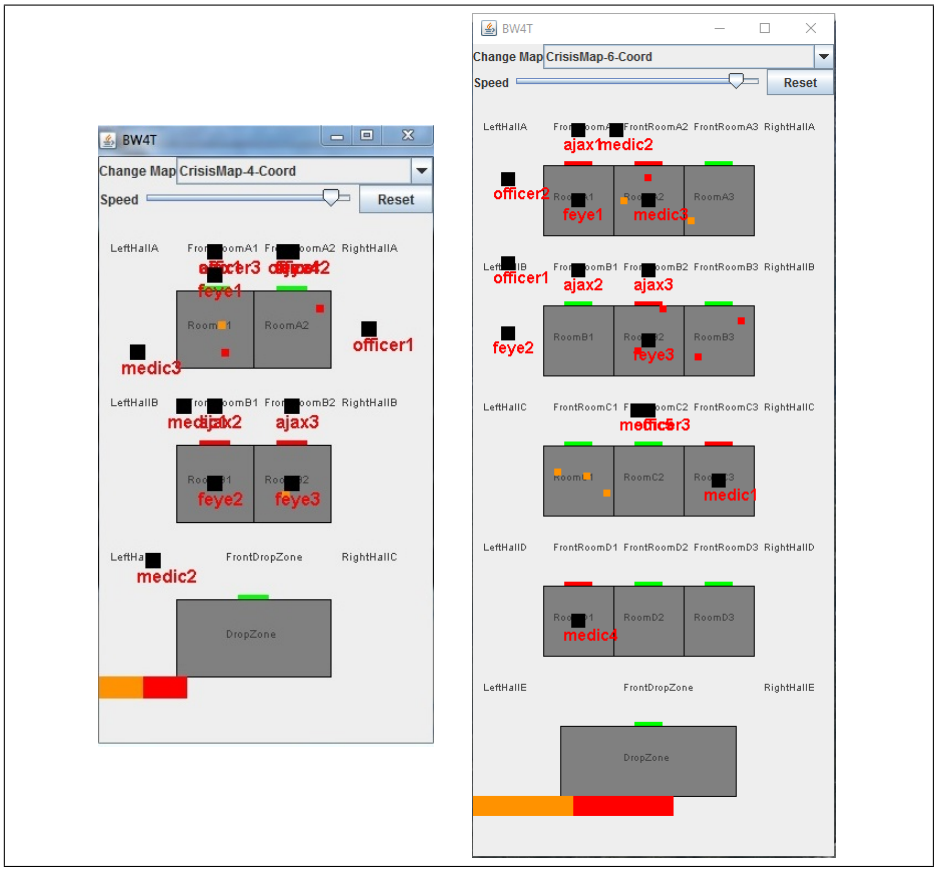


Figure 2.2: BW4T crisis maps

In the simulation, agents may be one of 3 types of agent: Bystander agent, Officer agent and Medic agent. By default, an organisational role (set of responsibilities) is associated with Medic and Officer agent type. Each agent type has a set of capabilities and actions that can be performed. The behaviour of Bystander agents is to interfere with the actions of other agents by entering rooms at random. Officer agents are responsible for clearing away Bystanders from rooms (by instructing a located Bystander agent to leave a room) and breaking up fights between Bystanders (by instructing the Bystanders to move elsewhere). Medic agents are responsible for finding the injured blocks and delivering the injured to the ambulance (drop zone). When an agent enters a room, the agent receives percepts from the environment to inform them of the existence and location of any (injured) coloured blocks in the room.

Figure 2.2 shows the interface of the BW4T map at the beginning of a crisis for two different scenarios. The simulation can be configured to change the number of agents, the number of rooms and the number of injured blocks requiring rescue. This figure shows on the left a configuration with 3 *medic* agents, 3 *officer* agents and 8 *bystander* agents. There are 4 rooms and a total of 2 orange blocks and 2 red blocks. The red blocks are in Room A1 and Room A2. The orange blocks are in Room A1 and

Room B2. The configuration on the right map is more complex. There are 5 *medic* agents, 3 *officer* agents, 6 *bystander* agents and 12 rooms. Increasing the number of agents increases the potential number of agent pairs who could perform a stretcher rescue. Increasing the number of rooms and number of injured results in more work to be completed in both search and rescue. There are 5 orange blocks and 5 red blocks to be rescued. Bystander agents are labeled as feye$n$ or ajax$n$.

The tasks we describe require agents to coordinate to achieve a combination of joint tasks as well as individual tasks that may be completed by agents autonomously and independently. At run time, agents adopt tasks according to availability, ability and individual priorities. Specifically, an agent who is assigned the role of Medic can adopt objectives that are associated with the medic role such as search, individual rescue, stretcher rescue. An agent assigned the role of Officer can adopt objectives associated with the officer role. In addition to roles, individual agents can be given specific individual capabilities (e.g. locate injured, stretcher rescue). At run time, agents may in some cases adopt goals to achieve tasks based on possessing the capability to perform those activities even if they are not allocated to a role responsible for achieving that goal.

In the simulation environment, there are a number of red and orange coloured blocks distributed across different rooms. The aim is for agents to move each of the coloured blocks to the dropzone. A red block represents an injured person who is mildly injured and can be rescued (picked up) and taken to the ambulance drop zone by one medical officer alone. An orange coloured block represents a seriously injured person that needs to be rescued with a stretcher and taken to the ambulance drop zone.

In our rescue response use case scenario, we deliberately specify rescue tasks requiring coordination between agents. As only one agent may enter a room at any one time, the virtual stretcher rescue requires two agents to coordinate their actions. Two agents involved in a joint stretcher rescue, must at run time negotiate which particular role they will adopt when enacting a shared rescue together. One agent must adopt the role of support (waiting at the door and then 'carrying the stretcher') whilst the collaborating partner agent adopts the role of providing the stretcher and putting the injured onto the stretcher (entering the room, picking up and holding the injured block). This run time behaviour requires that the agents have a dynamic mechanism for role selection or mutually adjust to adopt the correct role to fit in with their partner agent. The stretcher rescue task involves two agents first agreeing that they have a mutual plan to adopt that rescue task, then both dynamically enacting (and perhaps adjusting) individual plans to ensure the rescue is completed. The task requires that the two

agents communicate, collaborate and coordinate their individual adopted intentions and actions in order to complete the joint task simultaneously.

The use case for enacting a stretcher rescue requires that at run time, the 2 collaborating rescue agents need to mutually agree to complete the task together and individually adopt unique roles - i.e. support and pick up roles. Figure 2.3 presents a UML activity diagram (OMG n.d.) for an agent performing tasks to achieve the goal of a stretcher rescue. This shows the complexity involved in two agents proposing, reaching agreement and creating a commitment to perform a stretcher rescue. Medic agents initially search rooms to establish where injured blocks are that require rescue. If bystander agents are in a room so that they are obstructing a rescue, then an officer agent or medic agent with capability to remove bystanders will clear the bystanders by instructing them to move away.

A stretcher rescue requires that 2 rescue agents work together on a joint task to carry the injured block on a virtual stretcher to the drop zone. In most cases, the 2 stretcher rescue agents are medic agents with the capability to perform a stretcher rescue as part of their role, however, it is possible that a medic agent can delegate carrying a stretcher to an available officer agent. For instance, if 2 medic agents agree to perform a rescue together and then one is not available, the other agent can delegate the stretcher carrying role to an alternate officer agent who is nearby rather than waiting for the other medic partner to be available.

In order to test the ability for agents to adapt and improvise at run time beyond the roles specified at design time, we consider the possibility of agents in a particular role being unavailable. For example, if a stretcher rescue needs to occur and only one medic agent is available and a capable officer agent is nearby. Can the system adapt to allow the officer agent to help with carrying the stretcher?

2.3.3 Requirements for agent behaviour

The high level requirements for agent behaviour that are addressed in this thesis were introduced in section 1.6. In this section, we present these requirements, particular to the problem scenario expressed in terms of agent stories[1]. In brackets after each requirement, the corresponding general behavioural requirement from section 1.6 is provided. These stories describe the expected agent behaviour in the simulated emergency search and rescue scenario:

[1]The agent stories are expressed in the first person in a similar way to user stories that describe requirements

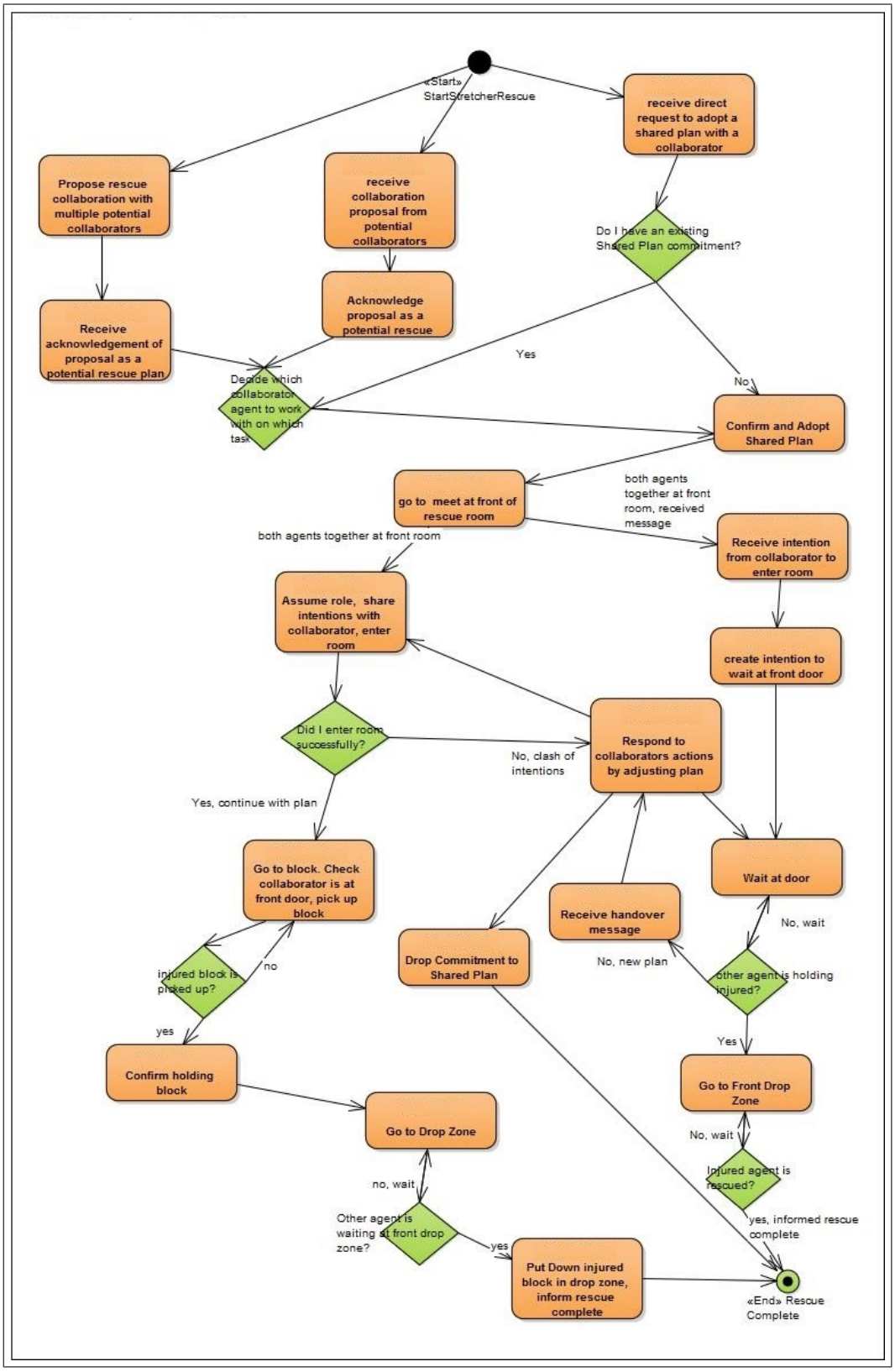


Figure 2.3: UML activity diagram for medic agent in stretcher rescue use case (OMG n.d.)

1. Adopt goals and work with others: (i) *as an agent with capabilities to act to change my environment, I wish to adopt goals that I am capable of achieving so that I support the organisational goals and resolve a problem. If I am capable of achieving part of a designed high level goal, I am happy to do so in collaboration with other agents who can achieve the necessary complementary required sub-goals in order to achieve the mission or organisational goal*;
2. Be aware of others and mutually adjust to fit in with others: (ii) *as an agent, I wish to be aware of other agents working in the space and able to fit in with others so that we can dynamically coordinate to work together to resolve a problem. This may involve planning a solution, agreeing on roles and then executing the plan, or partially planning a solution and then adjusting our goals to fit in with others as the detailed plan unfolds. For example, if I as a medic agent, have agreed to help with a stretcher rescue with another medic agent with the initial plan that I will meet my partner with the stretcher, but then my partner arrives first, I will adjust my own goals to fit in with the situation that my partner has begun to put the patient on the stretcher, then I will instead adopt the support role, carrying the stretcher. I will ensure to communicate my intentions with other agents to achieve coordination where necessary*;
3. Prioritise organisation goals: (iii) *as an agent, I wish to act autonomously to adopt goals according to my own priorities, whilst giving priority to the organisational goals so that these are achieved*;
4. Rescue injured: (iii) *as a medic agent, I wish to find and rescue injured to move them to the ambulance drop zone so that the rescue is complete*;
5. Coordinate to perform stretcher rescue: (iv, v) *as a medic agent, if I find a seriously injured patient, I wish to work together with another medic agent so that we coordinate a stretcher rescue to carry the injured patient to the ambulance drop zone. If another medic agent is not available (even if we previously agreed to do the rescue together), I am happy to request assistance with carrying the stretcher from another capable agent nearby so that the rescue can be completed in a timely way*;
6. Perform tasks autonomously: (iii) *as an officer agent, I wish to clear the area of bystanders so that the medic agents can proceed with search and rescue tasks*;
7. Help others if I can: (i) *as an officer agent, if I am capable of carrying a stretcher, I am willing to help medic agents in this way if asked, even though this is not part of my default role responsibilities, so that the rescue can be completed in a timely way*;

8. Share information about my intentions and actions: (vi) *as a collaborating agent, I will share information with* appropriate *other agents to inform them of my intentions and actions performed so that we can all work together efficiently*;

These stories describing agent behaviour will be referred to throughout the thesis.

2.4 Traceability of requirements to develop meta-model

The purpose of this section is to provide an overview of how the requirements can be traced through to the development of the meta-model and the MAS simulation.

In Section 2.3.3, the high level requirements are expressed in terms of agent stories to describe the desired agent behaviour. In Chapter 4 we clarify the requirements based on human case studies to form a set of meta-model requirements that will be addressed in the OJAzzIC meta-model. These meta-model requirements are summarised in Figure 4.11.

The requirements developed in chapter 4 are considered in the development of the meta-model. The OJAzzIC meta-model is presented in Chapter 5 in section 5.2 using MOF(UML) (OMG 2016) to represent the entities in an organisation. The meta-model is presented as a description of a number of meta-models: The meta-model includes an organisation meta-model, an agent meta-model, a functional meta-model and a social coordination meta-model specified using policies, commitments and SharedPlans[2].

The organisation meta-model is first described in section 5.3 in terms of an organisation as an entity and what it includes. In Section 5.4, we describe the OJAzzIC agent meta-model as an extension to the traditional agent BDI deliberation cycle to include two levels of agent awareness of others within an organisation — i.e. agents not only consider other agents in their own individual deliberations, but agents also deliberate with others in an organisation. We describe the meta-model for tasks and goals and how they relate to organisation roles in section 5.5. The interaction and coordination meta-model is described with contracts, social commitments, SharedPlans and policies.

The social model in terms of policies, commitments and contracts is described and then a number of example policies are described using first order logic and some defined predicates. In Section 5.6.2 we provide some justification regarding the decision to use social commitments and policies based on related work. In sections 5.8 and 5.8.2, we demonstrate how policies and commitments can be used to help with coordination. The

[2]The need for agents to agree on a coordinated action plan is of course, a generic requirement, in the thesis we adopt and use commitments to SharedPlans (B.J. Grosz and Sidner 1990) to realise this requirement.

scenario introduced in section 2.3.2 is also used in section 5.8.2 to demonstrate policies and how they are used. The scenario is further elaborated upon in section 5.8 with a partial use case description and a number of possible alternative use case scenarios. These use case scenarios are used to illustrate how policies can ensure the creation of social and information contracts that achieve knowledge cultivation and coordination obligations. An explanation of how policies are used to guide agents with run time improvisation is also provided.

The same scenario is also referenced in chapter 5 in the Z specification of a MAS in section 5.10. A selection of these policies discussed in section 5.8.2 are again defined in the Z specification in section 5.10. The Z specification of an emergency response system for this scenario specified in this section forms a specification for the system to be implemented using a MAS designed based on OJAzzIC.

In Chapter 4.2, A number of MAS meta-models are compared with reference to their ability to address the agent stories introduced as requirements in section 2.3.3. OJAzzIC is compared in more detail with a number of existing models to demonstrate the benefits of adopting the OJAzzIC meta-model.

The executable implementation of the MAS specified in section 5.10 is discussed in chapter 7. In Chapter 7, we use the rescue responder scenario described in section 2.3.2 with various configurations, to test a MAS modelled on the OJAzzIC meta-model. The ability of agents to coordinate and enact joint tasks is tested with shared stretcher rescue tasks that are not specified with detailed plans specifying agent role allocation and coordination at design time. Such tasks occur in EM because the domain is complex and dynamic involving uncertainty, so agent improvisation and adaptability is required at run time. The behaviour of agents in the demonstration MAS is evaluated with reference to the requirements.

Chapter 3

Background literature

3.1 Overview

In this chapter, existing work in the field of organisation based MAS is examined. It is analysed to attempt to identify knowledge and implementation gaps associated with modeling complex systems requiring dynamic coordination. First, in section 3.2, a brief overview outlines the coordination mechanisms adopted successfully in human organisations solving complex and dynamic problems. Then a number of key concepts regarding agents and organisations are introduced in section 3.3. Approaches used in meta-models for the functional representation of a problem are described as well as the constructs used within an organisation meta-model. A brief introduction to methodologies in the engineering and design of MAS is provided in section 3.4, before narrowing to organisation based meta-models.

In section 3.5, a high level analysis of organisation meta-models is provided with a comparison of meta-models in terms of the approaches taken to address flexibility. A number of existing MAS organisation based meta-models are then examined in more detail. The meta-models selected all provide features that address some of the requirements identified in the previous chapter. An analysis and overview of different approaches used for agent coordination, adaptation and flexibility in organisation meta-models is presented. The literature is then examined in terms of approaches used to define agent interactions particularly using policies and commitments in section 3.6. Mechanisms for establishing shared mental models between agents are discussed. The chapter concludes with a discussion of the gaps and limitations of existing meta-models in section 3.7.

3.2 Dealing with complexity in human organisations

In complex, dynamic domains, when people are involved in complex problem solving within an organisation, coordination is important (Smith and Dowell 2000). Complex organisational problem solving occurs in domains such as military command and control (J. Gorman, N. Cooke, and Winner 2006), Emergency Management (EM) (Smith and Dowell 2000) and naval navigation (E. Hutchins 1991). Investigations resolving accidents involving complex human decision making suggest that communication errors, incorrect assumptions or errors of omission in interactions can be significant causes of problems (Leveson 2004). In order to function successfully in complex situations, it has been suggested that humans improvise. In other words, people behave with flexibility and deliberately make rational choices to deviate from specified rules to establish and adopt practical operational behaviour that is successful (Leveson 2004). However, as a guide, the organisation provides a structure which organises and coordinates people and allocates responsibility in terms of how tasks are to be distributed within the organisation. Improvisation is not a replacement for procedures that are well designed for anticipated situations but it is an adaptive response when an unexpected situation occurs that was not anticipated (Trotter, Salmon, and Lenné 2013).

Organisations can take many forms. The organisation can also take on an identity beyond a set of individuals. Some researchers suggest that the group or system itself emerges with its own cognition with extended boundaries and different capabilities so that the organisation is more than an aggregate of coordinated individual members' behaviour (E. Hutchins 2000; G. Tidhar and E. A. Sonenberg 1999; Hollan, E. Hutchins, and Kirsh 2000). Generally, however, the focus of an organisation is to provide structure, coordination, definitions of roles and responsibilities and mechanisms for allocation of tasks. Organisations can be long term stable structures associated with external agencies or can be created in response to a short term objective. These short term organisations are, by definition, more flexible and reliant on initiative and mutual adjustment. Such ad hoc organisations have been called adhocracies. The adhocracy involves a network of multiple groups, decision-making in a rapidly changing environment (D. Mendonca, Jefferson, and Harrald 2007; Mintzberg 1980).

The way an organisation is structured depends upon the situation and purpose of the organisation. The structure of an organisation can have great impact on the performance and utility of that organisation (Mintzberg 1983). In the case of military command and control, decentralised command results in coordination emerging from multiple distributed locations. This situation involves localised decision making and it is not necessarily the case that one person is aware of the entire situation (J. Gorman,

N. Cooke, and Winner 2006). Similarly, in EM, responsibility and decision making is distributed and emerges from micro level decisions and interactions between members to develop the organisation at a macro level. Smith and Dowell suggest that the interactions between agencies involved in EM are very important so that the organisation as a whole shares situation awareness (Smith and Dowell 2000). As decision making is distributed, sometimes it is unclear where a decision originated. Plans evolve, sometimes beginning as incomplete, partial plans. Individuals need to coordinate in order to agree on plans and any improvisation necessary.

Efficient human and automated team functioning relies on good coordination (Chistoffersen and D. D. Woods 2002). One model shown to describe the cognitive processes involved in team collaboration in a complex, ill-structured, dynamic uncertain environment includes at least three aspects: individual knowledge creation; collaborative team problem solving; and team consensus (S. Hutchins, Bordetsky, T., and Bourakov 2007). Developing 'Common Ground' in effective teamwork involves more than sharing situation awareness regarding the problem.

Establishing a shared representation of a problem and maintaining common ground are foundations for cooperative work (Chistoffersen and D. D. Woods 2002). The problem itself in terms of goals, plans and situational awareness needs to be shared as well as a representation of the activities of other agents in the team (Chistoffersen and D. D. Woods 2002). If team membership and structure is dynamic, then a shared representation of who is currently involved may also be needed. A shared mental representation of a situation within the team is referred to as a Shared Mental Model (SMM) (Smith and Dowell 2000). Teams with SMMs perform better than teams without a SMM (Rouse, Cannon-Bowsers, and E. 1992). It is suggested that human team cognition is all about the interactions, not just the shared mental models present. The team cognition relates to how the team interacts to create and share information (N. J. Cooke, J. C. Gorman, Myers, and Duran 2013).

Coordination is a topic addressed by many organisational theorists. It is not our aim to survey the field of literature in this area. The established theory presented by Mintzberg (Mintzberg 1980; Mintzberg 1983; Mintzberg 1989) is adopted, as it is anchored in and is considered an authentic view of real world human organisational behaviour. Mintzberg has been an influential thinker in management and organisational science and his work has emphasised the importance of coordination. He suggests that the choice of an effective coordination and organisational model depends on the team size and the type of tasks being shared. He also identifies and describes six coordination mechanisms used by organisations:

1. mutual adjustment,
2. direct supervision,
3. standardisation of work processes,
4. standardisation of skills,
5. standardisation of outputs, and
6. standardisation of norms.

Mutual adjustment and direct supervision involve ad hoc coordination. Mutual adjustment refers to individuals adjusting their own plans to fit in with others and relies on informal communication between individuals. Mutual adjustment is used in simple situations when there is no need for a predetermined plan or standardised rules or policies. Mutual adjustment is also used in the most complex situations that rely on this coordination for success. Complex situations involve dynamic environments where predetermined plans and procedures cannot be relied upon without some adjustment. Direct supervision describes the coordination provided by a centralised leader who directs others.

Mintzberg's four standardisation mechanisms are based on predefined policies, procedures or norms within the organisation that ensure coordination. The standardised rules can control work processes, skills or outputs from tasks and behaviour of members so that coordination is achieved. These suit situations where the environment is predictable. Formal policies and procedures can exist within the organisation so that personnel are trained with specialised skills to standardise behaviour. For example, the Australian incident management system AFAC (AFAC 2004) provides detailed regulations and policies for the management of incidents in emergency response in Australia. These policies include specifications of organisation structures and inter-agency interaction during a crisis. Additionally, hierarchical structures can exist defining roles that specify authority and responsibilities and norms (Mintzberg's sixth coordination mechanism) specify expected behaviour and describe values and beliefs associated with the organisation.

Whilst Mintzberg has identified coordination mechanisms, they cannot be applied without some consideration of the complexity and dynamism of the environment. There are still major challenges in coordinating a group of people in a complex dynamic environment including:

1. having an efficient adaptive organisation structure in response to the dynamic environment (Comfort, Ko, and Zagorecki 2004; Entin 2001; Bigley and Roberts 2001)
2. ensuring that each member knows what is expected of them, or how to negotiate

this, toward reaching the organisation's goals (Bigley and Roberts 2001);

3. individuals being aware of others in the organisation and knowing what others are doing (Chistoffersen and D. D. Woods 2002; Smith and Dowell 2000; Schraagen and Ven 2011);
4. sharing situation awareness (M. Endsley, Bolte, and Jones 2003) or establishing common ground (Klein, P. J. Feltovich, J. M. Bradshaw, and D. D. Woods 2004);
5. enabling resource dependencies to be managed between individuals (Smith and Dowell 2000);
6. high information load detracting from human performance (Entin 2001; Omodei, McLennan, Cumming, Reynolds, Elliott, Birch, and Wearing 2005);

These challenges demand that people operate with an ability to adapt with appropriate interactions to support coordination in a dynamic context.

Team adaptation in the context of people within an organisation has been defined in terms of four processes in an adaptive cycle: situation assessment, plan formulation, plan execution and team learning (Burke, Stagl, Salas, Pierce, and Kendall 2006). The dynamic nature of the EM domain, requires that rescue personnel also apply a level of flexibility and adaptability in their behaviour (Bigley and Roberts 2001; Smith and Dowell 2000). It has been found that regardless of policies and norms, explicit plan formulation does not necessarily occur. Rather, decision makers in EM, for example, are more likely to move straight to a procedural plan execution phase following an initial situation assessment (Cohen-Hatton, Butler, and Honey 2015).

Organisational adaptation can also involve a restructuring of the formal structure. Bigley and Roberts (Bigley and Roberts 2001) conducted a study of the Incident Control System as employed by a fire agency in USA. They identified four basic processes for improving reliability and flexibility in organisational change: Structure Elaborating, Role Switching, Authority Migrating, and System Resetting. Structure elaborating refers to structuring the organisation to suit the situation demands, role switching refers to reallocating roles and role relationships, authority migrating refers to an allocation of or adoption of roles according to the expertise and capabilities of the individuals available (rather than based on rank or role in a hierarchy) and system resetting refers to the situation when a solution does not seem to be working and a decision is made to start with a new organisational structure.

In a command and control setting, during operation, organisations can also change between using predefined structures and organisational plans that match a situation and restructuring and also improvising in order to develop new solutions in order to cope with complications that are not anticipated (Bigley and Roberts 2001). Improvisation

at a system level is an emergent property that relies on appropriate communication, neither a strong nor weak specification of roles within the organisation and a good organisational memory (Trotter, Salmon, and Lenné 2013). The mutual adjustment (Mintzberg's first coordination mechanism) that occurs between individuals negotiating their responsibilities dynamically is crucial to success as "Sophisticated problem solvers facing extremely complicated situations must communicate informally if they are to accomplish their work" (Mintzberg 1983).

Regardless of domain or setting, to achieve flexibility in a dynamic and complex environment, detailed operational plans evolve based on high level plans rather than on scripts or detailed plans for coordination specified earlier. In this environment, plan revision, appropriate knowledge transfer and mutual adjustment to fit in with others and manage interdependencies are crucial to success. Improvisation is needed to adapt pre-existing plans, revise role descriptions and make do with existing resources in order to achieve a solution in a time critical situation. This improvisation to cope with non-routine events has been described as similar to the improvisation required of a jazz musician (W. A. Mendonca D. . W. 2007).

Improvisation, adaptation and mutual adjustment all rely upon the establishment of common ground and a shared situation awareness between people involved. As Tuomela (Tuomela 2006) discusses, maintaining mutual beliefs between individuals requires that each individual has beliefs about the beliefs of others in the team. When humans are working in complex environments, each actor is potentially simultaneously swapping between multiple contexts, each requiring decision-making (Fan, McNeese, B. Sun, Hanratty, Allender, and Yen 2010). Interaction and communication are important. The team cognition relates to how the team interacts to create and share information (N. J. Cooke, J. C. Gorman, Myers, and Duran 2013). In EM, due to the distributed nature of the system, one central shared artifact is not feasible. The organisations that make up the response system may include planned and ad hoc organisations. Collaboration and flexibility between the distributed adhocracies that form is central to effectiveness of the system (D. Mendonca, Jefferson, and Harrald 2007).

Communication is very important within an organisation and failures can be attributed to inappropriate interactions (Rouse, Cannon-Bowsers, and E. 1992; Schraagen and Ven 2011). As individuals are potentially engaged in multiple organisational groups, and swapping between contexts, it is necessary to establish context for interaction. *Coordination loops* were introduced as a unit of analysis to extend joint activity theory in the context of mixed human-robot team coordination (Voshell, D. D. Woods, Prue, and Fern 2007). Horizontal, vertical and projective loops describe the links be-

tween groups working together within a larger team. These links are important channels for communication. In these loops, team members adjust goal-oriented activity as situations change to achieve synchronisation and mutual adjustment of tasks (Voshell, D. D. Woods, Prue, and Fern 2007). Identifying potential loops in an organisational structure is a step toward enabling communication channels to be established and appropriate information sharing to occur (Prue, Voshell, D. Woods, Peffer, Tittle, and Elm 2008).

The examination of human behaviour during complex and dynamic problem solving has identified a number of issues of significance. These include the need to adapt and improvise in order to adjust plans to suit the context. Also, it is important that appropriate interactions occur in order to share relevant information about the situation and intentions of others. This relies on people having awareness of the situation and others. In the following sections, a number of organisation meta models that have been proposed in the literature for MAS are examined in terms of their ability to enable in agents some of the flexible behaviour demonstrated by humans operating in complex dynamic organisations.

3.3 Organisations, agents, meta-models, concepts

In this section, the main concepts used in modelling of MAS, particularly organisation based meta-models, are introduced. Then, the process of categorising meta-models based on features in their organisation structure is begun. In Section 3.5, more detailed analysis of selected models is presented. A comprehensive survey of organisational meta-models in MAS is not provided; rather the focus is on analysing selected meta-models that provide some features addressing flexibility and awareness. These meta-models are analysed with regard to their ability to enable flexible behaviour and coordinated improvisation by agents at run time. For more details regarding organisational meta-models in multi-agent systems and models/frameworks for social coordination the reader is directed to (V. Dignum 2009; Boissier, V. Dignum, and García 2016; Aldewereld, Boissier, V. Dignum, Noriega, and Padget 2016).

MAS have increasingly been used in dynamic situations to exploit the power of the autonomy and flexibility of agents. MAS research has included work on teamwork models, meta-models and architectures for agent coalitions, teams and organisations. One view for MAS is based around Agent Centred MAS (ACMAS) that focus on individual agents and mental states of each agent and communication between individuals. An opposing view considers MAS as Organisation Centred MAS (OCMAS) where the

organisation of agents is defined in terms of capabilities, constraints, roles, groups and tasks (Ferber, Gutknecht, and Michel 2004). In OCMAS, the agent behaviour including interactions between agents are modelled using organisational constructs. The organisation level specification describes what is to be achieved, but not how it is to be achieved. We focus on an analysis of OCMASs.

3.3.1 Definitions and concepts

In this section, some of the common concepts used in existing organisational MAS models are defined with some discussion regarding their use. A description is provided of how an organisation is used and defined, how an agent fits into an organisation and the use of roles and other organisational constructs. Additionally, differences that exist between meta-models are briefly highlighted including the distinction between a functional specification and an organisation or social specification. As agent awareness, autonomy and coordination are all important to the later discussion, it is useful to briefly examine approaches taken in some existing models to address these concepts.

A meta-model is a model of a model. In the context of MAS, a MAS system is a model, often a simulation model that represents some aspect of reality. The MAS model is designed based on a number of components including individual agents. A meta-model for a MAS describes the conceptual components and relationships between entities that are to be included in a specific MAS model. There have been attempts to create generic meta-models with the objective of standardising the software engineering of MAS. A generic meta-model attempts to include the main concepts needed in a meta-model. According to one of the first organisation oriented meta-models, the AGR (Agent/Group/Role) model (Ferber, Gutknecht, and Michel 2004), agents, groups and roles are defined as follows:

- An **Agent** is "... an active, communicating entity playing *roles* within *groups*. An agent may hold multiple roles, and may be member of several groups."
- A **Group** is defined as a set of agents sharing common characteristics and is used to create separate contexts to partition organisations.
- A **Role** is "an abstract representation of the functional position of an agent in a group ... and may be played by several agents".

In AGR, an agent must play a role in a group. In addition to specifying roles in AGR, interactions or social relationships are defined as part of the organisation centred design (Ferber, Gutknecht, and Michel 2004).

FAML (Beydoun, Low, Henderson-Sellers, Mouratidis, J.-J. Gomez-Sanz, Pavón,

and Gonzalez-Perez 2009) is a generic meta-model proposed for MAS development. In FAML, *roles* define the system and *agent definitions* relate agents to roles. The organisation structure defines multiple roles. Roles are responsible for tasks and system goals comprise tasks. Agents can adopt one or more roles (based on agent definition) and the role model includes specification of relationships between roles. An agent uses a **plan** that is defined as a set of actions that the agent can adopt in order to achieve a particular task outcome.

The concept of role in the agent organisation is very important. Roles are defined and then agents may play or enact a particular role. In AGR, the role is defined as a class that defines the normative behaviour of the agent (Odell, Van Dyke Parunak, and Fleischer 2003). Agents are associated with a role to determine the activities in which the agent may participate e.g. (Beydoun, Low, Henderson-Sellers, Mouratidis, J.-J. Gomez-Sanz, Pavón, and Gonzalez-Perez 2009; Jiang, V. Dignum, and Tan 2011; Odell, Nodine, and Levy 2005). The role describes a set of constraints such as obligations, responsibilities, capabilities and authority associated with the role. Also, role relationships are used to relate and specify social interactions between agents playing roles within each group of agents. Roles are commonly used in organisational meta-models in this way. The organisation objectives are commonly described implicitly in the system using roles. Effectively, if every role is fulfilled, the organisational objectives will be reached. Roles can describe the requirements for agent interaction and also effectively define the functional requirements of the system (Odell, Van Dyke Parunak, and Fleischer 2003). A similar use of roles occurs in OperA (V. Dignum 2004; Jiang, V. Dignum, and Tan 2011). A limitation of this approach is that agents may only participate as a *role enacting agent* within the constraints of the role and do not have any identity outside of the role specification.

Coutinho et al (Coutinho, Brand, J.S. Sichman, and Boissier 2008) describe four main dimensions in the modeling of MAS organisations: structural, functional, interactive and normative dimensions. It is generally accepted that an organisationally based MAS is defined in terms of an **organisational structure** and a dynamic component referring to the instantiation of that structure. The organisational structure is defined by the organisational meta-model adopted. It often consists of a set of static roles that are related. However, in some cases, there is some flexibility to choose roles that fit a situation at run time. The **instantiation** of the organisation includes the particular set of agents, roles and relations involved at a particular time. In many cases, the organisation refers to the *organisation structure* and the agents populating the structure may or may not be considered part of the organisation. There is a view that the organisation itself is an artifact and that agents' behaviour is specified using norms or

policies associated with roles within the organisation (Dastani, Grossi, J. Meyer, and Tinnemeier 2009). An opposing view would suggest that the organisation also *includes* the members populating the organisation.

The organisation structure may be referred to as the organisation meta-model. The organisation meta-model specifies the constructs and entities that are used to represent:

- the functions or objectives of the organisation,
- the way agents are organised socially and
- how agents are allocated to or adopt responsibilities.

. The way that the structure of the organisation is mapped to actual agents upon creation or instantiation of an organisation varies across models as does the mapping between the structure to a functional or goal decomposition of the domain problem.

Some meta-models have the organisational structure specified at design time so that it cannot change. In such cases, the set of roles available to be enacted by agents at run time is pre-defined statically at design time. Agents can be allocated or form contracts at run time to specify the social model for the organisation (e.g. (Weigand, V. Dignum, J.-J. Meyer, and F. Dignum 2003)). In more flexible approaches, based on the particular context, roles may be dynamically selected or assigned to agents at run time (e.g. (S. A. DeLoach, Oyenan, and Matson 2008)). In these cases, the organisational structure is derived to suit the situation.

It has been argued that the structural dimension of an organisation should be defined separately and independently of the functional dimension so that agents can effectively achieve the organisation purpose and also reason about each dimension separately (J. Hübner, J. Sichman, and Boissier 2002). This approach is adopted in MOISE+ (J. Hübner, J. Sichman, and Boissier 2002; J.F. Hübner, J.S. Sichman, and Boissier 2007), MOISE (Hannoun, Boissier, J. Sichman, and Sayettat 2000) and MOISE*inst* (Gâteau, Khadraoui, Dubois, and Boissier 2005) to define the functioning of system using missions, separate from agent allocations. An organisational structure can include sub-organisations within it or may at the base level involve only agents as entities (Ferber, Gutknecht, and Michel 2004).

According to AGR, each group within an organisation defines a context for partitioning agents and agents within a group may interact freely. Each group structure is hidden to agents outside that group. Regardless of the structure of the organisation, it is important that agents are aware of the organisation or group context if it is necessary that agents consider this in their reasoning. The social context of an organisation or

coordinating group of agents has been given various names including as a Sphere of Commitment (M. P. Singh 1999), or Coordination Loops (Prue, Voshell, D. Woods, Peffer, Tittle, and Elm 2008) or an Organisational Adhocracy (D. Mendonca, Jefferson, and Harrald 2007). A separate approach is to use Petri Net models to monitor and coordinate hierarchical team plans (Bonnet-Torrès and Tessier 2009).

Ferber (Ferber, Gutknecht, and Michel 2004) suggests that the organisational level imposes an abstract structure that describes agent activities at a functional level, i.e. a breakdown of tasks to be completed. The organisation specification is at the level of norms, laws or expectations that agents must be aware of, but not on how agents must implement tasks or responsibilities. Furthermore, Ferber argues that the organisational level structure does not specify any mental issues (such as beliefs, desires, intentions, goals etc.) but purely defines expected behaviour (Ferber, Gutknecht, and Michel 2004). The focus is on ensuring tasks are completed functionally, so this achieves coordination by defining outputs of each task. The functional specification does not specify how agents will reach the goal state. This approach is similar to Mintzberg's (Mintzberg 1980) classification of coordination by a standardisation of outputs.

If the functional breakdown is mapped to roles within the organisation, there is an assumption that a task level decomposition of the organisational objectives can be matched one-to-one against a set of roles. The roles in turn then are assumed to be able to match to the agents participating in the organisation. The role descriptors provide coordination by setting protocols and role based behaviour, similar to Mintzberg's classification of standardisation of skills. This approach does not rely on agents developing an awareness at the organisational level. Agents only need to fulfil their role.

A similar abstraction between roles and goals has been promoted in the KB-ORG system, designed for automatic allocation of tasks to agents in a dynamic organisation (Sims, Corkill, and Lesser 2008). KB-ORG is an externally controlled organisation design tool in which agents do not need to be organisation-aware, the system allocates agents to roles. In KB-ORG, roles contain an assignable list of responsibilities and if necessary, roles can be split between a set of agents. Roles are also distinguished as either application level roles or coordination roles. Coordination roles are created when an application level role is split between a set of agents.

Functional specification and decomposition of tasks in MAS to specify lower level tasks with synchronisation or coordination relations is not unique. It is found in models including for example STEAM (Tambe 1997) and TAEMS (Lesser et al. 2004). This approach has also been adopted in AGR (Ferber, Gutknecht, and Michel 2004) and the MOISE system (Gâteau, Khadraoui, Dubois, and Boissier 2005). MOISE-

Inst (J. Hübner, Kitio, and Ricci 2010) provides flexibility by abstracting objectives into missions that encompass multiple tasks. In the MOISE system (Gâteau, Khadraoui, Dubois, and Boissier 2005) a distinction is made between a separate structural specification and a functional specification. The functional specification provides syntax for representing goals in terms of sequence, choice and parallelism.

The interactive dimension specifies dynamic social relationships and interactions between agents in a MAS. These interaction structures are usually described in terms of roles within the organisation and can be specified in terms of interaction policies or protocols defined at design time (Baldoni, Baroglio, Capuzzimati, and Micalizio 2018a) or can be described in terms of more dynamic mechanisms such as agents explicitly forming dynamic commitments (Meneguzzi, Magnaguagno, and M. Singh 2018a; Torroni, Yolum, M. P. Singh, Alberti, Chesani, Gavanelli, Lamma, and Mello 2009; J.F. Hübner, J.S. Sichman, and Boissier 2007; Meneguzzi, Magnaguagno, and M. Singh 2018b), social contracts (V. Dignum, J.-J. Meyer, and Weigand 2002), or SharedPlans (B. Grosz and Kraus 1999). **Interactions** are the direct or indirect involvement between entities within the system. Interactions occur either directly as messages between agents following an interaction policy or messaging protocol, or indirectly guided by role specifications or external interaction artifacts.

Approaching the organisation viewed as a social network involves considering the social mental attitudes such as knowledge about the structure of the organisation. Tidhar suggests that this relational knowledge separates a team oriented system from an organisation oriented system (G. Tidhar 1999). This approach differs from AGR which states that organisations do not have any mental attitudes but just describe a structure (Ferber, Gutknecht, and Michel 2004). Modelling social knowledge within the organisation is also important in OperA (V. Dignum and F. Dignum 2012). In OperA+ (Jiang, V. Dignum, and Tan 2011), organisations are viewed as abstractions in a hierarchy so the organisation itself can be treated as if it were an agent in a higher level organisation. Similarly, in ROMAS (García, Valero, and A. Giret 2016), the organisation forms a context for interactions. Organisations can contain other organisations and organisation roles can be played by an agent or an organisation. Interaction protocols, norms and contracts are used as the means to formalize the relationships between roles, organisations and agents (Aldewereld, Boissier, V. Dignum, Noriega, and Padget 2016)

A **Policy** defines a rule specifying behaviour that is expected from agents (or organisations) in a given system. In many cases, policies are strict forms of guidance that must be adopted, however, in some models policies can be more loosely inter-

preted and applied 'when possible' (S. A. DeLoach 2009). Design time policies can be specified to control agent run time behaviour to create and enact commitments (e.g. O-MaSE (S. A. DeLoach 2014), FAML (Beydoun, Low, Henderson-Sellers, Mouratidis, J.-J. Gomez-Sanz, Pavón, and Gonzalez-Perez 2009)).

In many meta-models, agent behaviour is restricted by the role(s) adopted by agents (e.g. OperA+ (V. Dignum and F. Dignum 2012)). **Norms** are associated with roles and are used to define behavioural expectations on agents in an organisaion. Agents can choose to follow norms or not however, sanctions can be applied if agents do not adhere to the expected norms e.g. (J.F. Hübner, J.S. Sichman, and Boissier 2007). There is a considerable body of valuable research work over an extended period of time regarding norms and the use of norms to specify agent behaviour, monitor compliance and issue sanctions to agents within an organisation context. For more background see (F. Dignum 1999; Vázquez-Salceda, V. Dignum, and F. Dignum 2005; Dastani, Grossi, J. Meyer, and Tinnemeier 2009).

Another approach to the design of a MAS is to specify individual agent types to describe the system requirements (T. Miller, Lu, Sterling, Beydoun, and Taveter 2014). Odell and colleagues proposed a generic agent meta-model incorporating the notion of roles describing norms and contexts for agent classification and relational links between agents in a group context. In Odell's meta-model, agents may belong to groups that could be agentified or not (Odell, Nodine, and Levy 2005). An advantage of agentifying a team is that it enables the team to be treated as an agent in a higher level organisation. The agentified organisation can be allocated roles and interacted with directly as one entity, rather than needing to interact with members of the group as individuals. An advantage of an organisation being a first class entity is that this enables reasoning at an organisational level (G. Tidhar 1999). The organisation can exist independently from any individuals who are within the organisation at any point in time. The team itself has mental attitudes and if a team holds a belief, then an agent or sub-team of the team also holds that same belief.

Enabling a consistent interface to a group is beneficial toward addressing our requirements for flexible dynamic team membership, so using organisations as first class entities offers benefits. DeLoach and colleagues also identify agentified organisations as first class entities in OMACS (S. A. DeLoach 2009). It may not be possible to have the organisation itself possess a mental state, however, it is possible for agents within the organisation to maintain mutual organisational beliefs. In this way, the organisation exists as more than just a static structure, but as a set of interacting agents with shared common ground.

Organisation aware agents describe agents with some awareness of the organisational structure and the role/s and responsibilities of themselves as an agent. Such agents are capable of reasoning at an organisational level (A. Jensen, V. Dignum, and Villadesn 2014), An *organisationally adept* agent has been defined as one that adjusts its behaviour for operational decision-making in keeping with known organisational guidelines. Organisationally adept agents are organisation aware agents that are also capable of analysing of how well they are fulfilling organisational objectives (Corkill, Durfee, Lesser, Zafar, and C. Zhang 2011).

Within an organisation, mechanisms for agents being aware of the current organisational structure and their individual responsibilities include using artifacts (e.g. (Baldoni, Baroglio, Capuzzimati, and Micalizio 2018a; J. Hübner, Kitio, and Ricci 2010)), giving agents explicit awareness of the organisational structure as part of shared organisation beliefs, or by using commitments and conventions (N. R. Jennings 1993) to establish mutual beliefs (Tuomela 2006). **Artifacts** are external to the agents and can be shared amongst agents and used as a tool for sharing knowledge.

Grossi (Grossi and F. Dignum 2009) describes control as an important dimension of an organisational structure. **Control** relates to task execution and the supervision that ensures that a task is completed. This includes (re)assignment of agents to tasks or in the case of a system in which agents autonomously adopt tasks, it refers to agents knowing what tasks to adopt.

Multi-agent organisations may be created such that they are exogenous or endogenous organisations. Exogenous organisations are administered externally in separate systems to that of the agents who participate. Endogenous organisations are organisations within which agents are internally designed and implemented using the social organisational constructs within the agent system (Dastani 2014). Endogenous agent organisations are of particular interest in the current work, because agents in such organisations can be given the ability to reason with awareness of the organisation and others in the organisation. Agents need to know when to act, and the status of related dependent tasks, so a mechanism to control flow of information relevant to an agent's task is important (Grossi and F. Dignum 2009).

Existing meta-models for organisational MAS do not all incorporate agent awareness - of the organisational structure and/or responsibilities of agents. Some models rely on the organisational structure being imposed or designed externally, some models have a middleware layer that is responsible for creating the organisational design (e.g. (Gâteau, Khadraoui, Dubois, and Boissier 2005)) and allocating agents to roles. One approach for assignment of agents is to use a gatekeeper agent to allocate or autho-

rise agents to specific roles (e.g. (V. Dignum 2004; Ferber and Gutknecht 1998; Ferber, Gutknecht, and Michel 2004)). When the agents only have knowledge of their role and associated responsibilities and not the entire organisation, the coordination relies on the role descriptions. This is similar to Mintzberg's description of coordination by standardisation of skills (Mintzberg 1980).

As has been demonstrated, there are a number of design decisions regarding a meta-model that impact on the behaviour and flexibility of agents in a MAS built based on that meta-model. A more extensive survey of research regarding organisational models is provided in V. Dignum 2009. Important to flexibility of agent behaviour at run time is having a mechanism for coordination and agent awareness so that agents can reason at run time to adjust or adapt if a situation does not exactly align with the design time specification. In this section, a number of organisation based MAS concepts have been introduced and some approaches to functional specification and organisational specification briefly discussed. It can be concluded that flexibility can be helped by carefully choosing a coordination mechanism. The following ideas should be considered :

- keeping the functional (task) model separate from the organisation structure in the social (roles) model
- separating the functional activity in terms of task outcomes from procedural plans describing how agents achieve tasks
- having multiple possible functional pathways that can lead to a solution based on context
- having a flexible control mechanism for how individual agents adopt (or are allocated) responsibilities
- adopting organisations as first class entities
- agents having an awareness of the organisation structure, including roles and having an ability to reason about this
- agents having an awareness of others and an explicit interaction model to enable establishment of common ground between agents

In the next section, we broaded the discussion to examine methodologies in the engineering and design of MAS. Then in subsequent sections, exising organisation meta-models and approaches are examined in more detail.

3.4 Software Engineering methodologies for MAS design

A number of methodologies for agent-oriented MAS engineering and design have been proposed. These include O-MaSE(S. A. DeLoach and Garcia-Ojeda 2010; S. A. DeLoach 2014), Tropos (Bresciani, Giorgini, Giunchiglia, Mylopoulos, and Perini 2004), Gaia (Zambonelli, M. Jennings, and M. 2003), Prometheus (Padgham and Winikoff 2004), INGENIAS (Pavon, J. Gomez-Sanz, and Fuentes 2005), GORMAS (Argente, Botti, and Julian 2011; García, Argente, and A. Giret 2010) and ASPECS (Cossentino, Hilaire, Gaud, Galland, and Koukam 2014). A methodology considers the life cycle of development, including analysis, modelling, design and implementation. When designing and implementing a multi-agent system (MAS), generally the process includes adopting a conceptual framework, developing a platform independent design, detailed design then creating the system (Sterling and Taveter 2009). Agent oriented software engineering is based around the design of systems using agents and key characteristics and abstractions relating to agents. The software engineering process, particularly the analysis and design process is coupled with the abstractions and models adopted. A complete survey of work in Agent Oriented Software Engineering (AOSE) is not provided. The importance of software engineering, design methodologies and the relevance of model driven approaches to the design of OCMAS is discussed in this section. An overview of AOSE more broadly can be found in (Sterling and Taveter 2009; Dam and Winikoff 2013; Sturm and Shehory 2014; Abdalla and Mishra 2018). The reader is also directed to the EMAS series of workshops on Engineering Multi-Agent Systems published in springer link (see for example (Dalpiaz, Dix, and Riemskijk 2014; Cossentino, Hilaire, Molesini, and Seidita 2014; Weyns, Mascardi, and Ricci 2019b)). The development of multiagent based systems can involve multiple dimensions, including modeling the environment as well as using organisational models as a layer of abstraction above and separate from agents. Recently, attention has been given to merging multiple dimensions of development into a single paradigm aiming toward the development of complex distributed systems (Weyns, Mascardi, and Ricci 2019a).

In addition to methodologies for AOSE, software engineering methodologies particular to OCMAS have been proposed. As the focus in this thesis is on the development of an adaptive organisational meta-model, in the remainder of this section, a brief discussion of the design process more broadly is presented, then a more detailed overview is presented for two selected design methodologies specifically proposed for OCMAS: O-MaSE (S. A. DeLoach and Garcia-Ojeda 2010; S. A. DeLoach 2014) and GORMAS (Argente, Botti, and Julian 2011; García, Argente, and A. Giret 2010).

O-MaSE is a generic customisable design methodology proposed for the software

engineering of adaptive organisation based MAS. The O-MaSE methodology is designed so that it may be applied to any meta-model. Similar high level methodologies have been outlined to describe the design process with regard to TDF-T teams (Evertsz, Thangarajah, and Papasimeon 2017), team based extension to TDF (based on Prometheus methodology (Padgham, Thangarajah, and Winikoff 2014)).

GORMAS is a methodology for designing open MAS that are organisation based. O-MaSE is relevant to the work in the thesis because it is a methodology that can be adapted for any organisation meta-model. GORMAS is a more generic methodology that considers a broader perspective of the design of open agent systems. There are ongoing attempts from various groups to define generic processes and methodologies for the conceptual modelling of agent teams and organisations independent of the meta-model, software engineering approach or implementation platform chosen, for example, the TDF-T methodology for modelling agent teams (Evertsz, Thangarajah, and Papasimeon 2017).

3.4.1 The design process

As part of the design process, it is important to maintain awareness of the need for adaptation in the final system. Resilience engineering refers to the discipline of designing systems so that they have some capacity to adapt (D. Woods and Branlat 2011). Ultimately, the aim is to define systems that might have some adaptive capacity and have some built in resilience to cope with complex and dynamic coordination. A design based on fixed roles, plans or scripts with no capacity to adapt is more brittle than a design that enables some flexibility and ad hoc behaviour.

The design of a system should not conflict with the need for adaptation but facilitate behavioural improvisation or structural adaptation (Naikar and Elix 2016). The design process also must consider mechanisms for the regulation of agent behaviour at run time. One approach to control agent behaviour is to specify norms (García, Miles, Luck, and Giret 2015). This is very useful when agents are operating in an open environment. Norms can specify the regulations from an institutional perspective. and are used to specify agent obligations with respect to the organisation as an external entity or institution. In the current work, we are interested to specify agent obligations with regard to internal obligations within an organisation.

It is necessary to consider interactions between agents as part of the MAS design process and what level of specification is needed for agent interaction. It is possible to articulate agent interactions as requirements and model these separately in the design process for an agent system (T. Miller, Lu, Sterling, Beydoun, and Taveter 2014; Rah-

wan, Juan, and Sterling 2006). Agent interactions are not limited to occur only within the MAS or organisation but include interactions with other artifacts and resources in the environment. The BRAHMS environment (Sierhuis, Clancey, and Hoof 2006; Sierhuis 2001) provides a vehicle for modelling work flow, interactions and behaviour of agents at an individual and group level as well as interactions with systems, artifacts and the environment. Sierhuis and colleagues argue that it is important to have a platform for implementation and modelling that enables agents to enter an organisation and learn about an organisation structure in order to become organisation-aware (Sierhuis, C. Jonker, B. v. Riemsdijk, and Hindriks 2009). The BRAHMS system has been successfully used with MOISE+ to create such a simulation platform (Sierhuis, C. Jonker, B. v. Riemsdijk, and Hindriks 2009). Organisation-aware agents can potentially act with awareness to choose roles, missions and norms to commit to. Agents can also create new organisations at run time.

Others have investigated mechanisms for the design of ad hoc teams containing members who coordinate without communication and rely on observation of their team mates, for example, see the work of Stone and Agmon (Agmon and Stone 2012). This observational approach is applicable when the team does not have a shared strategy and individual agents have different knowledge and capabilities. Related work has also investigated the development of multi-agent plans that provide implicit coordination and encourage agent initiative (Engesser, Bolander, Mattmüller, and Nebel 2015).

Dam and Winikoff (Dam and Winikoff 2013) provide an overview of a number of agent-oriented software engineering methodologies and highlight similarities between these. In particular, the adoption of use case scenarios and adopting models for goals and entities such as roles and capabilities. Cossentino and colleagues argue that it is important, especially for complex systems, to combine an organisational approach with a holonic view (Cossentino, Hilaire, Gaud, Galland, and Koukam 2014). The holonic view considers that agents may be holons: composed of agents. The OMNI framework also takes an approach to include both agent based and organisation based perspectives during the design process (Vázquez-Salceda, V. Dignum, and F. Dignum 2005). OMNI includes 3 dimensions to be modelled: normative, organisational and ontological. As Garcia discusses, not all models consider specification from a normative perspective (García, Miles, Luck, and Giret 2015). Our interest is in the specification of the organisation dimension.

In the design of an organisation based system, the methodology would also include identification of organisation based entities. Dignum and Jiang propose methodological design guidelines (Jiang, V. Dignum, and Tan 2011) including Role Identification (in-

cluding mapping roles to objectives and identifying dependencies), Organisation Identification and Capabilities Identification used to match roles to role-enacting agents or organisations.

If the MAS is to be deployed as a part of an open environment, further considerations are necessary. Integrated MAS can be constructed for open and distributed situations utilising multiple services to create an open complex software system based on a Service Oriented Architecture (SOA). Agents can be used as a component to control and coordinate various distributed services and applications. The SOA is an integrated system architecture where distributed systems and resources are given some interoperability (Tapia, Rodríguez, Bajo, and Chorchado 2009; Dustdar and F. Li 2011). In SOA architectures, the functionalities performed by agents are modelled as services. Similar to OCMAS, there is a need in these architectures to model agent interactions. In FUSION@ (Tapia, Rodríguez, Bajo, and Chorchado 2009), a SOA-based Multi agent architecture, special agents are defined to perform coordination and supervisory functions (e.g. supervisor agent, directory agent, communications).

A more recent trend in SE methodologies for MAS design is the use of model driven engineering of MAS (Silva 2015) or adopting a model driven architecture (Cossentino, Hilaire, Gaud, Galland, and Koukam 2014). This involves consideration of models and meta-models as artifacts used in the software engineering of systems. Different meta-models can be adopted to describe different domains or components of the overall system (Cossentino, Hilaire, Gaud, Galland, and Koukam 2014). Schneider and Miller have developed a meta-model for the design, development and specification of human-agent teams in complex systems (Schneider, M. E. Miller, and McGuirl 2019). This meta-model uses the abstractions of roles and responsibilities to separate agent allocations from goal states to be achieved. Each agent adopts responsibilities that contribute to roles.

In the domain of disaster management and emergency response, the ALIVE project in Europe attempted to use the model driven engineering approach to create a new layered abstract model and create a system based on existing systems (Quillinan, Aldewerald, F. Dignum, V. Dignum, Penserini, and Wijngaards 2009; Nieves, Padget, Vasconcelos, Staikopoulos, Cliffe, F. Dignum, Vàzquez-Salceda, Clarke, and Reed 2011). The ALIVE framework defines 4 separate layers, each at a different level of abstraction - a web-services layer at the bottom, a service layer, a coordination layer, and then an organisation layer at the top level of abstraction. This comprehensive approach includes the provision to use existing services in a service oriented architecture. Considering the need for flexibility and adaptability in the final solution, designing a solution at

different layers of abstraction allows for some autonomy and independence within each layer of abstraction.

Coutinho et al (Coutinho, Brand, J.S. Sichman, and Boissier 2008) describe a tool and process for describing and mapping between different organisational meta-models so that an integrated model view can be created. More recently, researchers are taking a broader systems view and investigating software engineering approaches for the development of self-adaptive systems (Weyns 2017), scalable agent organisations (Schatten, Ševa, and Tomičić 2016) and open MAS (Uez and J.F. Hübner 2017).

As part of conceptual modelling for a MAS, an organisation meta-model can be adopted to specify the entities that must be included and specified in the design and final implementation of an organisation based MAS. Fornara (Fornara and Balke-Visser 2018) suggests that when choosing an organisation model for a MAS, the following questions should be considered :

> "Can the goals and tasks be divided into independent, formalised and standardised tasks?
> And if so, how to approach this best?
> Which of the tasks and sub-tasks have dependencies that need considering?
> Can tasks be grouped together and what are good means to group tasks (function, geographical location, client, process, etc.)?
> At what level have decisions to be made and controls to be set up?
> What kind of environment is the organisation located in (open, closed, static, dynamic)?
> What is the line of reporting in the organisation? Who has authority and what is the chain of command?
> What rules and formal processes are being required in the organisation?
> What level of predictability is the organisation to have? "
> (Page 581, (Fornara and Balke-Visser 2018))

A number of attempts have been made to use existing meta-models and combine them into a more generic model including a methodology for design of systems. Also, some methodologies are designed to be independent of underlying meta-models. The authors of JaCaMo+ for example, discuss the benefits of creating a MAS system based on separate specification of agents, environment, interaction and organisation (Baldoni, Baroglio, Capuzzimati, and Micalizio 2018a). JaCaMo+ is a framework that encompasses a number of existing meta-models as artifacts each addressing aspects of the overall system. JaCaMo+ integrates separate components for agent programming, environment programming, interactions between agents and organisation programming into one framework. The agent interactions are designed using a first class artifact containing explicit commitments between agents. Coordination is achieved dynamically by agents creating commitments. The commitment based interaction protocols specified

at design time align roles within the organisation to interaction protocols, however they do not specify the ordering of interactions between agents. This gives agents flexibility at run time to select interactions to suit the situation.

In order to achieve flexibility in run time behaviour, we cannot over-specify detailed plans at design time for how agents should achieve goals at run time without the risk of being too specific and thus exclude the applicability of the plans to the context arising at run time. However, using the organisational construct has been demonstrated to provide constraints and guidance to agents' behaviour at run time. Flexibility at run time can be controlled by

- providing alternative organisation structures at design time (e.g. OMACS (S. A. DeLoach 2009))
- allowing agents to dynamically allocate/adopt organisational roles at run time. (e.g. OMACS (S. A. DeLoach 2009), MOISE+ (J.F. Hübner, J.S. Sichman, and Boissier 2007)). Contract based systems such as OperA require that agents explicitly create run time contracts to specify the allocation of agents to roles (Weigand, V. Dignum, J.-J. Meyer, and F. Dignum 2003).
- flexibly and independently associating roles with the functional decomposition of a problem so that it is possible to match these dynamically

Where possible deferring resolution of specific operational decisions until run time affords agents more flexibility to adapt to find a solution based on agents and resources available at run time. The design of the coordination mechanisms to be used by the MAS and how these are specified at design time and what level of agent control and awareness exists regarding coordination at run time is also important to consider as part of the design process.

When agents need to work together at run time and coordinate their behaviour, it is necessary to ensure that there are mechanisms in place for this coordination. Coordination can be implicitly built into the scripts that encode plans for agents internally describing how to achieve a goal or coordination can be explicitly managed - by a special coordinator agent or by agents themselves creating commitments with other agents explicitly at run time.

We list six different categories in terms of the dynamics of coordination:

1. fully scripted coordination built into agent internal plans for how to achieve a goal state
2. fully scripted coordination built into agent internal plans, but multiple possible paths available to select from

3. partially scripted coordination in high level plans based on roles that agents might adopt at run time
4. aware agents can reason at their organisation at run time and create commitments with other agents toward an agreed plan
5. central manager or coordinator agent organises other agents and allocates them to tasks that will be coordinated with others
6. external to the organisation construct, coordination scripts or plans are created and imposed on the agents

The most autonomy and flexibility is given to agents with categories 3 and 4, however these also demand that agents have awareness and an ability to reason about their organisation structure and perhaps negotiate roles with other agents. Picard (Picard, H'ubner, Boissier, and Gleizes 2009) describes different levels of awareness in MAS and with particular reference to the potential reorganisation or adaptation of an organisational structure. Picard describes the reorganisation within an organisation centered multi agent system in which agents have an awareness of the organisational structure as follows: (i) change of the definition of the organisation itself and (ii) change of the allocation of roles to the agents, i.e. the way the multi-agent organisation itself is built or instantiated with agents. The redesign of an organisation can be performed by a designer or can be a result of agents dynamically changing the structure.

If we are guided by the behaviour of people, we could consider the possibility that agents need not formally restructure their organisation but could be given license to behave using the structure as a guide, but to improvise at run time in the implementation. This use of organisations is of interest to the current authors. Agents can work with top down organisational design that suggests roles that agents can adopt, however, if at run time, there is not an agent available with exactly the right match to adopt a role, agents can improvise.

Having identified some of the design considerations and generally how the process of SE is undertaken for MAS design, two methodologies are outlined in the remainder of this section. In Section 3.5, we examine particular meta-models for agent organisations.

3.4.2 O-MaSE

In this section, an overview of the O-MaSE methodology is presented as an example of a generic methodology describing the necessary processes in designing an organisation based MAS (S. A. DeLoach and Garcia-Ojeda 2010). O-MaSE provides a set of agent development tools and a customisable approach to building complex, adaptive MAS. Each task in the methodology produces a component or entity that contributes to the

eventual system.

Key tasks in the O-MaSE methodology are listed in Figure 3.1.

O-MaSE task	**Component produced**
Model goals	goal decomposition tree
Refine goals	task specification - synchronisation, concurrency, dependencies
Model domain	domain level tasks and interfaces
Model organisation interfaces	organisation models and inter-organisational interfaces
Model roles	role model (relationships and interactions)
Define roles	role descriptions (capabilities required) and responsibilities
Define role goals	role-goal responsibilities
Model agent classes	agent types, (capabilities possessed), relationships model
Model protocols	autonomy and control protocols
Model policies	social policies
Model plans	plan design
Model capabilities	agent capabilities model
Model actions	action rules model
Generate code	code

Figure 3.1: O-MaSE key tasks applied to produce design components

In O-MaSE (S. A. DeLoach and Garcia-Ojeda 2010), goals are used to define the objectives of the organisation, whilst roles are used to define positions within the organisation that can achieve a given goal or set of goals. A limitation of O-MaSE is that there is no provision for defining a system capable of splitting roles or sharing role responsibilities amongst a number of agents.

3.4.3 GORMAS (Guidelines for Organisational Multi-agent Systems)

The GORMAS methodology is very generic and high level for application to the design of open MAS incorporating many virtual organisations. It is not specific to a particular organisation meta-model. Activities in the design process using GORMAS (Argente, Botti, and Julian 2011; Esparcia, Argente, Vicente, and Botti 2014) are generalised, based around designing an OCMAS for an open MAS comprising of multiple Virtual Organisations.

The main activities include

- mission analysis;
- a system requirements analysis stage and a service analysis stage during which the services offered by an organisation are identified;
- organisation design; and

- organisation dynamics design

. In GORMAS, agents are assumed to be external to the organisation. The methodology involves more than specification of an organisation structure meta-model. The meta-model for organisations is a virtual model with five organisational dimensions: structural, functional, dynamical, environment and normative.

The various stages in the design process identify goals, services, requirements, how tasks are achieved, how goals relate to services and roles. At the organisation level, the design includes how groups are formed and how tasks are divided as well as control, coordination and standardisation using norms. The social dynamics within an organisation are also considered and specified in terms of policies for role adoption, control and decision making.

GORMAS is more general than the specific design process used in O-MaSE for creating a single MAS, however, it has clear overlap with O-MaSE in terms of specifying the need to model familiar entities such as goals, tasks, agents, roles, norms and policies. It also specifies explicitly in the design process, a selection of control and coordination mechanisms. These are important in the context of the current work. A number of existing organisational meta-models are presented and evaluated in the following sections.

3.5 Organisational meta-models in MAS

In this section, existing organisation meta-models are examined to compare and contrast these models in terms of flexibility, adaptation, coordination and improvisation. Each meta-model proposes an abstract model for the organisation and how agents operate within the organisation.

3.5.1 Categorising organisational meta-models

In this section, organisational meta-models are compared in terms of their flexibility and organisational adaptability, particularly how they approach coordination.

Determining an appropriate organisational MAS design for any given scenario is an open research problem, with some taking an empirical approach (Franco and J.S. Sichman 2014) and others attempting to define generic meta-models or frameworks by combining existing models e.g. JaCaMo (Boissier, R. H. Bordini, J. Hübner, and Ricci 2014) and FAML (Beydoun, Low, Henderson-Sellers, Mouratidis, J.-J. Gomez-Sanz, Pavón, and Gonzalez-Perez 2009). FAML is a generic model and does not attempt to

address the specific requirements of adaptability and flexibility however does have some features of interest. In this section, we select a number of meta-models for comparison that include features offering flexibility or adaptation.

Organisation centred meta-models that provide for some level of organisational adaptability include OperA+ (Jiang, V. Dignum, and Tan 2011) and OMACS (S. A. DeLoach 2009). Flexibility in terms of role adoption is addressed within OperA by including capabilities in role specifications and using a gate-keeper agent to allocate roles dynamically (Aldewereld, V. Dignum, C. M. Jonker, and M. B. v. Riemsdijk 2012). In this case, the gate-keeper agent selects an agent to play a role based on the agent matching the required capabilities. If an agent does not possess all the required capabilities, the role is not assigned. Similarly, OMACS achieves flexibility enabling goals and agent roles to be dynamically matched using capabilities.

A social model that describes agent interactions is made explicit in both JaCaMo+ and in OperA+. In OperA+ this model is described using contracts in which agents commit to enacting a particular role. In JaCaMo+, the interaction between agents is modelled using explicit contracts and modelled based on interaction policies. The social state containing the explicit commitments created is stored in a separate artifact, however agents cannot view that social state other than through role actions. This may limit agent awareness to rely on pre-scripted interactions built into role actions.

In a model that relies on coordination by standardisation of tasks, outputs or skills, agents need to know the task they are responsible for and the actions to perform in order to complete the task. Agents need not have awareness of others or how their behaviour might impact on others. If however, an agent is going to coordinate more dynamically by mutual adjustment, the agent needs to know who else may be impacted by decisions. The agent must deliberate with awareness of the organisational objectives, structure and current state. In order to foster awareness, a mechanism for sharing the appropriate knowledge is required.

The mechanisms used to foster awareness and share information between agents can also used to compare and distinguish between approaches.

On an organisation level, models can be separated based on how adaptable the organisation structure is and if the organisation is capable of restructuring at run time. If organisation structural change is possible, are agents responsible for this or is it controlled externally? Is reorganisation a structural change to the organisation or a change of allocations of agents to roles or responsibilities?

Figures 3.2 and 3.3 position a number of organisation meta-models in terms of

Coordination Mechanism					
Meta-model	**Design time coordination in plans**	**Role based coordination**	**Tasks/ goals defined independent of roles**	**Roles define agent capabilities**	**Agents adjust plans at run time**
SharedPlans (B.J. Grosz and Sidner 1990)					✓
KBORG (Sims, Corkill, and Lesser 2008)	✓				
JaCaMo (Boissier, R. H. Bordini, J. Hübner, and Ricci 2014)		✓	✓	✓	
JaCaMo+ (Baldoni, Baroglio, Capuzzimati, and Micalizio 2018a)		✓	✓	✓	
OperA (V. Dignum 2004), OperA+ (Jiang, V. Dignum, and Tan 2011)	✓	✓	✓	✓	
OMACS (S. A. DeLoach 2009)	✓	✓	✓		
MOISEinst (Gâteau, Khadraoui, Dubois, and Boissier 2005)		✓	✓	✓	

Figure 3.2: Positioning a selection of existing meta-models for coordination

the approaches used in each. The coordination approaches are categorised separately dependent upon how coordination is specified and how much flexibility is afforded to agents. The coordination can be specified in plans written at design time or using roles that define processes. Other points of distinction between models shown in Figure 3.3 include whether functional goals or states are specified distinctly and separately from roles, whether roles define agent capabilities and how roles are described in terms of responsibilities, outputs or procedures, and whether agents can mutually adjust their behaviour with some autonomy at run time.

With this general understanding of the different features in organisational meta-models, a selection of meta-models are individually analysed in more detail in the next four sections.

OMACS (S. A. DeLoach, Oyenan, and Matson 2008), JaCaMo+ (Baldoni, Baroglio, Capuzzimati, and Micalizio 2018a) and OperA+ (Jiang, V. Dignum, and Tan 2011) are all meta-models that have been designed with some consideration of flexibility or adaptability in agent behaviour, so in this section, these will be examined in more detail.

Organisation Structure					
Meta-model	**Explicit Organisation structure**	**Agents explicitly share intentions or outcomes**	**Structural change possible at run time**	**Select a predefined structure at run time**	**Fixed structure, flexible allocation process**
SharedPlans (B.J. Grosz and Sidner 1990)		✓			
KBORG (Sims, Corkill, and Lesser 2008)	✓		✓		
JaCaMo (Boissier, R. H. Bordini, J. Hübner, and Ricci 2014)	✓			✓	
JaCaMo+ (Baldoni, Baroglio, Capuzzimati, and Micalizio 2018a)	✓			✓	
OperA (V. Dignum 2004), OperA+ (Jiang, V. Dignum, and Tan 2011)	✓			✓	
OMACS (S. A. DeLoach 2009)	✓				✓
MOISEinst (Gâteau, Khadraoui, Dubois, and Boissier 2005)	✓				✓

Figure 3.3: Positioning a selection of existing meta-models regarding organisation structure

SharedPlans (B.J. Grosz and Sidner 1990; B. Grosz and Kraus 1999; B. Grosz and Hunsberger 2006) although not a meta-model, is also introduced as it provides a protocol for agents to dynamically create and adjust plans, similar to coordination by mutual adjustment as described by Mintzberg. Each meta-model is described and then some analysis and comparison of approaches used follows.

3.5.2 OMACS

Organisation Model for Adaptive Computational Systems (OMACS) (S. A. DeLoach 2009) has been proposed as a meta-model for organisational structures that require adaptation at run time. In OMACS, agents individually possess *capabilities* defining their skills. These capabilities are then matched to required capabilities for roles when allocating agents to enact roles. A problem is functionally decomposed at design time into a set of functions or tasks which are then later associated with roles (S. A. DeLoach,

Oyenan, and Matson 2008).

Each agent is assigned a role within the organisation. Roles achieve Goals. For every goal, there may be one role allocated. Agents possess capabilities. Role allocation is performed based on capabilities. Each role requires a set of capabilities. Agents must possess all the necessary capabilities in order to be potentially able to fulfil a role. OMACS makes an assumption that goals can be assigned and achieved by one, and only one agent playing an appropriate role in the organization (i.e., agents do not share goals) (S. A. DeLoach 2009).

OMACS enables dynamic system level reorganisation of organisational structures (goals and roles) as well as reallocation of agents to roles within an organisation. OMACS assumes that "Within the organisation, agents must have the ability to communicate with each other, ... accept assignments to play roles that match their capabilities, and work to achieve their assigned goals." (S. A. DeLoach 2009)

OMACS defines an organisation O as a tuple:

$<G, R, A, C, \Phi, P, \sum, oaf, achieves, requires, possesses>$

G represents goals of the organisation; R is a set of roles; A is a set of agents; C is a set of capabilities; Φ is a relation G✕R✕A agent/role/goal assignment; P is a set of policy constraints on Φ; $\sum$ is the domain model defining objects in the environment; *oaf* is an organisation assignment function defining the quality of an assignment set G✕R✕A; *achieves* is the function defining G✕R - how effective the behaviour of role R can be to achieve goal G; *requires* defines the capabilities required to play a role R; and *possesses* is a function defining the quality of an agent's capability.

To decide how well an agent can play a role, the requires and possesses functions are combined into a function: capable. To decide how well and agent can play a role to achieve a goal, the potential function is defined as the product of capable and achieves.

Policies defined at design time are abstractly used to define the processes for allocation of agents to roles (Assignment Policies), define behavioural obligations and relations between roles (Behavioural Policies) and define structural reorganisational processes such as how to reallocate tasks (Reorganisation Policies).

In OMACS, an organisation may be embedded within another organisation. Organisational Agents are a special agent representing an organisation. Thus, organisations are agentified first class entities. The organisational agent has capabilities and can be assigned to roles in a super-organisation. In this way, multiple agents in an organisation can fulfil a role in a super organisation by achieving that role's capability as an organisation. This allows for controlled team work where individual agents can

fulfil roles together as part of a sub-organisation. In this case, the coordination of the organisation relies upon individuals following a plan based on a goal design that organises the individual members. The goal design can order goals that are allocated to members.

It is possible to define multiple alternative roles that are capable of achieving a particular goal, however only one role can be allocated to one goal at any one time. DeLoach and colleagues have proposed that adaptability in planning can be addressed by having alternative paths available in a goal decomposition.

Coordination and Flexibility in OMACS OMACS does not provide for dynamic run-time coordination of multiple capable agents together performing one role to achieve a goal. Similar flexibility at run time would be achieved by the designer anticipating at design time and creating a role that could be achieved by an embedded organisation that has at design time, specified roles for agents to work together to achieve the goal.

In OMACS, the agents that populate an organisation are not considered part of the organisation, but the organisation is the structure that defines the behaviour of the agents (S. A. DeLoach 2009). As agents are not part of the organisation itself, it is not possible in OMACS for agents to self-manage an organisation in an emergent way or to dynamically create coordinated plans to enable individual agents to work together to fulfil a role. Rather, to adapt, the system can reorganise goals and reallocate agents to predefined roles in the organisation structure. In OMACS, the information required for coordination and collaboration between roles is embedded in the goals that are instantiated (S. A. DeLoach 2009). Defining capabilities enables flexibility in dynamic allocation and re-allocation of agents to roles based on a changing context. However, coordination between agents is implicit in the design of goals and agent's abilities based on capabilities associated with individual roles enacted by agents to achieve goals.

OMACS is associated with the creation of a dynamic goal design and run time representation using the Goal Model for Dynamic Systems (GMoDs) (S. DeLoach and M. Miller 2010). Using careful goal design to create a goal decomposition tree, goals can be ordered. The goal model indicates dependency between goals as well as alternative pathways to achieving a particular goal, so that if one set of goals cannot be satisfied, an alternative set may be chosen. This enables dynamic and flexible re-allocation of goals at run time based on the goal tree defined at design time.

When considering the potential for an organisation to be adaptive, the structure of the organisation needs to be examined - the roles defined, relationships between roles and the way that those roles achieve organisational goals. OMACS is designed

for adaptive organisations and suits situations where a restructuring of or reallocation of goals may be required in order to find a solution. This adaptation enables the organisation to be re-organised to find a different solution. An example of this might be when an agent leaves an organisation and a new structure is needed in order to achieve a goal. A number of potential solutions may exist using different configurations of agent-role-goal allocations.

The assumption that one agent in a predefined role can achieve each goal is significant. This assumption limits OMACS to the designs anticipated at design time and does not allow for goals that might be achieved by multiple agents coordinating individual actions dynamically, in order to fulfil a role to achieve a goal.

When agents are no longer available and goals cannot be met according to the original goal-role-agent assignment, the OMACS system automatically reorganises and newly revised roles or goals are selected based on the currently available agents' capabilities (S. A. DeLoach 2009). However, if there is no agent available with an exact match on a required role in the predetermined task decomposition, the OMACS system does not have the flexibility to address this. Another limitation is that the coordination is implicit in the task design and agent design, rather than being an explicit mechanism available to agents.

3.5.3 JaCaMo+

JaCaMo+ is an artifact based framework created by taking features from other meta-models to create one framework. JaCaMo+ uses interaction artifacts, an organisation meta-model, an agent model and an environment artifact. In JaCaMo+ the interaction artifact is external to the agent(s) and is used to represent a commitment based social layer that defines interaction protocols. Interactions between agents in terms of commitments are explicitly stored in the interaction artifact integrated as a first class entity in the system. The artifact represents the social state of the system. This helps to integrate the interaction between agents as an explicit observable component of the system. It also helps to decouple the interaction layer from the individual agent code.

Coordination in JaCaMo+ is achieved within an organisation structure using the *M*oise+ (J.F. Hübner, J.S. Sichman, and Boissier 2007) meta-model. In *M*oise+, social schemes are used to declare missions that describe the coordination of goals. Agents playing a role adopt goals that form part of these coordinated missions. Each mission is composed of one or more organisational goals. When agents are playing a role, norms define obligations for that agent in terms of missions that the agent should commit to achieving.

Individual agents in JaCaMo+, do not interact using an exchange of messages and cannot view the social state other than through role actions. "A JaCaMo+ interaction artifact encodes a commitment protocol, that is structured into a set of roles that agents can enact." (Baldoni, Baroglio, Capuzzimati, and Micalizio 2018a). The interaction artifact maintains the current social state as an explicit representation and is responsible for automatically providing this information to agents based on their context and the event they are focused upon. The interaction artifact behaves in a similar manner to an intermediary responsible for ensuring protocols for interaction are followed and for indirectly establishing sharing of beliefs between agents. The interaction component is defined in terms of transitions (based on obligations created for agents) between states.

In JaCaMo+, commitment protocols are associated with roles. Agents are programmed to react to obligations that are defined in the protocols. Commitments motivate agents to perform actions. Social commitments are represented explicitly as first class objects. This is based on concepts from the work of Singh (Torroni, Yolum, M. P. Singh, Alberti, Chesani, Gavanelli, Lamma, and Mello 2009). Agents create normative commitments explicitly using social commitments. In JaCaMo+, actions are associated with roles that can perform them, a context in which they can be performed and a set of associated commitment actions (such as create or release a commitment).

Social commitments in JaCaMo+ are first class entities that make agent's intentions explicit and enable agents to have clear expectations about how other agents will behave. The individual commitments between agents effectively coordinate these agents' activities. Additionally, commitment based interaction protocols enable agents to behave with some flexibility at run time without the need for specifying prescriptive scripts for interaction at design time (Baldoni, Baroglio, Capuzzimati, and Micalizio 2018a). Conditional commitments are possible in JaCaMo+ so agents can volunteer for a particular action and then later withdraw from this commitment if necessary. The interaction artifact specified at design time can guide agents at run time in terms of the creation or cancellation of commitments, without imposing a predefined or strict ordering to the interaction messages or actions.

The use of explicit commitments and commitment protocols provides some flexibility in JaCaMo+. A limitation is that these are tied explicitly to roles and that agents cannot autonomously interact directly with other agents outside of the role based protocols.

3.5.4 OperA+

OperA (V. Dignum 2004) and a related meta-model, OperA+ (Jiang, V. Dignum,

and Tan 2011), are contract based organisational meta-models. The organisational structure is described using three structures - *social, communicative and normative*. The organisational roles described in the organisational social structure implicitly define the organisational objectives, so there is not a separate functional specification of goals. Rather, by allocating agents to enact roles, the organisation is guaranteed to reach its objectives. By coupling roles with functionality in this way, it is presumed that there will always be agents available to enact roles and that the roles can be specified in a static way at design time. This limits flexibility in the resulting organisation because it assumes that there will always be agents available to fulfil each required role.

The specification dimension of the OperA+ organisation defines the organisation structure as a set of related roles. The enactment or social dimension specifies at run time the agents allocated to enact roles. This is similar to JaCaMo+ although the interactive social dimension in OperA is represented in contracts between agents, rather than in a separate interaction artifact.

Roles are also aligned with specific norms to constrain agent behaviour. Agents do not have an identity within the organisation separate from the role in which they are playing. Each role requires a set of capabilities and agents can be allocated to enact roles based on the capabilities they possess. Contracts are established at run time to specify which agents are performing or enacting which roles. When agents are allocated to roles, they become role enacting agents. When agents are allocated to roles, there must be an exact match of agent capabilities to fit the role. This is a limitation that restricts flexibility at run time if agent's capabilities degenerate or agents leave the system.

A gate-keeper agent is responsible for dynamic role allocation by matching agent capabilities with defined roles.

In OperA+, organisations are viewed within a hierarchical abstraction so an organisation itself can be treated as if it were an agent in a higher level organisation. This provides for some flexibility as the organisations can represent groups of agents capable of achieving a particular functional role in the higher level organisation.

Methodological design guidelines for the development of a MAS using OperA+ include processes for role identification, organisation identification and capabilities identification (for matching roles to role-enacting agents). The interaction dimension is defined using scenes. Scenes describe situations and contexts for defining agent interactions (including implicit coordination of agents). Scenes describe how agents will interact to achieve landmark states. Landmark states can be thought of as objectives. It is up to the individual agent how an objective is reached, but the interaction between

agents is defined in the scene script. It is possible to define multiple alternative scene scripts at design time so that agents have some flexibility to match up with these at run time.

Flexibility with OperA+ relies on careful specification of roles at design time including composite roles that can be adopted by an organisation. Coordination is implicit within the scene scripts written at design time. This is a limitation as agents do not have an explicit mechanism to allow individual mutual adjustment or dynamic negotiations to coordinate behaviour (outside of the predefined scene scripts).

3.5.5 SharedPlans

SharedPlans (B.J. Grosz and Sidner 1990; B. Grosz and Kraus 1999; B. Grosz and Hunsberger 2006) provide a framework for the deliberate sharing of decision-making between team members.

SharedPlans is based around agents committing to mutual intentions about plans. They also commit to ensuring that all agents concerned ensure such intentions, once adopted, are maintained as mutual. Intentions are categorised into four specific types - *intentions-to* perform an action, *intentions-that* a proposition be held true, and potential intentions (identified, but not yet adopted) for each of *potential-intentions-to* and *potential-intentions-that* (B. Grosz and Kraus 1999). Recipes for achieving a group activity, are defined in terms of sub-acts and parameters that can be filled dynamically. Collective group intentions are used to constrain that all participants have the uniform intentions, and that any agent updates their intentions only in accordance with the group (B. Grosz and Hunsberger 2006).

The formalisms used in SharedPlans (B. Grosz and Kraus 1999; Hunsberger 1999), define modal operators: Mutual belief, Intention to, Intention that and Belief shown in Figure 3.4.

Mutual Belief ($MB(GR, \phi)$)	group GR mutually believes proposition ϕ
Belief $Bel(G, \phi)$	Agent G believes proposition ϕ
Intention.To ($Int.To(G, A)$)	Agent G intends to do action A
Intention.That ($Int.Th(G, \phi)$)	agent G intends that proposition ϕ holds

Figure 3.4: Modal Operators used in SharedPlans (Hunsberger 1999)

"Agents have a SharedPlan to do α if and only if they hold the following beliefs and intentions: 1. individual intentions that the group perform α; 2. mutual belief of a (partial) recipe for α; 3. individual or group plans for the sub-acts in the (partial) recipe; 4. intentions

that the selected agents or subgroups succeed in doing their sub-acts (for all sub-acts that have been assigned to some agent or group); and 5. (in the case of a partial SharedPlan) subsidiary commitments to group decision-making processes aimed at completing the group's partial plan." (B. Grosz and Hunsberger 2006)

There are supporting processes identified as associating with maintaining intentions. In particular the *cultivate* process is of interest - a complex planning action that is responsible for creating support processes to deal with the dynamics of shared plans and enable coordination between team members. In particular, whenever a team has an intention-that some proposition holds, the cultivate process will ensure appropriate plan based reasoning regarding group intentions (B. Grosz and Kraus 1999). In the SharedPlan formalism, agent commitment to a group is enforced by the adoption of intentions-that toward planning so that agents reason about group activities. Grosz and Hunsberger (B. Grosz and Hunsberger 2006) introduce the term *intention cultivation* to refer to this reasoning, in support of group commitments that accomplishing shared intentions, cooperation and coordination.

Group activity related intentions are maintained by cultivation of a set of group decisions that must be made. Each team member commits to a goal and also commits to 'the team' successfully reaching that goal. In the case of the actual plan for action on the goal being incomplete, the team also commits to finding a solution.

SharedPlans can be adopted when incomplete, with a commitment to expand to completion over time. The focus is two-fold. One is to provide a dynamic structure decomposing a goal hierarchy, shared between team members. The second is obligating agents to create a commitment to a shared intention between members that they will collaborate on group actions to achieve the goal.

At first glance, it appears that SharedPlans provide a good formalism for describing this collaboration needed in dynamic and complex situations. However, typically the complex problem solving relies not only on collaboration relating to planning and performing actions or tasks, but there is also a need for some explicit sharing of emergent knowledge regarding a changing environment (including both situation awareness and organisation structural awareness). This requires collaborative-capable agents, with additional coordination capabilities. Resources and situation awareness about an evolving situation need to be shared as well as plans. SharedPlans provides the framework for collaborative-capable agents (B. Grosz and Hunsberger 2006) but not an explicit coordination framework for sharing knowledge.

3.5.6 Coordination in organisational meta-models

In this section, selected organisational meta-models are analysed and compared in terms of features enabling coordination. The coordination mechanisms in these models are examined and the level of agent awareness that would enable agents to improvise or mutually adjust their behaviour in order to coordinate dynamically with others.

Coordination devices adopted in various frameworks include negotiation protocols, transitions between scenes, shared artefacts, social schemes and design activity diagrams (Boissier, V. Dignum, and García 2016). Coordination by proxies or intermediate layers within an agent architecture has been suggested to enable open systems with heterogeneous agents to work together. For example, Scerri and others (Scerri, Pynadath, Schurr, Farinelli, Gandhe, and Tambe 2004) assign a proxy agent responsible for coordination to each team member. This enables domain specialised agents to work as part of a larger team, without the need for knowledge about the team itself. This has proven to be effective in some situations (Schurr, P., Pighin, and Tambe 2006). However, with this approach, the agents are not able to directly reason about team issues or coordination and this limits the applicability of this approach. Coordination of task allocation has been achieved using Contract Net Protocols (CNP). This allows agents to follow a protocol for requesting and bidding to perform tasks as described by Boissier and colleagues (Boissier, R. Bordini, J. F. Hübner, and Ricci 2019) who use CNP with artifacts to coordinate which agent will perform a task. This approach is good for an open situation where it is not clear which agents may be available with the capability to perform a task. It is not necessary in the context of the problems under consideration to find the 'best' bidding agent who can perform a task, rather the focus is upon finding a solution by any agent who is capable of achieving this in a timely way. So, this level of deliberation regarding which agent will complete a task is not necessary. It is desired in our context that agents adopt tasks and use their initiative to complete tasks in a timely way. Agreement is only needed when agents need to work together.

The coordination approaches used in existing meta-models use mechanisms ranging from pre-specifying coordination implicitly at design time (good for predictable problems) or pre-specifying alternative solutions that are matched to context at run time (good where alternative scenarios can be anticipated) through to more explicit coordination using external coordinators, middleware or special organisational agents that create organisational structures at run time. When coordination of tasks must be negotiated at run time, this relies on interactions can be directed based on agents complying with predetermined (context based) guidelines, policies or norms. The domain

characteristics are important in the selection of a meta-model and approach to coordination. A number of coordination approaches are described in (Aldewereld, Boissier, V. Dignum, Noriega, and Padget 2016).

In figure 3.5, Mintzberg's coordination mechanisms for human organisations are used to compare selected organisational meta-models for MAS.

Coordination mechanism	**MAS Implementation Features**	KB-ORG	OMACS	MOISE	MOISE+	MOISE *inst*	OperA+	ORA4 MAS
Mutual adjustment	Individual agents liaise with others to ensure coordination dynamically	×	×	×	×	×	×	×a
Direct supervision	Agents are coordinated explicitly	✓ e,i	✓ e	×	×	×	✓ e,i	✓ e,i
Standardise work tasks	Activities are well specified and described in well defined tasks. Tasks are allocated to agents or roles. Task design may be external at design time	×	✓	×	×	×	✓	✓
Standardise outputs	No specification of how to achieve tasks but clear specification of outputs expected as a result	✓	✓	✓	✓	✓	✓	✓
Standardise skills	Roles are clearly specified with competencies and protocols defining behaviour. Roles related to tasks to achieve organisational objective	×	✓	×	✓	✓	✓	✓
Standardise norms	Norms clearly specify agent behaviour	×	×	✓	✓	✓	✓	✓

Legend	
✓	Yes is present
×	No is not present
e	direct supervision external coordinator
i	direct supervision managed internally in MAS
a	using external artefact

Figure 3.5: Coordination mechanisms in organisational meta-models

The first column in Figure 3.5 lists each coordination mechanism. The second column articulates how this could be manifested in MAS. Subsequent columns indicate whether or not each meta-model listed in row 1 includes each of the coordination mechanisms. The organisation models in row 1 are models discussed in the previous section. KB-ORG is also included because this is a system that is attempting to address the need for adaptability in organisation design. JaCaMo+ uses MOISE+, so each of the MOISE models are included for comparison in this table. The table indicates yes

(✕) or no (✓) for each meta-model if it provides features relating to each coordination mechanism. In the case of direct supervision, the table indicates how the role structure is decided and supervised (allocated) as follows: An *e* represents an external coordinator (designer or external system) and an *i* represents an internal coordinator (leader or coordinator role or organisational agent). Existing agent organisational meta-models do not necessarily apply a particular mechanism exclusively. This is similar to real life situations where coordination may involve a combination of different mechanisms.

If agents are required to dynamically coordinate on interdependent tasks (such as when following Mintzberg's mutual adjustment coordination approach (Mintzberg 1980)), agents need some autonomy to make operational decisions as well as some level of awareness of others. Using layers of specification and abstraction is one approach to providing this autonomy to agents. This can be achieved by having an organisation treated as if it were an agent in a higher order organisation and then the details of how members of that organisation achieve the organisation's objective is left to the specific lower order organisation (e.g. (S. A. DeLoach 2014; V. Dignum 2004; G. Tidhar 1999; García, Valero, and A. Giret 2016)). Another approach is to have higher order missions or guidelines with agents given some autonomy at the operational level (Corkill, Durfee, Lesser, Zafar, and C. Zhang 2011; Gâteau, Khadraoui, Dubois, and Boissier 2005). High level guidelines have been used to describe constraints on how organisational objectives should be decomposed in a hierarchy. Separately, operational objectives represented as leaf goals in their goal decomposition can be operationally coordinated as required by the individuals involved in each team (Corkill, Durfee, Lesser, Zafar, and C. Zhang 2011; Sims, Corkill, and Lesser 2008). Agents operate individually at a micro level, however, it is important to ensure macro level success of the collective system. Meso-level mediation and control can ensure that micro-level, operational decision-making does not interfere with or cause undesirable macro outcomes (Pitt, Bourazeri, Nowak, Roszczynska-Kurasinska, Rychwalska, Rodriguez Santiago, Lopez Sanchez, Florea, and Sanduleac 2013).

OMACS achieves coordination based on the mechanisms of standardisation of outputs, tasks and skills. The organisational goal is decomposed at design time into possible sets of sub-goals that will reach success. Dependencies between goals are represented and non-leaf goals may be specified using AND and OR. A leaf goal is achievable by an individual role in the organisation. Each possible decomposition of a non-leaf goal represents a set of goals that will lead to a coordinated solution. The designer selects these alternatives to enable coordination by standardisation of outputs from each goal. At run time, based on the skills of agents available, a dynamic goal tree instance is created representing the goals that are valid. The coordination is embedded

in the goals and each goal has an associated capability skill set. The agents allocated to each task must have the required skills (capabilities) to ensure the correct outcome (standardisation of skills).

The organisational goal tree instance is chosen based on a run time algorithm that chooses valid goals as the best solution based on the agents available with the required capabilities and their potential scores for achieving tasks. All potential assignments of agent-role-goal mappings are given a value between 0 and 1. A value of 0 indicates that the agent is not potentially able to be assigned. Allocation of agents to tasks is done by the system itself. The default organisation assignment function assigns agents based on maximising the sum of potential scores in an assignment set. Domain specific functions can be written to override this. This approach provides flexibility at run time in terms of agent allocations, however, the coordination is external and agents individually do not have awareness of how their actions fit in with others. Agents do not therefore have any ability to self-organise to adopt goals that need to be completed or mutually adjust their behaviour in an emergent style at run time.

Coordination by standardisation of outputs is also used in the MOISE organisational meta-model (Hannoun, Boissier, J. Sichman, and Sayettat 2000) and it successors MOISE*inst* (Gâteau, Khadraoui, Dubois, and Boissier 2005) and MOISE+ (J. Hübner, J. Sichman, and Boissier 2002). The main goal is functionally decomposed into a set of missions describing the required functioning of the system. These missions define the required outputs, not how each functional goal is achieved. A scheme is defined outlining all potential missions (plan paths or sub-goal sets) that could achieve the main goal.

In MOISE+ and MOISE*inst*, agents are given more autonomy by defining norms that define acceptable behaviour for the roles. This achieves coordination by standardisation of the roles (skills). The MOISE meta-model is used in JaCaMo+ however, in JaCaMo+, agents do not coordinate within the organisation by direct interactions. An external artifact is instead used to coordinate agents. This would seem to limit the ability of agents in JaCaMo+ to mutually adjust their behaviour to fit in with others because they are only able to communicate using artifacts and predefined role based interactions.

Another extension of MOISE is the ORA4MAS system (J. Hübner, Kitio, and Ricci 2010). ORA4MAS is an artifact based implementation of MOISE and uses organisational agents to manage the organisation by referring to organisational artifacts that define control for the organisation. The artifacts can be adapted at run time as the organisation is considered a first class entity. ORA4MAS describes an Organisa-

tional Management Infrastructure(OMI) that sits above the organisational model. The OMI serves as a mechanism for agent awareness in order to control mutual knowledge. An organisational middleware agent manages the organisational infrastructure. Artifacts include the organisational board that describes the structure and members, the group board that describes the role assignments and the scheme board that describes the task decomposition into missions and commitments by agents. ORA4MAS is used in JaCaMo to create a framework for managing multi-agent organisation based systems (Boissier, J. Hübner, and Ricci 2016).

Another variant on describing coordination requirements is used in Organisationally Adept Agents (OAA) (Corkill, Durfee, Lesser, Zafar, and C. Zhang 2011). OAA conceptually describe the coordination needs in terms of agent awareness and run time adaptability requirements. Organisational restructuring is governed by context based guidelines for organisational control. In the description of OAA, Lesser and colleagues make a distinction between long term organisational control and the operational decision-making within the agent organisation (Corkill, Durfee, Lesser, Zafar, and C. Zhang 2011). Such agents would be capable of coordinating using mutual adjustment and also restructuring the organisational structure at run time.

An approach using coordination by direct supervision by an entity external to the OCMAS is used with KB-ORG (Sims, Corkill, and Lesser 2008). However, in this case, the external entity is the KB-ORG system itself. In KB-ORG, an agent organisational structure is created externally and then provided to the organisation. When necessary, the KB-ORG system creates coordination roles to ensure coordination is achieved. The organisational design produced is high level and does not attempt to address low level operational coordination. KB-ORG is an automated organisation design system and not an agentified system. However an advantage of KB-ORG is that it allows for roles to be split and when that occurs, to instantiate special explicit coordination roles. These help manage the coordination between multiple roles performing what might have previously been a single role. At the time of splitting a role, KB-ORG also identifies that there is a temporary team of agents formed that must coordinate and communicate appropriately in order to achieve the role. This dynamic organisation then is governed by obligations to coordinate. As the restructuring of the organisation occurs externally, outside the agent system, KB-ORG is not applicable for a system in which agents need to be aware of their organisation.

3.5.7 Organisational adaptation in MAS

In dynamic, complex situations, adaptability of the organisation structure and state at run time is important to enable the organisational MAS to function successfully. The adaptability of an agent organisational model can be described in terms of the ability for an organisational structure (roles) to change as well as the ability to change the state of current allocations of agents to tasks (or roles that achieve tasks). The task decomposition can be specified at design time, with multiple alternative pathways that can be selected based on context at run time (e.g. MOISE, OperA, OMACS). Alternatively, a new structure can be re-designed at run time (e.g. KB-ORG).

Defining a structure at design time that can be used (or adapted) at run time is a top down approach. When using a top down approach, a distinguishing feature between meta-models is the functional decomposition of tasks and the flexibility of how agents are allocated to tasks (e.g. via roles, or via capabilities). If a problem is described as a set of tasks and then each task is matched to possible roles (e.g. OMACS, MOISE, KB-ORG), and if there is more than one way that this matching can occur, then this affords agents more flexibility at run time than if the problem is directly described in terms of roles that must be fulfilled (e.g. OperA). Further run time flexibility is gained by matching roles to agents by capabilities rather than exact agent-role mapping (e.g. OMACS). Flexibility in task decomposition relates to the ability for an organisational structure to be changed. External re-organisation of the structure is provided in KB-ORG and internal re-organisation is facilitated in OMACS and OperA by specifying alternative goal-subgoal pathways or alternative scene scripts respectively at design time.

Self-organising systems adopt more of a bottom up approach where agents emerge into an organisational structure, rather than a structure being imposed top down. Some flexibility is achieved by using norms, guidelines or policies to constrain agents autonomy so that agents know the boundaries of acceptable deviation from predefined scripts. Top down design on the other hand, enables some encoding of organisational memory and knowledge with operational procedures and plans that can be selected at run time, however considering that it is not likely that the run time situation will always exactly match an existing design script, there is a need for agents to adapt and allow a solution to emerge based on the run time situation.

The domain requirements of improvisation and emergent coordination by mutual adjustment would suit using a mixture of top down design and bottom up emergent behaviour with a high level of agent awareness. The top down design could provide some structure and then organisation aware agents could improvise and allow structural

change or improvisation at run time.

3.5.8 Flexibility at the agent level: role and goal adoption

A goal or objective representing a problem needs to be mapped to agents who are responsible for performing actions to achieve that objective. In intention based, goal directed MAS with agents built based on the BDI (Beliefs, Desires, Intentions) architecture, each agent performs deliberation based on percepts received from the environment. In the deliberation process, the agent updates internal beliefs about the status of the environment and considers potential objectives that could be achieved, then creates intentions to adopt a goal and act to achieve its outcomes. Adopting a goal involves the agent selecting from a predefined library of plans that describe the actions that will achieve the selected goal. Generally the plan library contains plan rules that provide a sequence of actions that the agent can adopt. These actions are conditional on the current context represented as a set of beliefs about the environment. Each agent is programmed with decision-making processes to aid deliberation and goal selection. These decision making processes can also be influenced by policies or norms. The outcome of a deliberation process using means-end reasoning is a goal adopted then the selection of a plan that will reach that goal. There is not an explicit look ahead planning stage such as for example, described by Dimopoulos (Dimopoulos and Moraitis 2006) involved.

There has been research to suggest that there is value in incorporating an explicit planning process in the deliberation used in BDI systems (Sardina, De Silva, and Padgham 2006). However, look ahead planning involves agents spending time deliberating rather than acting. Modelling real-time constraints that agents should meet is important so that the MAS system is robust (Ashamalla, Beydoun, and Low 2017). It could be useful to use planning algorithms to capitalise on the fact that multiple paths or options may be possible in terms of a solution and enable agents to deliberate in order to generate or select the best plan. The focus in the current project is for agents in the domain to find any workable solution, rather than performing look ahead planning and deliberation in order to find the most efficient solution. So, the requirements demand a mechanism for selection of one action or goal on a pathway of tasks that lead toward reaching each desired goal. This can be satisfied by using predefined plan libraries. This is particularly the case if there is an ability to describe these in terms of goal states that may be reached in multiple ways. If agents use declarative rules that match to a particular set of beliefs, then it may be that the agent will select the first matching rule specified and adopt that without further deliberation. If agents

are capable of identifying multiple possible solutions at run time, then it is necessary to provide a mechanism for agents to prioritise and deliberate to choose a action to commit to. This could be achieved by specifying metrics to indicate the 'goodness' of a solution, similar to OMACS. It also could be achieved by using an explicit planning process as part of the agent deliberation reasoning cycle.

Flexibility at run time can be gained by providing alternative goal decomposition paths at design time so that there are alternative plans that can be chosen based on the run time context. When a problem is definable and not likely to change in terms of work flow and required tasks, but agents available may change, then providing alternative paths of sub goals to reach a goal allows adaptation at run time based on the agents available. In the event that there may not be agents predictably available with matching capabilities or roles to associate with the goals described at design time, then a flexible system is needed. A flexible system would enable agent behaviour at run time so that agents can be assigned or can adopt goals that they are capable of achieving. If permitted, agents can deliberate and improvise to choose intentions based on goals they are capable of achieving, without being constrained by roles or norms. Flexibility must be considered in the organisation meta-model components and processes used for problem representation, role specification and goal adoption. How the organisation structure in meta-models defines functional objectives and allocates agents to responsibilities in the organisation will now be examined.

In models such as OperA+ and OMACS, the organisational model defines a set of roles that achieve the system goals. In OperA+, the objectives of an organisation are achieved by a direct mapping of objectives to roles that are enacted by agents within the organisation. OperA+ does not suit our requirements because it needs agents available at run time with an exact match to be able to enact the organisational roles. Flexibility relies on careful specification of the organisation at design time defining alternative atomic or composite roles. JaCaMo is also built on an assumption that roles can be predefined at design time and agents adopting a role will be capable of committing to and achieving goals that are specified in associated missions. JaCaMo makes explicit the possibility that a number of potential schemes can be defined with high level guidelines for instance stating the number of roles required. Also, if agents have reasoning capabilities to redesign the functional schemes at run time, there is potential for agents to view a functional scheme and change it. This enables some flexibility at run time, although still requires that agents who are able to enact the required roles are available.

KB-ORG adds a layer of abstraction between goals and roles by defining a re-

sponsibility list to be associated with each goal, and each role has a list of assignable responsibilities it can perform (Sims, Corkill, and Lesser 2008). In this way, goals can be achieved by multiple roles and each role can achieve multiple goals. KB-ORG generates an organisational structure, including organisational roles as appropriate to control the system. The KB-ORG system is designed for automatic allocation of tasks to agents in an organisation (Sims, Corkill, and Lesser 2008). In KB-ORG, roles contain an assignable list of responsibilities and if necessary, roles can be split between a set of agents. In such a case, explicit coordination goals and roles are created. The system separates application level goal decomposition from coordination and associates roles with each leaf goal. After KB-ORG creates an organisation structure, then agents populate that structure.

In the MOISE approach (Gâteau, Khadraoui, Dubois, and Boissier 2005; Hannoun, Boissier, J. Sichman, and Sayettat 2000), a distinction is made between a separate structural specification and a functional specification. The functional specification or decomposition of tasks to be achieved by similar multi-agent systems has been referred to as a goal tree that can be associated with individual plan recipes that achieve each leaf goal (Horling and Lesser 2005; Sims, Corkill, and Lesser 2008). Goal Trees that decompose goals into tasks and allow for synchronisation or coordination relationships can be seen in other models e.g. STEAM (Tambe 1997) and TAEMS (Lesser et al. 2004). MOISEInst (J. Hübner, Kitio, and Ricci 2010) seems to have some added flexibility by abstracting objectives into missions that encompass multiple tasks. So, it seems that flexibility can be achieved by adopting entities to keep the functional specification and the structural specification separate or loosely coupled. Also, ensuring that in the design process, the decomposition of goals is performed at design time to allow for flexibility at run time to adapt and find alternative pathways. This is particularly important when agents may come and go or their capabilities degrade and a new solution path may be sought to match with different agents.

As discussed, the flexibility can be gained in a number of ways. One way to achieve flexibility is to provide a functional specification independent of the agent structure. Another important approach to flexibility involves enabling the creation of a dynamic organisation structure of agents at run time, based on the situation. In addition to the requirement of adaptability is the need to ensure that agents communicate appropriately so that knowledge about plans and situational changes are appropriately shared. The problem of how to enable agents to interact to dynamically improvise and adjust their goals to fit in with others in a coordinated way (beyond following a fixed script or plan, or reallocation based on predefined roles) is important to the current work. In addition to agents in the organisation needing flexibility in terms of the goals that can

be adopted in order to reach a solution, the agents need to have a model of knowledge sharing. If agents have a mechanism to share information about the situation and the state of the organisation with other agents, then they will be able to coordinate their individual plans as well as synchronise interdependent activities. In the next section, methods used to guide agent interactions are explored.

3.6 Supporting agent interaction

Having introduced the need for coordination, adaptation and improvisation in the organisation operation at run time, it is necessary to explore how this is achieved using meta-model design components. It is also important that agent interactions are managed so that agents share information appropriately. In this section we explore mechanisms adopted to manage appropriate interactions between agents.

3.6.1 Social policies and social commitments

In this section, the use of policies and commitments to manage agent interactions is discussed. Interactions between agents can be explicit or can be implicitly built into agent individual plans or interaction scripts at the organisation level. There are advantages in using a declarative, social approach to modelling agent interaction when agents are working without necessarily sharing common internal mental attitudes such as goals, desires or intentions (Torroni, Yolum, M. P. Singh, Alberti, Chesani, Gavanelli, Lamma, and Mello 2009; M. P. Singh 2012). Using social semantics such as commitments provides for flexibility so that agents do not have to follow fixed scripts defining interactions, but adhere to predefined interaction constraints. Explicit interactions are traceable, the communication can be established by following policies that specify agent obligations for when and who to communicate with. If agents are responsible for choosing to share knowledge with others, agents need to know who else is relevant and interested in shared knowledge. The organisational structure can help establish a context for communication. The organisation can also be used to define context for the norms or policies that apply to agents.

Within the organisation, it is possible to use policies to guide agent creation of commitments and in so doing, adopt the position proposed by Singh, that a policy is a higher order meta-commitment (M. P. Singh 2012). Using social policies as guidelines within an organisational context to govern the creation of social commitments can ensure that appropriate coordination of knowledge and behaviour is achieved (Carabelea and Boissier 2006; Telang, Meneguzzi, and M. P. Singh 2013; Van Diggelen, J.

Bradshaw, Johnson, Uszok, and P. Feltovich 2009). The social policy can be defined at design time particular to a context, enabling agents to create public social commitments at run time (Van Diggelen, J. Bradshaw, Johnson, Uszok, and P. Feltovich 2009). A social commitment is created between two agents toward achieving a particular goal. Explicit social policies can obligate members of an organisation to adopt interaction and coordination commitments e.g. (J. M. Bradshaw, Uszok, Breedy, Bunch, T. C. Eskridge, P. J. Feltovich, Johnson, Lott, and Vignati 2013; M. P. Singh 2012).

Policies have also been categorized as relating to either authorisation or obligation (including coordination) policies (J. M. Bradshaw, Uszok, Breedy, Bunch, T. C. Eskridge, P. J. Feltovich, Johnson, Lott, and Vignati 2013). Collective obligations can be implemented as policies to govern joint activity and teamwork (Van Diggelen, J. Bradshaw, Johnson, Uszok, and P. Feltovich 2009). Policies have been used similarly to describe communication protocols between agents as conversation policies (Kremer and Flores 2006). In FAML policies are external design time classes that describe the organisational rules, however agents are not aware of the policies at an internal agent-level (Beydoun, Low, Henderson-Sellers, Mouratidis, J.-J. Gomez-Sanz, Pavón, and Gonzalez-Perez 2009). In OMACS, policies are abstractly used to define the processes for allocation of agents to roles *(Assignment Policies)*, define behavioural obligations and relations between roles *(Behavioural Policies)* and define structural reorganisational processes such as how to reallocate tasks *(Reorganisation Policies)*.

Just as important as the social policies created at design time, are the social commitments created between agents at run time. Social commitments have been used to express an agent's commitment to another agent to perform something (M. P. Singh 2012). Social commitments are relational - based on two or more agents interacting so they make behaviour explicit and predictable. This is important especially when working in a domain in which agents and humans potentially need to interact (Bunch, Carvalho, J. M. Bradshaw, T. Eskridge, P. J. Feltovich, Lott, and Uszok 2012; N. R. Jennings 1993). Adaptability is of interest and the avoidance of pre-scripting detailed interactions with strict protocols at design time. Social commitments provide an observable and adaptable specification of expected agent interaction (Baldoni, Baroglio, Capuzzimati, and Micalizio 2018b). Using social semantics such as commitments provides for flexibility so that whilst agents do not have to follow fixed scripts defining interactions, they do adhere to predefined interaction constraints.

Social commitments have been used to define dynamic social contracts between agents in an organisation (Carabelea and Boissier 2006). Obligations can be thought of as directed commitments between actors. Policies describe "how an agent (or organisa-

tion) may decide to act upon- or decide not to act upon- a commitment.." (M. P. Singh 2012).

Fornara's work with institutions of agents and with the Ontology, CommitmEnts, Authorizations, Norms (OCeAN) meta-model also uses commitments (Fornara and Colombetti 2007; Fornara and Colombetti 2009; Fornara and Balke-Visser 2018). OCeAN provides a general framework for heterogeneous agents to work together in open systems. There is no assumption on the internal structure of an agent. Social commitments are created between agents activated based on Norms or communicative acts. Norms are used to model obligations and prohibitions externally imposed on agents based on the roles being enacted by agents at the time, so norms govern the behaviour of agents. Fornara's work provides a broad meta-model to enable agents to work together where agents are not part of an organisation but an institution. Commitments are explicit and independent of the agent's individual architecture but fundamental to the operation of the institution. In this context, norms are enforced using sanctions or rewards. Agents do not commit to an organisation goal and are not aware of or collectively working toward a common objective. A limitation of this approach is the lack of flexibility for agents to ignore a role based norm without a sanction.

It is possible to use norms to express interaction obligations within the organisation context e.g. (Vázquez-Salceda, V. Dignum, and F. Dignum 2005). Norms have been used to regulate agent behaviour in organisations. The ROMAS meta-model (García, Valero, and A. Giret 2016) integrates the concepts of contract, norm, organizational structure, service, role and agent. The focus is on open systems that require regulation. Organizations are used to constrain agent behaviour with norms and define relationships in the org structure. Norms can be associated with roles or organisations. Agents must play a role in the organisation and agree in a contract to accept the norms associated with that role. Roles are associated with the desired functionality of the organisation system. Contractual agreements represent commitments between agents. Cooperation and communication between agents in an organisational unit is specified at design time so this meta-model does not offer the desired flexibility within the organisation at run time.

Social policies are very similar to the norms in Normative Organisation Programming Language (NOPL) used to combine an organisation meta-model (MOISE) with a higher level organisation management system (J. Hübner, Boissier, and R. H. Bordini 2009). The obligations created from norms in NOPL are similar to social commitments if very specific obligations are created, particular to goals for communicative acts. Norms have been used to create obligations for agents, in a similar way to the use

of social policies. Hübner and colleagues show that norms can be formally expressed using the language NOPL (J. Hübner, Boissier, and R. H. Bordini 2009; J. Hübner, Boissier, and R. H. Bordini 2010). In NOPL, norms express conditions that result in obligations (with an associated time deadline) or a direct fail status (in the case that the norm is used to specify a regimented rule that cannot be ignored). NOPL provides a language to clearly express conditions which trigger obligations for particular agents. The obligation refers to a state of the world that the named agent must try to bring about and is declared as : $obligation(a, r, g, d)$ for agent a, based on reason (norm) r, the obligation to reach goal state g, by deadline d. In NOPL, the obligations are defined as a result of a norm declaration. The norm can specify obligations for agents to accomplish particular goals and can be used to assign responsibility for goal achievement based on an agent's role. Norms in NOPL can create obligations regarding goal completion and goal adoption. The obligations are related to goals that can be fulfilled when a declared state is achieved. An obligation can be in one of the following states: active, fulfilled, unfulfilled or inactive. NOPL norms could be used to define obligations requiring agents to create goals relating to the creation of social commitments. The use of temporal conditions in NOPL offers a potential advantage that is not provided in by social policies. If temporal constraints were needed with social policies, these would need to be specified as constraints associated with the policy. Social commitments can be defined using temporal logic. With regard to the OJAzzIC meta-model, the most important point is that an explicit mechanism is needed in order to specify obligations for agents regarding behaviour and resulting in agent commitments. It may well be argued that this could be achieved using either social policies or norms.

Coordination by proxies or intermediate layers within an agent architecture has been suggested to enable open systems with heterogeneous agents to work together. For example, Scerri *et al* assign a proxy agent responsible for coordination to each team player (Scerri, Pynadath, Schurr, Farinelli, Gandhe, and Tambe 2004). This enables domain specialised agents to work as part of a larger team, without the need for knowledge about the team itself. However, with this approach, the agent players are not able to directly reason about team issues or coordination and this limits its applicability in our context. Stone and colleagues (Stone, Kaminka, Kraus, Rosenscheind, and Agmona 2013) describe ad hoc teams, similar to our requirements, teams that may form or emerge without fixed scripts to follow. Their approach is purely utility based and they focus on the situation when there is no communication between agents. Though the problem they address shares some similarity to ours, we are allowing for some shared mental models and explicit communication. They are using utility based agents (not using explicit mental attitudes), we are interested in BDI agents with explicit awareness

between agents of beliefs, goals, plans, capabilities, responsibilities and obligations.

In addition to having a mechanism for managing interactions between agents in an organisation meta-model, it is important that agents clearly understand the tasks that they should be adopting. In the next section, models for knowledge sharing are examined.

3.6.2 Models for knowledge sharing

To perform successfully, coordinated agents need to have access to a shared knowledge base within the organisation (J. Hübner, Kitio, and Ricci 2010). Sharing knowledge enables the agents to mutually adjust to fit in with each other and work together to reach the organisation goals. In the approach used in the MOISEinst (Gâteau, Khadraoui, Dubois, and Boissier 2005) and ORA4MAS (J. Hübner, Kitio, and Ricci 2010) systems, each organisation is considered as a first class entity, so agents have access to the mental state of the organisation directly. However, in these systems, the first class entity does not have agent characteristics, but is an external tool or artifact that agents use. This contrasts with Tidhar (G. Tidhar 1999) who defines an organisation itself as an agent in a system of related agents and other organisations. Tidhar describes teams as meta-agents, with reasoning abilities to deliberate about the team. In both of these approaches, the organisation itself has associated artifacts (conceptually similar to internal beliefs) that can be inspected or queried by agents in the organisation. These artifacts encapsulate shared knowledge about groups within the organisation. They also provide details about plans or schemes that have been adopted as well as norms that define behaviour and obligations on agent members.

Sharing a mental model as a representation of the shared knowledge of the situation has been deemed important for agent coordination in a team oriented architecture (Yen, Fan, S. Sun, Hanratty, and Dumer 2006). The sharing of knowledge regarding plans can be achieved by agents using a strict shared mental space or agreed public commitments to particular intentions such as in SharedPlans (B. Grosz and Hunsberger 2006). However sharing a mental model of the situation may not of itself be enough.

The shared mental model in the RCAST architecture is more than a set of mutual beliefs, all agents share a commitment toward maintaining shared awareness and proactive sharing of relevant information (Yen, Fan, S. Sun, Hanratty, and Dumer 2006). This is similar to the approach used in JaCaMo+.

It is suggested that human team cognition is all about the interactions, not just the shared mental models present. The team cognition relates to how the team interacts

to create and share information (N. J. Cooke, J. C. Gorman, Myers, and Duran 2013). It is beneficial to the agents' adaptability if agents have a general understanding of the high level plans of others in the organisational space (Corkill, Durfee, Lesser, Zafar, and C. Zhang 2011). The context of agent interaction and relationship is important in terms of agents being able to identify relevance of future interactions (N. J. Cooke, J. C. Gorman, Myers, and Duran 2013).

Another perspective is provided through the Agents and Artifacts approach to software engineering of MAS. This approach adopts the idea that agents operate in an environment composed of artifacts (Ricci, Viroli, and Omicini 2007). This approach is used in CARTAGO (Common ARTifact infrastructure for AGent Open environment). JaCaMo+ also uses artifacts external to agents to store protocols for interaction as well as keep records of social interactions between agents (agents do not interact directly, but interact through the artifact).

Rather than use a shared artifact to represent or control shared knowledge, an alternative is to have individual beliefs that are shared and kept consistent by agreed update processes. The update processes can involve individual interaction between agents either directly or indirectly. When agents work together in a coalition, it is possible to broadcast interests to other agents in the group to establish information needs and enable mutual support (Siebra and Tate 2005). When the team structure is dynamic, or distributed, broadcasting to all agents may not be appropriate. The appropriate selective sharing of knowledge relies upon an ability to recognise when knowledge is relevant to others. Explicit commitments or contracts created between agents are a mechanism for agents to articulate agreed update processes. This would seem to give agents some dynamism and opportunities for awareness.

Regardless of the mechanism used, it is important for the organisation meta-model to include a process to ensure that agents all share knowledge appropriately with others. The knowledge that is shared must be kept consistent and should include situational knowledge as well as organisational knowledge. It is also important that the components used in the organisation structure at design time allow agents to behave with some flexibility to adapt or improvise at run time.

3.7 Gaps and limitations in existing models

The MAS literature is rich in terms of concepts for teamwork and organisations of agents. Tambe and colleagues (Tambe 1997) argued for a generic teamwork model as a way to reduce the burden on the agent designer in the development of complex coordi-

native scenarios. Others have contributed to the proposal of organisation meta-models and features in a selection of these have been examined. Motivated to find components that must be present to address improvised coordination requirements, it is necessary to provide a design that can be adapted dynamically, in order to achieve effective coordination. This must occur in situations where not only the external environment changes but the way agents form an organisation is organic.

None of the agent organisation based meta-models examined enables agents to dynamically and explicitly coordinate sharing knowledge and plans, and demonstrate some run time emergent improvisation regarding individual agent behaviour.

The level of organisational awareness needed by agents in organisations in order to coordinate effectively needs further investigation. In disaster management, it has been suggested that one important mechanism needed for coordination is improvisation and anticipatory organisation (Smith and Dowell 2000). It is speculated that agents that are capable of initiative and anticipation in terms of their coordination, need to be self-aware and aware of others in the team. This would enable helpful anticipatory behaviour and improvisation. Our requirements lead to the need for an adaptive organisational structure within which agents can maintain consistent knowledge of the problem context as well as meta-organisational elements such as a set of roles, goals, plans and agent members. Potentially, these are all dynamic. The situation demands an ability to dynamically reallocate tasks to agents or have agents adjust their individual goals to achieve desired organisational outcomes.

Existing meta-models provide features that address some of our requirements. SharedPlans provides an intention based logic for establishing commitments to mutual knowledge of plans within a team. OperA and OperA+ use contracts established at run time which specifies which agents are performing particular roles. OMACS provides us with a flexible organisational structure that distinguishes between goals and roles using the concept of capabilities. OMACS agents are also dynamically allocated to roles based on availability and capabilities. However, no meta-model allows for roles to be split and shared between agents in an emergent or dynamic way (although KB-ORG allows for the re-creation of a new organisational structure by splitting roles and creating new coordinator roles). Models such as OperA and SharedPlans have a top down approach to designing an organisational structure or set of alternative objectives to match a set of problem requirements. A more flexible design is sought, that can be adapted at run time to enable the agents that are available to coordinate their actions and find a solution, given their capabilities.

At a conceptual level, mutual adjustment and improvisation by agents to coordi-

nate at an operational level is considered in the description of organisationally adept agents. It is necessary to explore an approach that provides a top down design that can be flexibly adjusted by agents to allow for mutual adjustment of individual plans and agent improvisation. For example, if an agent is not available with an exact match to a predetermined role, then is it possible for an agent with the required capabilities (but in a different role) to show some initiative to 'step in' and complete the task, or for multiple agents to combine their skills and coordinate and collaborate to complete a task?

OMACS has elements of adaptability in terms of separately defining agent capabilities, however coordination in OMACS is specified at design time. In OMACS, the coordination required between agents working together on a goal, is established by 'hardwiring' goal precedence into the goal model and using explicit coordination goals. Explicit deliberation on coordination is not possible. This is a significant reason why OMACS cannot be directly used to address our requirements. A similar approach to coordination is used in OperA that relies on coordination being specified implicitly in scripts at design time.

In OMACS, it is possible to define a problem in terms of a flexible decomposition tree of sub-goals involving alternative goal plans that could be adopted (S. DeLoach and M. Miller 2010). This allows for some flexibility at run time. However, dynamic and improvised coordination between agents sharing a goal is not provided.

SharedPlans provides mechanisms for agents to manage coordination of knowledge specific to plans, however, it needs further extension to include coordination of situational knowledge observed by agents. It is therefore or interest to combine these features into a new meta-model for organisations that will have the desired flexibility in terms of agents run time behaviour. The consideration of the meta-model must also be positioned in terms of a methodology for designing a MAS system. It is important that the design process is also conducted with consideration given to the need for run time adaptation and flexibility.

3.8 Conclusion

In this chapter, an overview of background research in organisational modelling and design from human and agent perspectives was presented. Issues relating to coordination, improvisation and adaptation are introduced and key organisation meta-models are identified that offer some features to address some of these issues. The limitations in these models are highlighted, regarding providing agents with awareness, ability to

mutually adjust, improvise and coordinate at run time.

Flexibility at run time can be designed by the designer by using components that keep the goals and the roles separate (functional specification vs structural specification). Additionally, it is valuable to have a mechanism for agents at run time to select a goal pathway that matches with available agents and if necessary dynamically change the way agents are allocated to (or adopt) responsibility for tasks. Thirdly, it is important to have a framework for the specification of agent interactions so that within an organisation, knowledge is shared appropriately. It is necessary for agents to share knowledge about the situation as well as knowledge about the organisation and the current state of the organisation. It is also necessary for agents to have an ability to coordinate activity in individual plans so that they can work together to achieve organisational goals. This coordination relies on agents establishing and maintaining shared mental models and SharedPlans about their intentions. The importance of improvisation was discussed and the fact that existing models constrain agent behaviour to conform to roles and that this limits individual improvisation.

Features in existing organisation meta-models that do provide flexibility have been highlighted. In particular:

- in OMACS the goal-tree defines the organisation functional objectives separately from the roles specified in the role model. Also, capabilities are used as an abstraction to enable run time allocation of agents to goals. Policies are used to specify rules to govern agent behaviour. These include policies that must be adopted and policies that may be selectively adopted.
- in OperA contracts are made explicit between agents regarding adoption of roles;
- in JaCaMo+, agents use social commitments to express their commitments explicitly and protocols are defined to specify obligations regarding agent interactions;
- in SharedPlans, a mechanism for the creation of mutual shared mental models regarding plans is presented. This is based on creation of explicit commitments about intentions.

In subsequent chapters, a new meta-model is presented, based on combining some of the above features, to address our requirements for improvisation and flexibility in agent behaviour.

Chapter 4

Toward a Grounded Model Design

4.1 Overview

In this chapter, we consider the requirements described in Chapter 2.2.5. Using the user stories introduced in section 2.3.3, in section 4.2, we evaluate a number of existing models to compare their ability to address these requirements. We then focus on the emergency management domain as an example in Section 4.3. We introduce a case study of an incident previously analysed in terms of human behaviour. We use this case study to clarify the requirements of flexible behaviour and improvisation that are to be addressed by our meta-model. We examine people's behaviour dealing with similar requirements and analyse in detail two agent models that address some of these requirements for agents, SharedPlans (B. Grosz and Hunsberger 2006; B. Grosz and Kraus 1999) and OMACS (S. A. DeLoach 2009).

Sections 4.4 and 4.5 present an analysis of the knowledge sharing requirements illustrated in the case study introduced in in section 4.3 and in particular, how the SharedPlans formalism might address some of these requirements. In the second component of analysis in section 4.6, we identify organisation design components that would assist with our requirements. In section 4.6.1, we adopt constructs found in OMACS (introduced in section 3.5.2) and identify extensions to OMACS to suit our requirements. We highlight a number of observations throughout this chapter to inform our meta-model design. In section 4.6.2, we articulate some adopted design decisions. Our observations are listed as requirements in Figure 4.11, in section 4.7. These are requirements to be addressed by our meta-model. They are also linked to related agent stories from section 2.3.3.

We have chosen SharedPlans as it provides an intention driven formalism based on adaptive human behaviour. Agents adopt explicit intentions to fulfil a goal and also adopt intentions to share their individual intentions with others. In Section 4.4[1], we argue the need for an extension to the SharedPlans formalism required to support the sharing of knowledge about a dynamically unfolding situation, specifically: who is in the team? and who holds relevant knowledge? Our rationale for such an extension is presented based on an analysis of the coordination and communication activities amongst the disaster management (DM) team during recovery and response using a prior case study of a railway accident. We conclude that in addition to the obligations imposed by the standard SharedPlans framework regarding cultivation of intentions that a certain proposition will hold (B. Grosz and Kraus 1999), agents in complex unfolding scenarios also need *knowledge cultivation* processes. Situation awareness is very important and involves more than just collecting data, it involves understanding and interpreting information. Situation awareness involves knowledge relating to the dynamic state of the environment but does not include static knowledge such as policies and rules (M.R. Endsley 1995). It is this dynamic knowledge that must be appropriately shared within the agent organisation. Knowledge cultivation processes involve reasoning about the dynamic organisation structure and the changing world state and taking actions to ensure these beliefs are shared appropriately. We briefly express the requirements of knowledge cultivation as obligations that could be imposed on agents. We argue that in order to facilitate appropriate knowledge cultivation, agents need access to explicit models of organisational knowledge. This knowledge encapsulates the relational structure of the team, along with shared beliefs, goals and plans. We briefly present a formal representation of the organisation model in order to clearly identify the rich information needed in an adaptive organisation.

OMACS provides an organisation meta-model facilitating adaptation and flexibility at run time, using agent capabilities to select goals and using policies to specify behaviour. We highlight the beneficial features offered within OMACS and also its limitations with reference to the case study example addressing our particular requirements. In Section 4.6, we highlight the limitations of OMACS relating to our requirements for adaptive and flexible *coordinated* behaviour with particular emphasis on goal or role

[1]The motivation for knowledge cultivation between agents with reference to extending existing approaches using SharedPlans as discussed in Chapter 4 was presented at the Organised Adaptation in Multi-Agent Systems OAMAS08 workshop at AAMAS in 2008 K. Keogh, E. A. Sonenberg, and W. Smith (May 2008). "Coordination in Emergent Organisations: Extending SharedPlans with Knowledge Cultivation". In: *Proceedings of International workshop on Organised Adaptation in Multi-Agent Systems (OAMAS08)* and led to a publication in 2009 K. Keogh, E. A. Sonenberg, and W. Smith (2009). "Coordination in Adaptive Organisations: Extending SharedPlans with Knowledge Cultivation". In: *Post conference proceedings- Organised Adaptation in Multi-Agent Systems (OAMAS08), LNAI 5368*. Vol. 5368. Springer, pp. 90–107.

sharing and allowing for agents to accept multiple perhaps competing roles. We make some observations highlighting meta-model constructs to address our requirements for agent improvisation and flexible behaviour.

DM involves the establishment and maintenance of a *management system* as well as directing and controlling operational tasks. The management system is the dynamic group of agents and resources engaged with sharing the goal of resolving the disaster. Following Smith and Dowell (Smith and Dowell 2000), we define DM coordination as "the resolution of interdependencies between the activities of the disparate resources of the incident organisation". An *organisation* is a set of actors, with a social order that work toward a common goal. It is suggested that the organisation structure defines roles and enables coordination (V. Dignum and Tick 2007). In this chapter, we work toward defining how the organisation structure should be modelled.

4.2 Critical evaluation of existing meta-models with reference to requirements

The purpose of this section is to critically analyse a number of existing meta-models and frameworks[2]. The requirements of dynamic, flexible behaviour are used to compare and evaluate each meta-model. This evaluation contributes to the justification for development of a meta-model to address our requirements.

We refer to the same emergency management scenario first introduced in section 1.2.1 and described as a simulation in section 2.3.2 then further elaborated upon in section 5.8. Using this emergency management simulation scenario as a focus, we examine how a number of comparison models could be used to design a MAS addressing particular use cases. This provides additional details to support the claim in section 3.7 that no existing meta-model will individually, directly support agents to perform with the flexibility to improvise and coordinate at run time.

We compare features found in each of OMACS (S. DeLoach and M. Miller 2010), JaCaMo+ (Baldoni, Baroglio, Capuzzimati, and Micalizio 2018a), OperA+ (V. Dignum 2004; Jiang, V. Dignum, and Tan 2011) and SharedPlans (B. Grosz and Kraus 1999). These meta-models each offer features to provide flexibility or coordination between agents: OMACS, SharedPlans, OperA+. In addition we compare with JaCaMo+ as it has been proposed more recently as a framework addressing broader requirements of MAS in distributed, open systems. The rescue use case scenario introduced in Sec-

[2]There are both meta-models and frameworks (comprising multiple meta-models) being compared in this chapter. For simplicity, they are all referred to as meta-models.

tion 2.3.2 highlights requirements for coordination and improvisation between agents. We reference these requirements to provide an overview of the requirements that would be met in a MAS system, built based on each meta-model. We do not provide a detailed design based on each meta-model, rather we highlight where the requirements challenge each approach.

These meta-models were chosen because they have each address some element of agent adaptability in a dynamic situation. These features include dynamic allocation of agents to tasks, using policies to guide agent behavior, using artifacts and commitments to define interactions, using contracts to define responsibilities, and creating dynamic plans.

4.2.1 OMACS meta-model

OMACS (S. A. DeLoach 2009) provides an organisation meta-model facilitating adaptation and flexibility at run time, using agent capabilities to select goals and using policies to specify behaviour and is described in section 3.5.2.

An OMACS organisation defines the behaviour of the agents (S. A. DeLoach 2009). The OMACS organisation refers to a structure, the agents populate the organisation structure and play roles in the organisation, but the organisation is external to the agent. Agents do not constitute the organisation. Flexibility in agent allocation is gained with OMACS because of the use of capabilities. Agents possess capabilities and roles require capabilities. Roles are responsible for achieving goals and agents are allocated to roles based on their capabilities.

Coordination of agents working together in OMACS is implicit in the programming of the agent to achieve its capability. An organisation can be considered to fulfill a role in a super organisation if the organisation has the capability to do so. This requires that within the organisation, individuals follow a predefined plan design chosen for roles at design time.

In Section 3.5.2, coordination and flexibility offered in the OMACS meta-model is described. The OMACS model separates the functional specification in the goal tree from the organisation structure and roles, however it does rely on a direct mapping from a goal to a role. The use of capabilities enables dynamic reallocation of agents to roles and introduces run time adaptability, however agents must possess all required capabilities defined for a role and it is assumed that one agent will be capable of achieving one goal. So, agents cannot share goal responsibilities. Using a goal tree provides flexibility at run time by defining potentially multiple solution paths at design

time. The run time goal selection can be dynamic and alternative paths can be selected if one pathway of goals cannot be satisfied, however the goal tree model does not allow for the specification of interdependent goals or explicit dynamic coordination between agents to share work toward achieving a goal. Goal dependency is represented within the goal model and agent interactions are defined at an agent level. This coordination is defined at design time.

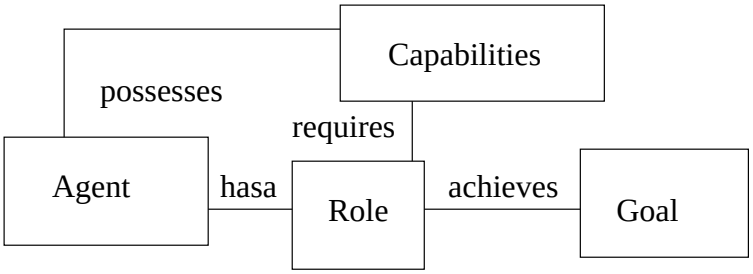


Figure 4.1: Abstractions for agent-goal relationships in OMACS

Figure 4.1 shows an abstract high level view of the relationships between agents, roles, tasks and goals in an OMACS organisation (S. A. DeLoach 2009).

The Role Model in OMACS is fixed as a set of relationships between predefined roles (represented as 'Role' in Figure 4.1) that align to meet goals. This assumes that one role will achieve one goal and that it will be possible to find an agent possessing all the capabilities needed for a role.

Figure 4.2 summarises how OMACS could address the requirements specified by the agent stories in section 2.3.3.

The OMACS meta-model is an organisation model and does explicitly:

- identify goals - individual or organisation goals
- allow for multiple solution paths to be defined at design time to enable adaptation at run time
- identify who is in the organisation
- capture information about the structure of the organisation including roles and responsibilities
- specify capabilities of agents and which agents can act in roles and which roles can achieve goals
- specify how agents are allocated (by the system) to tasks or roles

OMACS does not explicitly provide mechanisms to:

- create organisations at run time
- obligate agents to share information regarding the organisation structure
- obligate agents to share (update) information regarding beliefs (e.g. location of

Agent User Story	Ability to address this requirement
1 adopt goals, work with others	if coordination is hard coded into agent actions that are tied to roles allocated to agents
2 be aware of others and mutually adjust to fit in with others	no ability to adjust on the fly, need coordinated actions hard coded at design time
3 prioritise organisation goals	can be achieved by design of roles
4 rescue injured	with hard coded action plans matching roles to assign to agents
5 coordinate to perform stretcher rescue	coordination must be hard coded in goal design and action plan scripts that agents use to enact a given role
6 perform tasks autonomously	agents perform actions based on accepting a role allocation, roles define how agents achieve goals, at design time can specify multiple goal paths
7 help others if I can	agents are allocated roles based on capabilities, agent must possess all necessary capabilities, no room for individual improvisation
8 share information about my intentions and actions	needs to be hard coded into actions or agent behavioural policies at design time

Figure 4.2: Agent User Story Requirements addressed by OMACS

injured, completed rescues)

- identify which agents need to be informed of updates regarding the situation and regarding an agent's intentions
- allow multiple agents to dynamically plan and commit to interdependent coordinated plans to achieve one goal
- allow multiple agents to commit to incomplete plans
- allow multiple agents to revise plans
- allow agents to improvise beyond the organisation design (for example, help out with a task outside of role)

4.2.2 JaCaMo+

Figure 4.3 summarises how JaCaMo+ could address the requirements specified by the agent stories in section 2.3.3. Limitations are that in JaCaMo+, agents are not fully 'aware', the system (external to the agents) drives agent behaviour and agents do not have mechanisms for improvisation or dynamic coordination of individual plans at run time.

The JaCaMo+ meta-model is an framework including an organisation model and does explicitly:

- identify goals - individual or organisation goals (expressed as missions)
- allow for multiple solution paths to be defined at design time to enable adaptation

Agent User Story	Ability to address this requirement
1 adopt goals, work with others	based on hard coded roles and role protocols defined at design time
2 be aware of others and mutually adjust to fit in with others	shared artifact enables awareness of others' commitments, role protocols hard coded
3 prioritise organisation goals	hard coded into role protocols specifying commitments
4 rescue injured	yes, following predefined role protocol and agent plans to achieve commitments
5 coordinate to perform stretcher rescue	yes, following predefined role protocol and creating commitments between agents
6 perform tasks autonomously	yes, following predefined plan
7 help others if I can	only if programmed into agent actions coded in an interaction protocol valid for a role being enacted
8 share information about my intentions and actions	stored in centrally shared artifact, the artifact notifies registered agents of updates

Figure 4.3: Agent User Story Requirements addressed by JaCaMo+

at run time (by providing multiple commitment protocols)

- identify who is in the organisation
- capture information about the structure of the organisation including roles and responsibilities
- specify which agents can act in roles to achieve goals (but not explicitly using separate capabilities)
- specify how agents are allocated (by the designer) to roles, roles specify what actions agents may create a commitment to achieve
- allow multiple agents to follow a commitment protocol and commit to interdependent coordinated plans to achieve one goal
- allow agents to withdraw from a commitment
- obligate agents to share (update) information regarding beliefs (e.g. location of injured, completed rescues) by defining these in role protocols

JaCaMo+ does not explicitly provide mechanisms to:

- create organisations at run time
- obligate agents to share information regarding the organisation structure (this information is embedded implicitly at design time in role protocols)
- identify which agents need to be informed of updates regarding the situation and regarding an agent's intentions (in JaCaMo+, agents all have access to the inspectable artifact representing the social state of all commitments between agents).
- allow multiple agents to commit to incomplete plans

- allow multiple agents to revise plans
- allow agents to improvise beyond the organisation design (for example, help out with a task outside of role)

4.2.3 OperA+

Figure 4.4 summarises how OperA+ could address the requirements specified by the agent stories in section 2.3.3.

Agent User Story	Ability to address this requirement
1 adopt goals, work with others	defined in scripts enacted in roles
2 be aware of others and mutually adjust to fit in with others	no, only following predefined scripts
3 prioritise organisation goals	central gatekeeper would look after this in allocation of roles
4 rescue injured	based on enacting role
5 coordinate to perform stretcher rescue	coordination needs to be predefined at design time in scripts
6 perform tasks autonomously	based on actions defined for role
7 help others if I can	no improvisation, only enact roles allocated
8 share information about my intentions and actions	only if specified in a script

Figure 4.4: Agent User Story Requirements addressed by OperA+

Organisation aware agents deliberate and reason to prioritise individual as well as organisational goals, however in OperA+, agents do not have an individual identity outside of a role being enacted. In opera, the interaction structures define protocols for interaction between roles according to scripts, agents are constrained to behave according to the pre-scripted interactions aligned to roles. The interaction layer is external to agents. This limits agent autonomy, improvisation and does not provide for dynamic coordination of plans (outside of a script).

The OperA+ meta-model is a framework including an organisation model and does explicitly:

- identify goals - individual or organisation goals (as landmark states to be achieved)
- allocate agents to potential roles at design time, roles are implicitly tied to achieving landmark states
- allow for multiple solution paths to be defined at design time to enable adaptation at run time (by providing scene scripts and interaction models for each anticipated context)
- identify who is in the organisation
- define the structure of the organisation using roles

- use an agent in the role of gatekeeper to specify which agents are allocated to act in roles to achieve goals
- allow multiple agents to follow a prescriptive scene interaction script to achieve a particular landmark state

OperA+ does not explicitly provide mechanisms to:

- create organisations at run time
- obligate agents to share information regarding the organisation structure (this information is embedded at design time in scripts)
- identify which agents need to be informed of updates regarding the situation and regarding an agent's intentions
- run time dynamic coordination of individual agent plans
- allow multiple agents to commit to incomplete plans
- allow multiple agents to revise plans
- allow agents to withdraw from a commitment
- obligate agents to share (update) information regarding beliefs (e.g. location of injured, completed rescues) by defining these in role protocols
- allow agents to improvise beyond the organisation design (for example, help out with a task outside of role)

4.2.4 SharedPlans

The SharedPlans framework offers a form of adaptive dynamic team planning (see section 3.5.5). In Section 4.4 analysis and discussion outlining the limitation of SharedPlans is provided.The SharedPlans theory describes explicit group obligations regarding the cultivation of shared group intentions and dynamic plans. Each individual is obliged to ensure that intentions are shared and kept consistent by mutual agreement, represented as commitments.Team Situation Awareness and Shared Mental Models of the task and team are not made explicit in SharedPlans, although they are important in enabling adaptation in human teams (Burke, Stagl, Salas, Pierce, and Kendall 2006). The SharedPlans formalism focuses on knowledge related to decision-making for plans, and although intentions-that in SharedPlans could be employed to address a group's commitment to maintaining accurate and up-to-date shared beliefs about the environment (e.g. situational awareness or availability of agents), this has not been explicitly addressed in SharedPlans. SharedPlans does not explicitly address sharing of knowledge - regarding the team/organisation structure or the situation.

Figure 4.5 summarises how SharedPlans could address the requirements specified by the agent stories in section 2.3.3.

Agent User Story	Ability to address this requirement
1 adopt goals, work with others	hard code agent goal adoption preferences
2 be aware of others and mutually adjust to fit in with others	yes dynamic planning possible, if design includes hardcoding to give agent awareness of others
3 prioritise organisation goals	code goal priorities in individual agents
4 rescue injured	yes design individual plans to achieve goals
5 coordinate to perform stretcher rescue	yes dynamic planning with others is possible, given possible plan details at design time
6 perform tasks autonomously	yes given action plans
7 help others if I can	if it is hard coded into my reasoning at design time, no improvisation
8 share information about my intentions and actions	if it is planned and coded at design time as part of my action plans (recipes) that achieve a goal

Figure 4.5: Agent User Story Requirements addressed by SharedPlans

Using explicit commitments and mutual beliefs, the SharedPlans formalism does explicitly:

- specify groups of agents at design time and relate these groups to actions they can bring about
- allow multiple agents to dynamically plan and commit to interdependent coordinated plans (recipes) to achieve a shared goal
- allow multiple agents to commit to incomplete plans
- allow multiple agents to communicate and revise plans
- allow for specification of intention-that rules at design time in plan recipes to obligate agents to communicate beliefs

The SharedPlans formalism is not an organisation model and does *not* explicitly:

- identify goals - individual or organisation goals
- identify who is in the organisation
- capture information about the structure of the organisation including roles and responsibilities
- obligate agents to share information regarding the organisation structure
- obligate agents to share (update) information regarding beliefs (e.g. location of injured, completed rescues)
- identify which agents need to be informed of updates regarding the situation and regarding an agent's intentions
- specify capabilities of agents and which agents can act to achieve goals
- specify how agents are allocated to tasks or roles (or how agents would self-select their own goals to adopt)

Due to this, agents in a MAS built based on the SharedPlans formalism would

do well with dynamic behaviour and adjusting individual goals to fit in with other agents (requirements 2 and 5, section 2.3.3), however, the agents would rely on the specification of agent groups at design time in order to know which other agents are available and interested to work together. Additionally, the agents will not necessarily share beliefs or intentions unless this is hard coded into the process of negotiating a particular SharedPlan. These features will need to be explicitly programmed into the solution specified at design time. At design time, specify agent goals, agent capabilities and responsibilities - in terms of goal adoption and obligations to create intentions-that to obligate agents to share beliefs and organisation structure (e.g. roles adopted). At design time, intentions-that rules would be created to obligate agents to maintain consistent shared beliefs regarding the environment, location and rescue status of injured. In SharedPlans, *intention cultivation* obligations define the requirements for agents to establish shared mutual intentions.

In the next section, we develop a detailed understanding of the requirements of coordination in complex and dynamic domains such as disaster and emergency management. We hope this analysis will inform believable and predictable agent behaviour which is important if agents work with people (Klein, P. J. Feltovich, J. M. Bradshaw, and D. D. Woods 2004). We present a model of the components that comprise an adaptive organisation and express these using an extension to an existing formalism. This formal organisation model provides a language for describing the adaptive organisations in our case study. In subsequent chapters, we use these requirements, elicited based on human experience as inspiration for the design of a meta-model for artificial agents.

4.3 Case study - Ais Gill train accident

In this section, we describe the scenario of a train accident based on previous work by Smith and Dowell (Smith and Dowell 2000). Our analysis is focusing on identifying beliefs and knowledge emerging about the situation and the associated communication to share this. We highlight the need for appropriate sharing of beliefs across a distributed team in order to successfully coordinate a response. This case study motivates our proposal in section 4.4. Throughout this section and subsequent sections, we highlight a number of *observations* based on the behaviour of people that we can use to articulate requirements to inform a design for our meta-model.

4.3.1 Overview of accident scenario

At 18:49, 31st January, 1995, UK Emergency Services were notified that a train had become derailed somewhere between Kirkby Steven and Blea Moor in the county of Cumbria. Six minutes later, a second train crashed into the derailed train resulting in escalation of the incident. A train conductor was killed, 6 passengers and a train driver were seriously injured. A significant period of time elapsed (about an hour) during which the exact location of the train crash was not clear. Initially, the emergency services were unaware that the second train had crashed into the derailed train and the number of injured was thought only to be 2. A number of agencies were involved in the response: Ambulance, Fire, Police, Volunteer Mountain Rescue, and Railtrack. The accident site was not easily accessed by road and the first to arrive on the scene were fire crews 30 minutes after the accident. When ambulance crew from Brough arrived (55 minutes after the initial accident) it became known that there were closer to 30 injured people and one deceased. It took a further hour after the location and number of injured was known, before decisions regarding how to transport the injured to hospital were finally settled.

The seriously injured driver was carried along the train track to meet an ambulance at a road bridge. It was raining and this proved hazardous. Earlier, a request was made inquiring about the possibility of sending a rescue train to transport the injured via rail to Carlisle train station, to be met by ambulances. This emerged into a reality, and so other injured people were kept dry in the train waiting for the rescue train. Communication difficulties occurred with the inappropriate dissemination of this decision: ambulances were not informed in a timely way and continued en route to the disaster site rather than being redirected to Carlisle railway station. There was also confusion regarding the eventual destination of the injured, so one hospital remained on standby longer than necessary.

4.3.2 Development of situational knowledge regarding the incident

As with many disaster situations, the information regarding details of the incident was not clear initially and were established with difficulty over an extended period. In a DM team, the people involved share obligations to pass on relevant information to others in the team. As the actual situation changed at Ais Gill, beliefs were shared and revised. This is 'fill in the blanks' type of coordination - establishing uncertain facts such as location and number of injured. During this first phase, the team focus is information gathering and situation awareness (Burke, Stagl, Salas, Pierce, and Kendall 2006). Information is shared according to protocols and obligations to keep others in

the disaster recovery system informed and up-to-date. Some incorrect conjectures were made whilst details were uncertain. It was assumed that there wouldn't have been many passengers on the train, so the number of injured passengers was (incorrectly) assumed to be small.

Communication was significantly constrained. Firstly, in the hilly terrain there was disruption to radio. Second, ambiguity and uncertainty about the situation made it difficult to know what to communicate. Third, while many agencies are in the process of building up their response, each individual person knows little about who to communicate with. The difficulties and errors in communication highlight these challenges in such a dynamic situation. Passing on relevant information to others in the system is crucial. The fire crew, arriving at 19:25 passed on accurate location information "1 mile north of Ais Gill" to the fire control, who passed it to railtrack, who passed it to ambulance control centre in Carlisle, who then passed it to the Brough ambulance crew with instructions to change course en route to the site. Examining this communication with the eyes of a potential designer of a response system including artificial agents, we conclude that agents engaged in such a scenario need obligations to follow similar protocols to humans: to pass on new or revised situational information such as the accident location and number of injured. This situational information is an important component of the Shared Mental Model (SMM) of the people involved in the disaster recovery effort.

Observation 1: The agents in an adaptive team need obligations defined to pass on new or revised information about the disaster situation to relevant others in the system. This implies that agents need to have a mechanism to judge the relevance of information to pass on. This is complicated by the continuous nature of change in the task and that relevance is relative to the recipients' context - knowledge and experience that the sender doesn't necessarily possess.

Creating an associated obligation and explicit commitment to agreed knowledge updates would enable agent responsibility regarding communication of knowledge to other agents to be formally represented. Representing relevancy of information or representing a group of relevance is important to enable agents the capability to know 'who to tell' if a plan is changed or new or revised knowledge learnt. To recognise relevance, an agent needs a representation of their own and others' focus (So and E. A. Sonenberg 2007). It is beneficial for agents to create explicit representations connecting relevant tasks to help with identification of relevance to aid information sharing (Ofra, B. Grosz, and K. Z. Gajos 2016).

4.3.3 Coordinating interdependent dynamic goals

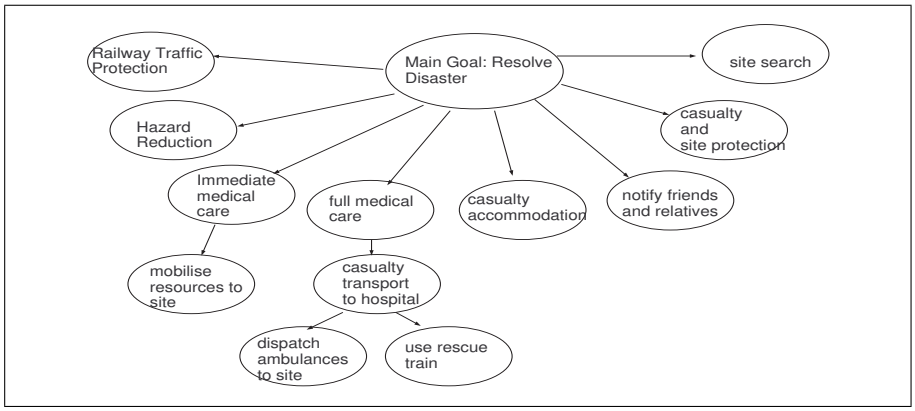


Figure 4.6: Initial goal decomposition

It could be said that the people involved were all motivated by the high level goal to resolve the disaster, enacting a sub-goal to mobilize resources to the site. Control centres off site have responsibility for strategic and planning goals. Unknown parameters such as exact location and number of injured need to be established. Based on an analysis of this incident (Smith and Dowell 2000), we can suggest a potential partial hierarchy of goals (Figure 4.6) that represent the system early in the response. This is based on the high level incident plan and a default allocation of roles and responsibilities in any incident (Figure 4.7). Each agency would likely operate following generic recipes that are initially partial and then elaborated and verified as the situation unfolds. In this case, there was the need to maintain multiple possible goals and to make (and enact) multiple, possible alternative (partial) plans based on available information. It is not practical to wait and establish facts in isolation before any action is taken, so it is better to enact multiple options until it becomes clear which will be fully enacted. Plans are revised, elaborated or dropped as more details are established as will be discussed in section 4.3.6.

The high level incident plan for the disaster response in Figure 4.7 shows the broad allocation of responsibilities to each agency (Smith and Dowell 2000). Sometimes these goals might be enacted autonomously, though in most cases, interdependencies require collaboration and communication between agencies for coordination.

At 21:42 when the rescue train arrived at the accident site, for example, collaboration and coordination of interdependencies occurred between many agencies: fire, police, ambulance, railtrack and the railway company.

Recipe for **GOAL**: Save Lives and Relieve Suffering
SUBGOALS: (listed in order of priority)

subgoal	group allocated to enact this goal
1.Railway Traffic Protection	Railtrack
2.Hazard Reduction: Fire, instability	Fire and Regional Railway Company
3.Immediate Medical Care	Fire and Ambulance
4.Casualty Transport (to hospital)	Fire, Ambulance, Volunteers, RailTrack, Regional Railway Company
5.Full medical care	Hospital H
6.Casualty Accommodation	Regional Railway Company
7.Notification of friends and relatives	Police
8.Casualty and Site Protection	Fire and Police
9.Site search	Police

Figure 4.7: Generic high level incident Plan

4.3.4 Evolution of the organisation structure

The system which manages a disaster - people, resources and technologies - is not only distributed over space but its structure evolves over time. A simplified organisation network representing the system early in the response and then about 2 hours later, is shown in Figure 4.8. These diagrams focus on the 3 main agencies: Fire, Ambulance and Police and comparison indicates that the organisation structure becomes more complex during this period. There are more members involved in each organisation and roles change as the situation changes.

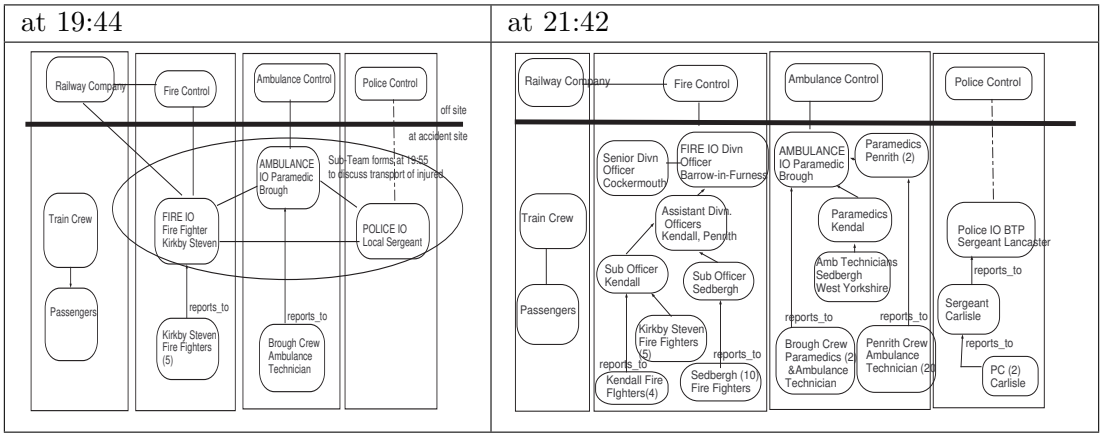


Figure 4.8: Development of organisation structures at Ais Gill

The disaster recovery system includes each of the response agencies: police, fire, ambulance, plus organisations such as railtrack, volunteers and emergency services. As the incident grew, the organisation structure changed in response to the size of the incident. The command structure develops organically. Roles of individuals are adjusted flexibly responding to the disaster with resources available (Smith and Dowell n.d.). Organic role adjustment occurs with the Fire Incident Officer (IO) role reallocated to

4 different people, driven by new arrivals at the site. At 19:20, the leading Fire Fighter from Kirkby Steven assumes the IO role. As a more senior ranking officer arrives at the scene, that person may take over as IO. At 19:35 a sub-officer from Sedbergh became IO. At 20:01 two Assistant Divisional Officers (ADOs) were present and one of these (Penrith) took over as IO, with assistance from the other (Kendal). At 20:11 the Divisional Officer from Barrow-in-Furness arrived, worked collaboratively with the 2 ADOs, then gradually took over the IO role. In some cases, the role of IO is not handed over - for example, when the Senior Divisional Officer from Cockermouth arrived at 21:00, he made it explicit that he would not be taking command from the existing IO, but would remain and observe.

As the structure of the organisation changes, the team needs to be aware of current team structures to enable appropriate adjustment and sharing of planning and situational knowledge. At Ais Gill, there were difficulties in maintaining a Shared Mental Model (SMM) of the organisation (Smith and Dowell 2000) and as will be described in the next section, this hampered communication of a change in the plan regarding transport of injured to hospital. This led to further coordination problems.

The literature confirms that team SMMs of resources, tasks and team members are essential for team processes (Burke, Stagl, Salas, Pierce, and Kendall 2006). People working in teams use SMMs to represent information about the task, resources, roles adopted by people and the people involved (Scheutz, S. A. DeLoach, and Adams 2017). These SMMs help teams to coordinate their activities more effectively when they include tracking of the goals, intentions and beliefs of others (Scheutz, S. A. DeLoach, and Adams 2017). A shared representation of the organisation is needed to enable knowledge transfer and task level SMMs (Smith and Dowell 2000).

Observation 2. In order to appropriately share relevant information, adaptive team members need to have some representation of the current DM system's organisation structure.

In complex domains, a coordination framework is needed. SMMs improve coordination. Gervits and colleagues investigated human-robot teams performance with and without explicit SMMs and found that SMM policies improve performance and efficiency (Gervits, Fong, Thurston, Pham, Thielstrom, and Scheutz 2020). They describe the SMM to include domain knowledge, agent capabilities, agent and task states, norms and obligations, activites and functional roles of agents in the team (Gervits, Fong, Thurston, Pham, Thielstrom, and Scheutz 2020). SMMs also enable coordination by providing a model for interaction. Scheutz proposes a computational framework for SMMs in artificial teams, based on human teams. Scheutz argues that the SMM

includes models of the team and task and is used to interact as well as track progress on goals so that it is essential that the SMM framework includes processes and interactions to maintain the SMMs (Scheutz, S. A. DeLoach, and Adams 2017).

Observation 3. Adaptive Team Agents need defined obligations to explicitly ***maintain*** *an appropriate SMM of the DM system's organisation structure.*

Observation 4. As the organisation is dynamically adapting structurally and members may join/leave, there is a complexity beyond a simple predetermined hierarchical structure that could control sharing of information. This requires an awareness of relevancy, linking knowledge to actors in the organisation, to enable appropriate, selective sharing.

4.3.5 Development of plans in response to the incident

During an incident, decision-making occurs to assign resources to operational or management tasks. These decisions are informed by the current incident plan and knowledge about the current situation (in agent terms, such knowledge would be referred to as agent intentions and beliefs). Both of these are changing with time as more details are known. The incident plan is initially based on generic predefined plans described at an organisational level; it is then elaborated or revised in response to the actual situation. In DM, there are typically two types of coordination: Filling in the unknown details into existing predefined default plans (e.g. location of incident, type of incident, number and type of casualties); and revising a chosen plan recipe in response to the situation, (e.g. the decision regarding the transport of injured to hospital).

Knowledge regarding the Ais Gill incident was distributed between people and the decision to use a rescue train was shared. A sub-team including the paramedic, police sergeant and leading fire fighter were partially responsible for the revised decision to use a rescue train, although it remains unclear when this proposition became an actual decision. They queried if a rescue train might be possible at 20:00. At 20:10 the reply, without further communication or collaboration was 'ETA train 60 minutes and also coach from Robinson of Appleby is mobile'.

This revised plan was not shared with ambulances off site, resulting in them continuing to travel toward the roadside near the accident where they could not be useful. The ambulance controller in Carlisle didn't know about the rescue train until 20:55. During the 45 minutes prior to 20:55, ambulances and patient transport services were dispatched toward the site where they weren't needed. This illustrates the importance of keeping relevant others informed of new/revised plans. The communication regard-

ing the decision and adopting of a revised plan to transport the injured to hospital using a rescue train involved a plan revision. This revised decision was not well communicated and led to coordination problems later (for example, not enough ambulances were available in the correct location at the train station when the rescue train arrived; staff at Lancaster hospital that could not have been used due to the limitations on the direction of the train were kept on standby for longer than necessary; and adequate police resources were not at Carlisle station to provide protection for the injured passengers from media present).

4.3.6 Multiple uncertain, partial and adaptive plans

Typical of disaster recovery, at Ais Gill, multiple plans were concurrently considered and partially enacted, due to uncertainty and incomplete knowledge availability. A disaster team cannot afford to do nothing until all information is known. Planning and action occur in parallel. As in this case, decisions may not be made explicitly, but may emerge as the best option and the actual selection of the final decision may not be clearly distinguished. Knowledge sharing in a human-agent team needs to account for multiple plans and may well need to provide a mechanism for recognising the certainty related to knowledge (for example: uncertain, possible, probable).

1	Establish location of accident (LA)
2a	CHOICE 2a: transport casualties to Hospital, H. via ROAD using ambulances with pick up at LA access point. [DEFAULT option assumed in initial plan]
2b	CHOICE 2b: transport casualties to Hospital, H. using a RESCUE TRAIN to railway station RS
2c	CHOICE 2c: transport casualties to Hospital, H. on foot along train track then via ambulance with pick up at LA - nearby road-bridge
	unknown parameters: LA, RS, H(either Carlisle or Lancaster)

Figure 4.9: Potential recipe for transporting injured to hospital

Assuming predefined recipes outlining actions toward goals, existed at least implicitly for the people involved as Ais Gill, we might imagine that there was a recipe for the goal: Transport Injured to Hospital. It could look something like the recipe shown in Figure 4.9. Initially, all the participants would have adopted the default plan assuming that the injured would be transported to hospital via road using ambulances. Due to particular constraints of this disaster, the plan was revised. At Ais Gill, all three choices were partially enacted, before choice 2b became the final decision. We would presume the poor communication regarding the change of plan was not due to the people concerned being unaware that others needed to know, but it could well be the case that the obligation to tell was not clearly defined. Who was responsible for telling those away from the disaster scene? It could have been presumed by the people

at the scene that others away from the scene had already been told by the railway authority. During phase 2 of the team response - plan formulation (Burke, Stagl, Salas, Pierce, and Kendall 2006), this example serves to indicate that when creating artificial agents to engage with people in coordination in such a domain, **obligations to share knowledge about (new or revised) plans need to be clearly stated**. This is in part addressed by the SharedPlans formalisation within the decision-making team, though needs extension to include explicit sharing with others in the system who may need to know. This will be discussed further in section 4.4.

Observation 5. Adaptive Agents need obligations to share knowledge about changed elements of plans to relevant others in the system of organisations.

This particular incident exemplifies some general features of DM that are important to note:

- The changing demands of a disaster situation result in instability of the incident organisation (Smith and Dowell 2000);
- Decision-making can be highly reactive due to the novelty and instability of most disasters; and
- Decisions can be made in part by following predefined rules and protocol but also a level of flexible adjustment of these plans is needed to deal with the developing nature of an incident.

During phase 3 - plan enactment (Burke, Stagl, Salas, Pierce, and Kendall 2006), further team adaptation and flexibility is needed to respond to changing plans and awareness is needed understanding uncertainty relating to multiple plans being partially enacted concurrently. At Ais Gill, the ambulance controller recalled the ambulance crews en route to site at 20:55 when he learnt of the alternative plan to use the rescue train.

The case study has demonstrated a number of features that need to be included in a meta-model to address a complex situation so that agents can reason and share consistent and sychronized SMMs. Scheutz suggests that a SMM for a team should address: consistency, reactivity, proactivity, coordination and knowledge stability (Scheutz, S. A. DeLoach, and Adams 2017); and argues that a comprehensive computational framework that addresses all of these requirements for artificial agent systems is needed (Scheutz, S. A. DeLoach, and Adams 2017). Scheutz proposes a framework including five key components:

- agent capabilities and properties;
- agent and task states - beliefs, goals and plans;

- obligations and norms governing behaviour, including negotiation of activities;
- resources;
- functional roles and responsibilities of agents.

(Scheutz, S. A. DeLoach, and Adams 2017). With these principles in mind, we look at processes and architectural models for agent organisations to adopt in order to create and maintain the appropriate SMMs in the remaining sections of this chapter.

4.4 Knowledge cultivation - extending SharedPlans

The importance of creating and maintaining a consistent SMM within a team has been established in the previous section. In this section, we focus on an analysis of SharedPlans (B. Grosz and Kraus 1999) and how they can be extended to support development of an appropriate SMM. As Tuomela (Tuomela 2006) discusses, maintaining mutual beliefs between individuals requires that each individual has beliefs about the beliefs of others in the team. In DM, due to the distributed nature of the system, one central shared artifact is not feasible, so we seek to describe *processes* to ensure sharing and maintaining of knowledge about the situation. This includes beliefs such as location of the incident, number of injured, current resources deployed etc. Knowledge cultivation as described in this section begins to describe the details of how we might achieve sharing of relevant knowledge in adaptive teams.

The importance of communicating to pass on relevant knowledge is captured in *observation 1.* It is also important that this information is maintained as mutually consistent within the organisation (Kinny, E A Sonenberg, Ljungberg, Tidhar, A S Rao, and Werner 1994; Scheutz, S. A. DeLoach, and Adams 2017).

Observation 6. Agents within the organisation need explicit processes to ensure mutual knowledge about the situation is kept consistent amongst the relevant others in the system.

We introduced SharedPlans in section 3.5.5. Grosz and Kraus have defined obligations that must be adhered to for a group of individuals to maintain shared (individual) plans around a shared goal (B. Grosz and Kraus 1999). These *intention cultivation* obligations enable dynamic, real time planning and decision-making in a team without relying on shared artifacts, but controlling the mutual knowledge by obligations regarding reasoning about intentions. Grosz and Hunsberger, (B. Grosz and Hunsberger 2006) argue that group decision-making in planning and intention updating are crucial to establishing collective intentionality for coordinating dynamically. Coordinated updates of group-related intentions are the key to modelling collective intentions.

We adopt an intention driven approach similar to SharedPlans, although our meta-model introduced in Chapter 5 and the demonstration system described in Chapter 7 are not explicit executable versions of this theory.

The SharedPlans formalism satisfies two requirements in our DM system: adaptive planning - enabling a group of agents to share decision-making dynamically; and distributed team work, such that a problem can be decomposed into sub-goals that can be allocated to independent sub-teams to autonomously enact. In DM the team involved is not fixed at the start of the problem – team structures emerge as the event unfolds.

In this section, we introduce *knowledge cultivation obligations*; and *explicit models of the organisation network* - linking agents/people and knowledge. These components are necessary in order to establish a consistent SMM.

We propose an extension to SharedPlans including three components:

- *TSMM* - a *Task level SMM* including situational knowledge of the problem;
- *RSMM* - a *Reflexive SMM* representing the structural organisation of the dynamic team (Smith and Dowell 2000); and
- *KC* - *Knowledge Cultivation* obligations: intentions to establish and maintain common shared content amongst the team.

In the remainder of this section, we define Knowledge Cultivation obligations. In Section 4.5, we describe a representation of the structural organisation for our case study at Ais Gill. This is used to demonstrate beliefs about the organisation and the plans and to illustrate how knowledge cultivation obligations impact on knowledge transfer within the organisation.

Knowledge Cultivation defines the processes involved in ensuring that agents maintain a consistent SMM. In particular, agents must maintain consistent shared representations of the task situation (TSMM) and the organisation structure (RSMM). These processes are triggered by defining obligations for agents.

We adopt the formalisms used in SharedPlans formalism (B. Grosz and Kraus 1999; Hunsberger 1999) introduced in Figure 3.4. We introduce additional predicates to informally represent elements needed in the descriptions, described in Figure 4.10.

We define SGR, the System GRoup, to represent all the agents/people who are part of the DM system at any stage during the DM response. For now, we represent any organisation network as RSMM and the state of the world at some time as TSMM (these will be more formally elaborated upon in section 4.5). The following obligations describe the processes that we impose on the SGR.

- SGR mutually believes that all members of SGR are committed to success of the goal to resolve the disaster;
- SGR mutually believes that all members of RSMM are committed to updating RSMM so that, at any time, the set of all RSMM (multiple organisations) reflects the current system structure;
- SGR mutually believes the need to update and maintain relevant knowledge about the state of the world (TSMM).

These obligations are imposed on any sub-team that forms. The SGR is made up of members of all RSMM that occur during the team disaster response.

RM(gr,t)	Reflexive Model for group gr at time t. This represents the organisational structure of group gr and identifies relationships between agents.
A Rel.to B	Agent A is Related to agent B. A and B are both linked in an existing organisational hierarchy or coordination loop - a reflexive model(RM).
Add(x,gr)	Add agent x to group gr.
Update(x,RM)	Update RM to include agent x.
Relevant(b, BS)	belief b is relevant to a set of beliefs BS.
Focus(b,GR)	Belief b is in the relevant focus of interest to an agent in group GR. This may be established by a goal that is related to this belief or a role allocated to an agent in GR that is associated with this belief

Figure 4.10: Predicate definitions

1. Awareness of system group goals

The system group represents all involved - the injured passengers, the ambulance, fire and police agencies, British Rail and volunteers. The high level shared goal is to resolve the disaster. Each agency does not work in isolation, but is aware that other agencies are involved. There are dynamic sub-teams representing each agency, Fire, Ambulance, Police, Railway Company, similar to Figure 4.8. There are also sub-teams that form between agencies such as the Incident Officers responsible for decision-making considering options for transport of the injured. Even though, undoubtedly there was an awareness of others off site, there were problems with communication of their revised plans to ambulance control centre in a timely fashion. To avoid this kind of error occurring with artificial agent teams, we need to be explicit in creating an awareness of the system group (distributed over time and space).

We adopt the modal operators from SharedPlans as introduced in Figure 3.4. This group SGR has a mutual belief (MB) that all members have an intention that to achieve α - the goal to resolve the incident.

$$MB(SGR, \forall\, g \epsilon GR_i : g \epsilon SGR \wedge Int.th(g, \alpha)) \tag{4.1}$$

2. Obligation to update RSMM as needed As Figure 4.8 indicates, the organisational network, thus RSMM representing the situation at Ais Gill is very dynamic. The communications during initial phase of establishing the details of the situation relied on appropriate notification to others in the DM system. This could only occur because there was knowledge of who else was involved. At Ais Gill, the decision-making team on the ground needed to know to share their decision regarding how to transport the injured with the ambulance controller to enable plan revision and strategic response to reallocate resources (the hospital on standby and the ambulances en route). We capture this as:

The group mutually believes that all members are committed to updating the RSMM to include new members as they join the system, where the current RSMM at time, t for group SGR is represented by RM(SGR,t).

$$MB(SGR, (\forall\, g_j \epsilon SGR, Int.th(\exists\, g_k, \exists\, g_i : \neg g_k \epsilon SGR \wedge g_i \epsilon SGR : g_k Rel.to(g_i)) \\ \Rightarrow Int.to(g_j, Add(g_k, SGR) \wedge update(g_k, RM(SGR, t)))) \quad (4.2)$$

3. Knowledge cultivation: obligations to share relevant knowledge (beliefs and intentions)

SharedPlans obligations require agents to share information about plans. Knowledge cultivation obligations broaden this to include obligations to share beliefs.

If the decision-making team was aware of the system group and obligated to share relevant intentional knowledge, such as shared decisions with those who need to know in the entire system group, as represented in the current RSMM, this would enforce appropriate communication. If the team were obligated by SharedPlans and aware to share beyond the initial decision-making team to the wider organisation system, coordination is possible. RSMM network may also be useful toward identifying potential relevancy of information to agents. The RSMM needs to be updated to include new agents as they arrive on the scene - such as the local mountain rescue volunteers who helped carry the driver out. The group also shares awareness of a relevant sharing of situational knowledge: beliefs about the state of the 'world' - TSMM. TSMM is a union of belief sets (BS) with associated relevancy/focus tags to identify relational links and links to agents who have an interest in this knowledge.

The group has a mutual belief regarding the need to update shared situational task knowledge TSMM with new relevant beliefs that are not already shared. Relevant beliefs need to be identified as such by a relationship to existing beliefs or tags to show

relevancy in the belief set, BS. Updates can also be motivated by recognising that some agents have an existing focus (interest) in these beliefs. The update intention: *Int.to(g, Add(b,BS))* will result in appropriate interactions for knowledge sharing between agents to ensure that individual and organisational belief sets remain consistent for all agents in the group. If an agent has a belief b that is relevant to the focus for the group and not in the currently maintained set of group beliefs BS, then the agent will adopt an intention to add this belief to the group belief set. Details of how the belief is added are not provided at this stage.

$$MB(SGR, \forall\, g \epsilon GR : Bel(g, b) \wedge \neg b \epsilon BS \wedge (Relevant(b, BS) \vee Focus(b, GR)) \Rightarrow Int.to(g, Add(b, BS))) \tag{4.3}$$

The knowledge cultivation obligation as defined in the above formalism specifies that all agents in the group create intentions to add organisational beliefs to their own individual belief set following updates from other agents and similarly to share new information with other agents.

In R-CAST (Fan, S. Sun, B. Sun, Airy, McNeese, and Yen 2005), when information is distributed, and proactive sharing required, there is a need to represent information relevancy so that it is possible to ascertain if a particular piece of information is relevant to another agent. In R-CAST, information needs graphs are used to associate information with context, enabling matching of knowledge to previous experiences and providing a mechanism for identifying when information might be of future relevance to another team-member (Fan, S. Sun, B. Sun, Airy, McNeese, and Yen 2005). In the DM case, shared focus may be identified by relational links between agents within a sub-team or organisation, or identifying the coordination loops (Voshell, D. D. Woods, Prue, and Fern 2007) that are present. Information sharing to enable teamwork in a dynamic environment has been supported by identifying mutual influence potential networks as part of the design process (Ofra, B. Grosz, and K. Z. Gajos 2016). In the next section, we begin to represent the structure of knowledge representing the disaster management organisation more explicitly.

4.5 Representing the organisational knowledge at Ais Gill

In this section, we formally represent the shared mental models in the organisations involved in our case study. We identify elements that define an organisation and give some examples of the RSMM organisations that exist based on the Ais Gill case study. This analysis takes a step closer to expressing an adaptive organisation system in a

computational tractable way.

Knowledge cultivation obligations define intentions to maintain these models within an organisation.

To represent each organisation involved over time during the disaster scenario, we use a formal organisation model (V. Dignum and Tick 2007): Given a set of worlds, W, an organisation O is defined in a world $w \epsilon W$ as: $O^w = \{A_o^w, \leq_o^w, D_o^w, S_o^w\}$ where $A_o^w = \{a_1, ..., a_n\}$ is the set of agents, $\{\leq_o^w\}$ is a partial order relation on A_o^w, reflecting the structure of the organisation, $D_o^w \subseteq \Phi$ is a set of objectives (states to achieve), and $S_o^w \subseteq \Phi$ is the set of current states relevant to this organisation holding at a given moment. We extend this model to include two additional components in an organisation: B_o^w is the set of current mutual beliefs of the organisation and SP_o^w is the set of current SharedPlans held by the organisation.

At Ais Gill, there were multiple organisations, with some overlapping objectives and some agents belong to more than one organisation. Each organisation, at any time, can be described using the above formalism. The SGR includes all agent sets involved in each organisation during the scenario, so SGR in this case is a union of the fire, ambulance, police and incident officer teams over time The RSMM representing each organisation is the shared model of structure $A_o^w, \leq_o^w$. The TSMM is the union of beliefs, objectives(goals) and SharedPlans: $B_o^w, D_o^w and SP_o^w$. The set of objectives (D_o^w) can be broken into two subsets - action based objectives and information needs objectives. In the case of the DM system, as discussed, it is necessary for the organisation to access networks of relevancy. Expressing our organisation model using this formalism imposes a structure through which relevance associated with each organisation can be identified.

Agents in our DM system include Fire Fighters (ff), Ambulance Paramedic (ap), Control (fc), Fire Incident Officer (fio), Ambulance Incident Officer (aio), Police Sergeant (ps). Objectives include: Transport to site (tS), Hazard Reduction (hR), Immediate Medical Care (mC), Casualty Transport (cT), Site Protection (sPr), Assess situation (aS), Share location details (shLoc). Information needs objectives include: Establish Location of Incident (getLoc), Establish Number of Injured (noInj). Each organisation that forms has a shared objective - either strategic or practical.

The Fireteam and Ambteam organisations are hierarchically based and share generic goals as shown in Figure 4.6, page 94. At Time 1, 19:20, the organisation model for the fireteam could be described as follows.

$$O^1_{oft} = \{A^1_{fireteam}, \leq^1_{oft}, D^1_{oft}, B^1_{oft}, SP^1_{oft}, S^1_{oft}\} where$$
$$A^1_{fireteam} = \{fc, fio, ff1, ff2, ff3, ff4, ff5\}. \quad (4.4)$$

$\{A^1_{fireteam}, \leq^1_{oft}\}$is a hierarchical structure:

$$fc \leq^1_{oft} fio, fio \leq^1_{oft} ff1, fio \leq^1_{oft} ff2, fio \leq^1_{oft} ff3, fio \leq^1_{oft} ff4, fio \leq^1_{oft} ff5. \quad (4.5)$$
$$D^1_{oft} = (hR, mC, cT, sPr). \quad (4.6)$$
$$B^1_{oft} = (traincrash, loc : 1mileNthAisGill, 2injured). \quad (4.7)$$
$$SP^1_{oft} = (shLoc, aS). \quad (4.8)$$
$$S^1_{oft} = (traincrash, loc : 1mileNthAisGill, 30injured, 1DOA) \quad (4.9)$$

Information was shared from the fire crew from Kirkby Steven who arrived at 19:25 to fire control to give accurate details on the location (originally thought to be Birkett Tunnel). Fire control then passed this information to Railtrack who passed it to Ambulance control and the Brough ambulance crew en route to Birkett Tunnel changed route to take B6259 to the accident site. The presence of the RSMM $\{A^1_{fireteam}, \leq^1_{oft}\}$ establishes awareness of membership of the organisation that enables this information sharing within the fireteam organisation.

At time 2, 19:55, the ambulance crew from Brough had arrived on scene. The more senior ambulance paramedic assumed the role of Ambulance IO. The ambteam organisation hadn't yet changed in structure, but TSMM begins to develop new beliefs and SharedPlans. These would be represented in new values for B and SP respectively in the model of the ambteam organisation.

The IOteam forms on site at 19:55. This comprises the Ambulance IO, Fire IO and Police Sergeant, shown circled in Figure 4.8, page 95. Members of the IOteam are in multiple organisations - at least one for their agency (Fire/Ambulance) and one for the IOteam. The IOteam generates multiple partial SharedPlans within the IOteam for how to transport the injured to hospital Multiple plans were enacted with ultimately one being chosen eventually.

Knowledge Cultivation obligations would motivate the IOteam to share this eventual plan choice with *relevant* others in SGR. In this case, each IO would be obliged to share the plan with their relevant control officer who is in their other organisation RSMM - Fireteam (oft) or Ambteam (oat). The IOteam structure is not a hierarchy, so $\leq^2_{oio}$ does not contain any ordering relations. Their SharedPlan SP^2_{oio} would reflect

Figure 4.9. The IOteam is represented as:

$$O^2_{oio} = \{A^2_{ioteam}, \leq^2_{oio}, D^2_{oio}, SP^2_{oio}, S^2_{oio}\}. \tag{4.10}$$

$$A^2_{ioteam} = \{fio, aio, ps\}. \tag{4.11}$$

$$D^2_{oio} = (PlancT). \tag{4.12}$$

$$SP^2_{oio} = (incompletePartialPlansRecipe, \tag{4.13}$$

$$eliminateOption1, \tag{4.14}$$

$$planCarryDriverAlongTrackToRoadBridge, \tag{4.15}$$

$$arrangeKendallAmbulanceToMeetAtRoadbridge). \tag{4.16}$$

$$S^2_{oio} = (conditionsDifficult). \tag{4.17}$$

At time 3, 21:45, more rescue personnel had arrived on site, the incident officer role had been handed over to more senior officers and the organisations were more complex. We show the definition for the new fireteam organisation as an example. *(ado: assistant divisional officer, do : divisional officer, ff : firefighter, -k : kendall, -s : Sedbergh, fio : fire incident officer, so: sub-officer)*

$$\begin{aligned} O^3_{oft} =& \{A^3_{fireteam}, \leq^3_{oft}, D^3_{oft}, B^3_{oft}, SP^3_{oft}, S^3_{oft}\} where \\ & A^3_{fireteam} = \{fc, fio, do, ado1, ado2, so1, so2, ff-ks\{1..5\}, ff-k\{1..4\}, ff-s\{1..10\}\}. \\ & \{fireteam, \leq^3_{oft}\} = \{fc \leq^3_{oft} fio, fio \leq^3_{oft} ado\{1,2\}, ado\{1,2\} \leq^3_{oft} so\{1,2\}, \\ & so1 \leq^3_{oft} ff-ks\{1..5\}, so1 \leq^3_{oft} ff-k, so2 \leq^3_{oft} ff-s\{1..5\},\} \end{aligned} \tag{4.18}$$

As the above examples show, the appropriate transfer of knowledge relies upon a structure for representing relevance of knowledge. Associating knowledge with relational organisational structures provides a network of linking knowledge with groups of agents and enables agents in an organisation to determine which other agents may have an interest in knowledge. As agents may belong to multiple organisations, each with a different focus, this limits the scope for broadcasting information, but enables information to be passed on to appropriate agents.

Common Ground When agents are working in a distributed organisation that is maintaining common knowledge - structural organisational knowledge as well as domain

related situation awareness, a mechanism is needed for ensuring common knowledge is held consistent. Intentional approaches such as SharedPlans (B. Grosz and Hunsberger 2006) require that agents commit to shared knowledge about plans and only update such knowledge in agreement with others. As discussed earlier in this chapter, it is useful to adopt a similar intentional approach to establishing mutual knowledge about the situation and the organisation.

One approach to establishing mutual knowledge or common ground is to enable agents to hold individual beliefs apart from the organisation's beliefs with semantics defining how the organisation's beliefs are agreed upon. An explicit process for acceptance of knowledge as common to the organisation would enable the organisation to form grounded beliefs. Grounded beliefs are those that are publicly accepted as true by all the participants (Gaudou, Herzig, and Longin 2006) and represent the view of the 'organisation' rather than of individuals per se. This approach would then require that agents have an ability to reason about organisational beliefs separately from their individual beliefs. As the environment is dynamic and the situation is changing, the possibility of inconsistency between organisational beliefs and individual beliefs would require an addition to the agent deliberation cycle to update individual beliefs to align with organisation beliefs.

We conclude that our meta-model must be supported by processes to ensure that agents within an organisational structure establish and maintain common ground using an intentional approach. This means that agents within an organisation need to have a goal to share beliefs and plan status and an obligation to adopt intentions to fulfil that goal. An agent as part of the organisation must adopt such a goal if it is stipulated as part of the policies of the organisation. At design time a decision can be made to define organisation policies. The simplest design decision in the absence of a mechanism to prioritise or assign probabilities to beliefs is to specify that organisational agents will adopt all organisational beliefs as individual beliefs. Similarly, if an agent is informed of a new or revised belief, or an adopted intention by another agent in the organisation, this information is accepted in trust and used to update individual beliefs.

Observation 7: Agents can benefit from an intention-based mechanism to maintain common ground.

Using the case study, we have established a number of agent requirements. Following the detailed presentation and analysis of requirements and critical evaluation of existing models, in the next section, we begin the design process of creation and evaluation of artifacts toward the development of a meta-model. Following the design science approach, we begin the development with reference to existing knowledge bases. In the

next two sections, our analysis continues we propose extensions based on OMACS and SharedPlans to contribute to the new meta-model.

4.6 Toward a flexible organisation meta-model

In section 4.2.1, we analysed OMACS in light of our requirements. We have identified some components in OMACS that enable agent run time flexibility. We draw attention in this section, to some ways that OMACS could be extended and articulate further observations to address our requirements. These observations are used to inform our subsequent meta-model design in Chapter 5.

4.6.1 Building on OMACS meta-model

Considering our requirements, we desire that agents have mechanisms for run time adaptation, especially in situations when the available agents don't exactly match a pre-defined solution design. We would like a meta-model that allows agents to coordinate run time behaviour and improvise if necessary rather than follow scripts exactly. The capabilities abstraction in OMACS provides flexibility in terms of run time allocation of agents to matching goals. It could be extended to give greater flexibility if goals are broken down into smaller tasks so that multiple agents could combine capabilities to share responsibility to achieve a single goal.

Observation 8: Flexibility in task adoption or allocation can be facilitated by using the capabilities abstraction to relate agents to tasks and relating a role to a list of lower level responsibilities that can be matched to tasks and sub-goals.

If available agents do not exactly match with the required capabilities, OMACS does not provide an explicit mechanisms for agent improvisation regarding role adoption or coordinating the sharing of role responsibilities between multiple agents. In OMACS, most of the information required for collaboration between roles is embedded in the goals that are instantiated (S. A. DeLoach 2009). It is presumed that if agents need to work together, the coordination requirements of the interdependent tasks performed by each agent would be implicit in the predefined ordering of goals.

OMACS agents are coordinated by individual design time plans specifying how to carry out their assigned roles and achieve goals. The plan includes communication between agents, specified at design time. In OMACS, there is not a generic coordination mechanism or framework defined within the organisation. If agents are not individually capable of completing all the activities in the plan, then the goal may not be reached.

However, if agents have an ability to coordinate and work together then it could be possible for agents to collaborate by each adopting tasks to contribute to achieve a goal.

Observation 9: An explicit coordination mechanism within the organisation could help agents manage dynamic collaboration rather than embedding implicit coordination within tasks.

Agent autonomy and improvisation The OMACS meta-model is based on the assumption that agents will accept certain limitations on their autonomy. If assigned a role to play in the organisation, in order to achieve one of the organisation's goal, the agent must agree to play that role in the organisation. OMACS assumes that when allocated to fulfil a role, an agent will accept that allocation and create an intention to fulfil it. There is no direct mechanism for autonomous agents in to adopt a varying level of commitment to a role or task and/or accept multiple roles simultaneously. This constrains the environments in which OMACS could be applied to agent systems in which agents are specialised and dedicated to a particular set of potential roles. Even if agents have some capabilities outside of their adopted role, they cannot improvise to 'help' others unless there is a particular role defined.

OMACS provides flexibility in the mapping of goals to roles, so that at design time alternative roles can be selected based on capabilities. However, OMACS does not provide an ability to individually prioritise and accept a (partial) role allocation toward a goal that is not as important to the agent, until such time as the agent is needed for a role of higher importance. If agents could do this, the opportunity for improvisation could be created.

In environments that include uncertainty and a need for responsive action in a timely way (including domains such as emergency management), the ability for agents to flexibly adopt roles/tasks and drop or suspend them if a higher need arises is valuable (Corkill, Durfee, Lesser, Zafar, and C. Zhang 2011). In this environment, plan revision and mutual adjustment to fit in with others and manage interdependencies are crucial to success. Improvisation is needed to adapt pre-existing plans, revise role descriptions and make do with existing resources in order to achieve a solution in a time critical situation.

Observation 10: Agents need an ability to prioritise the adoption of goals not only based on roles allocated but also considering the organisation's objectives as well as personal goals.

Observation 11: In a dynamic setting, agents within an organisation need permis-

sion to improvise if necessary to find a solution, based on the capabilities of available agents. This may involve an agent adopting a goal based on their capability rather than based on their role.

The OMACS system is very well designed for adaptive goal selection and reassignment, however OMACS is not designed to allow for agents to dynamically coordinate and communicate their activities (e.g. to collaborate on a joint goal) beyond a scripted plan that specifies which agent will complete each task. another approach is to use an external coordinator. The scenario presented by DeLoach (S. A. DeLoach 2009), demonstrates that when agents are no longer available and goals cannot be met according to the original goal-role-agent assignment, the OMACS system automatically reorganises and newly revised goals are selected based on the currently available agents' capabilities.

Organisational coordination in dynamic situations can be very complex as dependencies between tasks must be managed and priorities can change as the situation changes. Additional complexities are introduced if the organisational structure is allowed to evolve during the collaboration. When agents engage in joint activity, they need predictability of behaviour so that each agent can predict the behaviour of others in order to coordinate their own plans (Klein, P. J. Feltovich, J. M. Bradshaw, and D. D. Woods 2004).

We are particularly interested in agents being able to self-coordinate within a coherent organisation to work together to achieve organisational goals, similar to ORA4MAS (J. Hübner, Kitio, and Ricci 2010). Coordinated agent players need to have access to a shared knowledge base within teams (J. Hübner, Kitio, and Ricci 2010). In the approach used in the MOISEinst (Gâteau, Khadraoui, Dubois, and Boissier 2005) and ORA4MAS (J. Hübner, Kitio, and Ricci 2010) systems, each organisation is considered as a first class entity, so agents have access to the mental state of the organisation directly. However, in these systems, the first class entity does not have agent characteristics, but is an external tool or artifact that agents use. Artifacts are also used in JaCaMo+ (Baldoni, Baroglio, Capuzzimati, and Micalizio 2018a).

The meta-model we adopt needs to provide explicit knowledge to agents and organisations to give awareness of the structure of the organisation, organisational goals, current membership and allocation of responsibilities to members. Also, as members may leave and join the organisation, the interface to the organisation is best not directly linked to a particular individual member. This is a key factor in favour of agentifying organisations or having an organisation as a separate inspectable entity in the meta-model. We see a benefit if agents are able to be aware of other agents in the

'neighbourhood' even if they are not linked via a direct relationship i.e. sharing a goal or working together to achieve something as part of an organisational mission. This could be part of the environment model.

4.6.2 Meta-model design decisions

In this section, the design of the organisation meta-model, structure, social model and the relationship between roles in the organisation structure and the functional specification of a problem is considered. Motivated by the requirements, a number of design decisions adopted for the meta-model are articulated.

We first consider the use of roles as well as the relationship between roles and a functional specification. As discussed in the previous chapter, roles are useful to define relationships within an organisation and roles can be used to provide context (Odell, Van Dyke Parunak, and Fleischer 2003) for agent reasoning. We adopt that roles be used to define the set of responsibilities associated with achieving a goal, however will be defined independently of goals.

Observation 12: In order to achieve flexible and dynamic coordination between agents working together, agents need an explicit representation of the interdependencies between tasks required to achieve a goal.

Agents within an organisation need a mechanism to support sharing responsibilities and working together so that different agents can adopt individual goals that together combine to achieve one organisation goal. This relates to *observation* 9 that states that agents need an explicit coordination mechanism to enable them to work together.

We choose to define the social model in the organisation in terms of relationships between roles. We propose that the organisation defines policies that govern command, control and interactions between agents. As agents are enacting or instantiating roles, roles in our meta-model can represent a functional mapping to a list of responsibilities required to achieve a goal. This approach is consistent with the work of others (S. A. DeLoach, Oyenan, and Matson 2008; Odell, Nodine, and Levy 2005). Similar to Deloach, we adopt capabilities to enable flexible definition of skills required to enact a role, so capabilities can be used to allocate agents (or by agents to adopt roles) dynamically. If an agent leaves or loses a capability and then can no longer fulfil the requirements of a role, then that particular capability needs to be fulfilled by another agent. The capabilities abstraction enables a dynamic reallocation of agents to roles.

In order to achieve flexibility in our goal design we adopt the goal-tree used in OMACS extending this so that a goal might be broken into synchronized/ordered tasks

that could be assigned to agents. If a goal can not be achieved by and agent enacting one role, then multiple agents enacting separate roles (or perhaps agents acting outside of a role) can combine individually to achieve a goal working together. We describe goals as being composed of synchronised tasks and tasks can be performed by capable agents. So each task can be associated with a particular capability. This abstraction is introduced to enable flexible and dynamic planning by multiple agents to commit to adopt tasks and create a coordinated plan to work together to achieve a goal if necessary.

When a set of agents are engaging together to achieve a goal, explicit coordination mechanisms are needed (Sims, Corkill, and Lesser 2008). We choose to split goals into separate synchronised tasks rather than split roles, as intuitively, this abstraction fits with our observations from real life examples in the emergency management domain. For example, consider a goal to move an injured patient on a stretcher that might be achieved by three medical personnel - two in the role of 'side stretcher carriers' on the left and right, and a third in the role of 'carrier at the front', also taking responsibility for direction. The goal to move the patient can be decomposed to involve 3 interdependent carrying tasks that occur concurrently and each task is accomplished by a capable individual allocated to one of the required roles. In order to ensure that the individuals then carry the stretcher in a coordinated way, they also need a SharedPlan (or equivalent) to ensure they lift together, wait for each other and move together, in the same direction. If one the medical personnel is unavailable at the time but a nearby security officer has the capability to carry the stretcher, then it should be possible for that officer to be requested to help.

In the remainder of this section, we briefly discuss approaches for representing the problem functionally, maintaining agent awareness in the context of the organisation, and coordinating agents within the organisation. We identify further implications for the meta-model design.

Sims, Corkill, Durfee and Lesser use high level guidelines to describe constraints on how organisational objectives should be decomposed in a hierarchy. Separately, operational objectives represented as leaf goals in their goal decomposition can be operationally coordinated as required by the individuals involved in each team (Corkill, Durfee, Lesser, Zafar, and C. Zhang 2011; Sims, Corkill, and Lesser 2008). We adopt this approach to 'leave the details' to the smaller groups similarly. We consider that these smaller groups may be adhocracies - temporary ad hoc organisations with the infrastructure associated with an organisation. The organisation is expected to have

more structure than a team, that may not necessarily have a formal structure[3].

Inspired by this, we develop a design to facilitate abstract agents working together in organisations using high level plans specified at design time and allowing agents at run time to adapt to suit a particular situation. This can be achieved using Shared-Plans.

We aim to facilitate agent improvisation by providing appropriate organisational awareness so that autonomous agents can reason and work together to achieve goals.

We are interested in a situation where agents are working with some autonomy and in individual organisations, but as part of a larger organisation responsible for an overall goal to resolve a situation. We consider the organisation as a first class entity and agents in the organisation are aware of organisation goals and the composition of the organisation (e.g. organisation roles, members of the organisation). This is similar to Team Oriented Systems as described by Tidhar. The approach we adopt differs from Tidhar's work regarding the sharing of beliefs. We consider that agent beliefs are held as individual beliefs and explicitly shared by agents with direct messaging, so agents do not share a common separate artifact holding organisation beliefs.

Agents need to have a general awareness of others to avoid interference and ensure success of a high level shared goal. Representing the connections between agents using multiple organisations enables levels of abstraction or hiding of detail so that agents can have multiple individually defined contexts (associated with each organisation). Each organisation can have a detailed awareness of its own goals, plans, membership and a more general awareness of others at a higher, less detailed level of abstraction. This approach is adopted by Jiang, Dignum, Tan (Jiang, V. Dignum, and Tan 2011). The value in creating short to medium term organisations, is that for the duration of the organisation, obligations and shared organisational mental attitudes can be used to help ensure that our complex, dynamic, coordination requirements are addressed.

We adopt the idea that organisations can be created at design time and at run time and be used to define contextual boundaries for agent reasoning and agent interactions. We adopt the name Organisational Adhocracy (D. Mendonca, Jefferson, and Harrald 2007) to refer to organisations.

Collective Obligations have been used to collectively represent a group of agents who share responsibility for a particular obligation. Using policies, the obligations are then mapped to individual obligations on agents (Van Diggelen, J. Bradshaw, Johnson, Uszok, and P. Feltovich 2009). Whatever the term used, it is necessary to place a

[3]A team could of course also have infrastructure such as roles and responsibilities, obligations and norms however an organisation must have these

boundary around the organisational social context within which social policies and commitments are defined. In order to achieve agent coordination, we choose to adopt a notation for social commitments as described by Singh with social policies notation suggested by Carabelea (Carabelea and Boissier 2006).

Dignum previously used social contracts with landmarks to define agent interactions (Weigand, V. Dignum, J.-J. Meyer, and F. Dignum 2003). In OperA+ (Jiang, V. Dignum, and Tan 2011), there are three distinct models: organisational, social and interaction. One difference between our approach and theirs is our decision to decompose functionality as a goal and task hierarchy rather than as a set of role descriptions. We chose to do this to address the dynamic coordination of individual as well as organisational plans. Social contracts specify interaction rights and obligations. Using social semantics such as commitments provides for flexibility so that agents do not have to follow fixed scripts defining interactions, but adhere to predefined interaction constraints. Commitments are state-based constructs, so can be defined based on conditional events and may be dynamically created, deleted, assigned, cancelled, suspended etc. Using commitments enables agents (and designers) to clearly define obligations.

4.7 Conclusion

We target an organisational meta-model enabling both reallocation of agents to tasks and dynamic goal decomposition and achievement. We have examined SharedPlans (B. Grosz and Hunsberger 2006) in the context of human coordination in the emergency management domain and highlighted that agents need to ensure that they cultivate knowledge about their organisational structure as well as domain knowledge - plans and situation awareness. These requirements are not unique to the emergency management domain and are relevant to many emergent situations where agents initially form or enlist in an organisation with a common goal, then within that organisation, smaller organisational groups form to autonomously work on distributed, but possibly interdependent sub goals. The issue to highlight is that each organisation needs to be aware and coordinated within the organisation and across any encompassing organisation. We are particularly interested in awareness and coordination of knowledge and behaviour between, across and within organisations.

We have described the complex demands of DM and proposed an extension to SharedPlans that offers a step toward meeting these demands. SharedPlans provides a framework for sharing and maintaining a team mental model of decision-making and intentions but does not account for changing team structures. We have used an exist-

Requirement	Observation	Specific Requirement *(Agents must ..)*	Implication for meta-model design	Related agent stories
Share with relevant others		*Share information appropriately*		
	1	pass on new or revised situational information to *relevant* others in the system	organisation needs to identify relevant other agents, agents need obligations to share	1, 8
	5	selectively share knowledge about developing plans to relevant others in the system	need policy/norm to create obligations for sharing intentions	8
	6	adopt explicit processes to ensure consistent mutual situational knowledge among agents	need policy/norm to create obligations for sharing beliefs	8
Maintain awareness		*Create shared mental models of situation, intentions and organisation*		
	2	be aware of the current DM system's organisation structure in order to find relevant others	agents need organisation as a first class entity	8
	4	maintain awareness of others and link information needs to relevant others	agents need to identify relevance and relate information to appropriate others	8
	3	maintain shared mental model of the organisation.	agents need to know membership of organisation	2, 8
Coordinate with others		*Coordinate plans, mutually adjust to fit in, prioritise org goals*		
	7	adopt intention based mechanisms to maintain common ground	need explicit representation of and sharing of intentions	7
	8	behave with flexibility in task adoption or allocation using individual agent capabilities as well as roles	need flexibility in representing tasks and aligning to capabilities	4
	9	use an explicit coordination mechanism rather than embed coordination in agent actions	need coordination mechanism to mutually adjust and negotiate plans at run time	2, 5
	10	prioritise the adoption of goals not only based on roles allocated but also considering the organisation's objectives and personal goals.	agent reasoning must consider organisation objectives	1, 3, 6
Improvise if necessary to find a solution		*Act outside role responsibilities, use capabilities to achieve tasks*		
	11	improvise if necessary to find a solution, based on the capabilities of available agents outside roles	need mechanism to allow agents to act outside of role	7
	12	explicitly represent the interdependencies between tasks required to achieve a goal	need breakdown of goals and relationships between tasks	2

Figure 4.11: Requirements drawn from analysis, related to agent stories (section 2.3.3)

ing formalism to represent the additional requirements for an adaptive organisational system.

The contribution in this chapter is to clarify our requirements based on a realistic case study, to evaluate existing models and make some preliminary design decisions regarding components needed in our meta-model to address the requirements.

Based on this analysis, we introduced knowledge cultivation processes to complement the intention cultivation processes already in SharedPlans and highlighted that the OMACS organisation meta-model provides some flexibility for modelling MAS but is not designed to address all of these requirements. We have articulated a number of requirements, in terms of agent behaviour and their implications in terms of the design of a meta-model. These observations and design implications are listed in Figure 4.11. The meta-model requirements are also linked to related agent stories from section 2.3.3. In Chapter 5, we develop these ideas further to propose a meta-model addressing the complex requirements we have identified.

Chapter 5

OJAzzIC Meta-Model for organisation and agents

5.1 Overview

In this chapter, we present the rationale and design of OJAzzIC (Organisations Juggling Adaptation with Improvised Coordination), a meta-model for Organisations of agents that improvise in response to a dynamic situation[1,2].

The key ideas and features in the OJAzzIC meta-model are introduced in Section 5.2. The meta-model includes an organisation meta-model, an agent meta-model, a functional meta-model, an environment model and a coordination model specified using policies to guide agent interactions. The organisation meta-model is developed further in section 5.3. In Section 5.4, we describe the OJAzzIC agent meta-model as a modification to the traditional agent BDI deliberation cycle to include two levels of agent awareness of others within an organisation — i.e. agents not only consider other agents in their own individual deliberations, but agents also deliberate with others in

[1]The OJAzzIC model presented in Chapter 5 was first proposed at the 12th International COIN workshop held at AAMAS 2011 K. Keogh, E. A. Sonenberg, and W. Smith (May 2011). "Organizations with Improvised Coordination: OJAzzIC". in: *Proceedings of 12th International COIN Workshop, Internation joint conference of Autonomous Agents and Multi-agent Systems, AAMAS 2011*. Ed. by S Cranefield and P Noriega. Springer and published in 2012 K. Keogh and E. A. Sonenberg (2012). "Adaptive Coordination in Distributed and Dynamic Agent Organizations". In: *Coordination, Organizations, Institutions, and Norms in Agent System VII, LNAI 7254*. Ed. by S. Cranefield, M. Birna van Riemsdijk, J. Vázquez-Salceda, and P. Noriega. Vol. 7254. Heidelberg: Springer-Verlag, pp. 38–57.

[2]Further explanation of the use of social policies to guide the agents in an OJAzzIC organisation in Chapter 5, Section 5.6.2 was presented at the 16th International COIN workshop held at PRIMA in 2013 K. Keogh and E. A. Sonenberg (Dec. 2013). "Coordination Using Social Policies in Dynamic Agent Organizations". In: *Proceedings of 16th International COIN Workshop, PRIMA 2013*. Ed. by T Balke and M B Riemsdijk and published in 2014 K. Keogh and E. A. Sonenberg (2014a). "Coordination Using Social Policies in Dynamic Agent Organizations". In: *Coordination, Organizations, Institutions, and Norms in Agent System IX, LNAI 8386*. Ed. by T. Balke, F. Dignum, M. B. Riemsdijk, and A. K. Chopra. Vol. 8386. Springer, pp. 1–20.

an organisation; We describe the functional meta-model for tasks and goals and how they relate to organisation roles in section 5.5. In Section 5.6, the coordination model is described. Contracts are described in section 5.6.1. In Section 5.6.2 we provide some justification regarding the decision to use social commitments and policies. The external environment model is considered in section 5.7. In Section 5.8 we use a scenario from EM to illustrate how policies can ensure the creation of social and information contracts that achieve knowledge cultivation and coordination obligations using commitments. We revisit the requirements introduced earlier in the thesis in section 5.9. In section 5.10, an example system specification is provided for a MAS model based on OJAzzIC This relates to the rescue scenario simulation introduced in Sections 2.3.2 and 5.8. In section 5.11, OJAzzIC is compared in more detail with a number of existing models to demonstrate the benefits of adopting the OJAzzIC meta-model.

5.2 OJAzzIC: Agents in Organisations Juggling Adaptation with Improvised Coordination

In this section, we describe our meta-model, OJAzzIC. The meta-model addresses our requirements for improvised coordination as outlined in section 4.7. The requirements we consider result in a meta-model that can be used to model MAS that can capture the necessary static and dynamic knowledge to enable members to behave as a Jazz musician might — to improvise and adapt their script on the fly, but not in such a way that it would interfere with the script or plan adopted by others. The plans need to be clear, but flexible. Behaviour needs to be coordinated, but not prescribed. OJAzzIC builds on the adaptable organisational structure used in OMACS (S. A. DeLoach 2009) and combines with features from contract based systems (V. Dignum, J.-J. Meyer, and Weigand 2002), commitments (Carabelea and Boissier 2006; Telang, Meneguzzi, and M. P. Singh 2013) and intentional approaches to joint planning used in SharedPlans (B.J. Grosz and Sidner 1990; B. Grosz and Kraus 1999; B. Grosz and Hunsberger 2006). OJAzzIC defines a static organisation structure that can be defined at design time as well as a framework for how this structure can be used to create organisation instances at run time.

The novelty in our approach is to combine the adaptive nature of a organisation structure—enabling both flexible (re)allocation of agents to tasks and dynamic goal decomposition— with a social contract that defines explicit obligations and coordination policies. Dignum suggests that a social contract can be based on a predefined script that can be well defined or loosely governed by predefined landmarks (Weigand,

V. Dignum, J.-J. Meyer, and F. Dignum 2003). Defining landmark states as sub-goals that may be reached along a pathway toward the overall goal enables agents to dynamically work toward reaching the final goal. The social contract defines the responsibilities and obligations of individual agents within the organisation. In OJAzzIC, agents create social commitments based on predefined policies in a social contract adopted by an organisation. These policies provide guidance for necessary coordination and knowledge sharing between agents. Agents in OJAzzIC reason individually, giving consideration to organisation goals. Additionally, our proposal for the use of multiple *instances* of overlapping organisations at run time enables coordination within and across organisations.

A problem is resolved by agents taking action to achieve goals and reach resulting landmark goal states. A plan comprises a set of goals that when adopted and successfully fulfilled by agent(s) will reach a desired state. Agents have a library of actions that they can use in order to reach a goal state. At any point in time, an agent can create an intention to reach a landmark goal state by adopting a goal as a current active goal. Goals may be adopted based on knowledge that reaching that goal is a step toward reaching a higher level goal. An agent uses the information defined in the organisation to inform its behaviour. Individually, the agent deliberates to consider goals to adopt and select action plans in order to reach organisation goals and individual goals.

Organisation role responsibilities are reached by an agent acting in that role performing actions and conforming to role obligations. An agent can enact an organisation role, defined for a particular context, however the agent is not necessarily limited to only adopt role related goals. In order to successfully complete actions, agents require the capabilities to match these. Individual agents adopt a plan of actions in order to achieve particular adopted goals. The goals and sub-goals that represent a pathway to a problem solution are explicitly defined. Goals can be described in terms of sub-goals and tasks that comprise these. Agents might also adopt individual goals outside of roles adopted if they possess the matching required capabilities. If it is necessary for multiple agents to work together in a coordinated way in order to achieve an organisation goal, agents at run time create explicit commitments to adopt individual goals in order to achieve the organisation goal.

Plans are based on instantiating plans defined at design time by selecting valid goals from a decomposition of goals and tasks in a Goal Tree. Goals are considered to be valid based on the particular context. Enacting a plan many involve dynamically negotiating more detailed plans with another agent. Plans to achieve goals requiring coordination between multiple agents are established dynamically using Shared-

Plans (B.J. Grosz and Sidner 1990; B. Grosz and Hunsberger 2006). When two agents agree to create a SharedPlan to coordinate their individual behaviour this SharedPlan forms part of a contract between the agents. The agents each individually commit to this SharedPlan by creating explicit commitments at run time. When agents have created a SharedPlan, these agents have a realistic expectation of the behaviour of their collaborating partner in terms of working together to achieve the goal related to the SharedPlan.

Figure 5.1 is a conceptual diagram with rectangles representing components in the organisation. These components are external to the agent, however the agent will have knowledge or beliefs regarding

- the capabilities that are possessed by agents
- roles being enacted by agents
- organisation membership (agent list)
- organisation goals and sub-goals

Figure 5.1 shows how each organisational instance relates to agents at run time. When an agent creates commitments and SharedPlans at run time, the agent negotiates with and informs other collaborating agents so that each agent has an internal representation of shared mutual knowledge or beliefs regarding the agreed intentions. The organisation contract defines policies that specify agent obligations. These policies guide agents to behave in a coordinated way, including appropriate sharing of beliefs and intentions across the organisation.

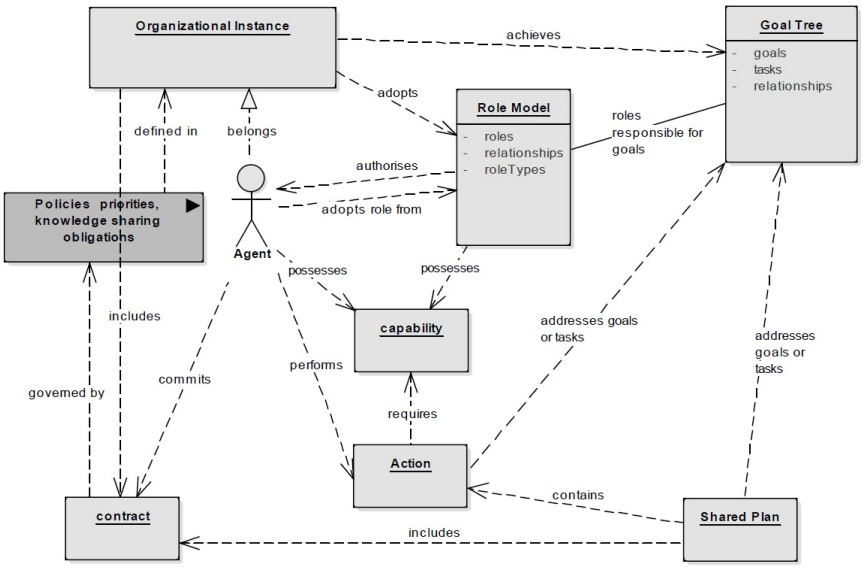


Figure 5.1: Conceptual components in an OJAzzIC organisation

An organisation structure is instantiated to create each actual organisation in-

stance at run time. This organisation instance may be quite a stable and persistent structure defined at design time and based on formal roles, but it may also be a short term organisation created at run time so that a group of agents can work together dynamically in a coordinated way. In the latter case, the coordination may be negotiated at run time rather than be based on predefined scripts. In the former case, the coordination may be based upon or adapted from default scripts.

A short term organisation is referred to as an *adhocracy* (D. Mendonca, Jefferson, and Harrald 2007). Figure 5.2 represents the life cycle of an organisation at a conceptual level. This diagram is presented using the notation of a UML activity diagram. Activities of agents and the organisation are represented in rounded rectangles. If the designer has created organisations at design time, these are available to enable agents at run time to interact and coordinate with each other. If, at run time, agents who are not in an existing organisation need to coordinate, these agents can dynamically form a run time organisation (adhocracy). When an organisation exists, agents within the organisation are obliged by organisation policies that can be used to guide their coordination of beliefs, intentions and actions. It is up to the designer to decide if it is necessary to formally finalise and close an organisation or to leave it in existence for the duration of a problem solving scenario. If an organisation remains in existence, there is a future potential for agents within this organisation to collaborate.

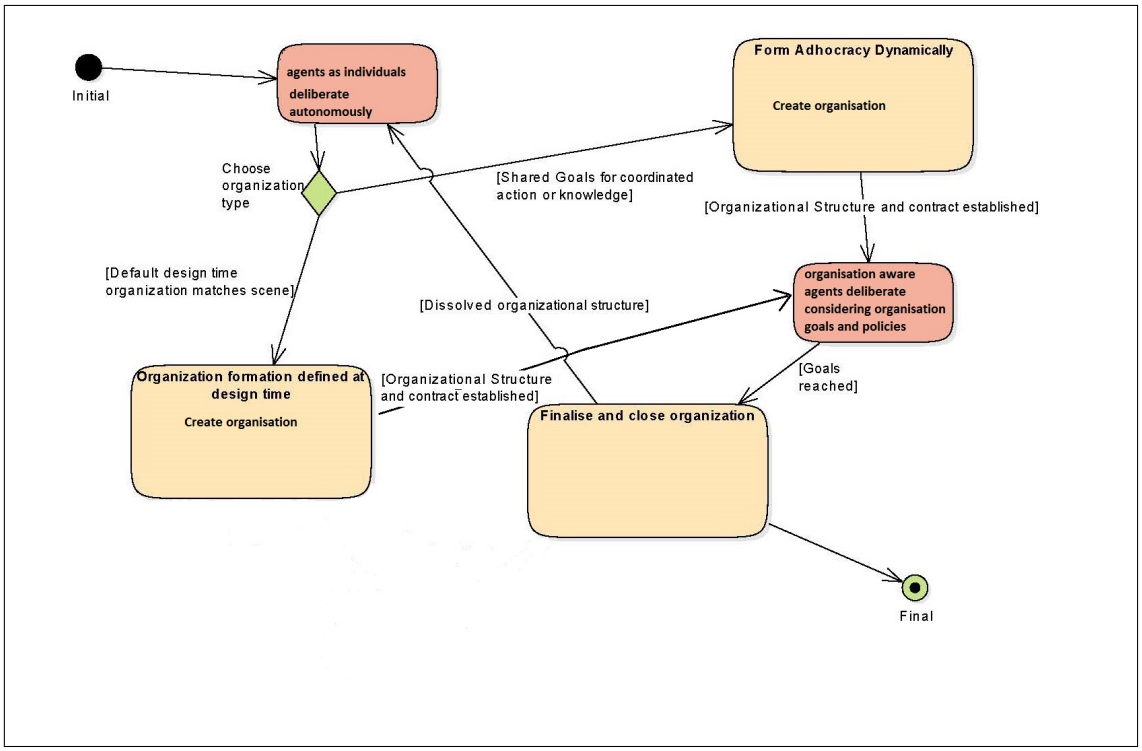


Figure 5.2: Life cycle of an organisation

5.3 Organisation meta-model

In this section, the OJAzzIC meta-model is described in more detail. The description has two parts. Firstly, the organisation meta-model is described as a entity representing the organisation and secondly, the way that organisation instances created based on the organisation meta-model are used as part of the execution model is described.

5.3.1 Organisation meta-model structure

Figure 5.3 shows the OJAzzIC organisation meta-model as a group of related elements in a MOF model (OMG 2016). The diagram uses symbols similar to that found in a UML structure diagram to indicate the associations and relationships between the elements. Where multiplicity is not indicated it is a singular relation. Each organisation in OJAzzIC is created based on this meta-model.

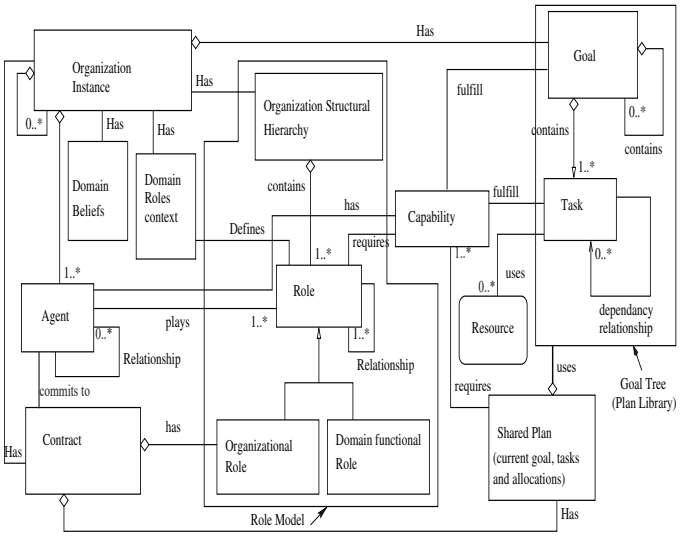


Figure 5.3: The OJAzzIC organisation meta-model

A definition of the OJAzzIC organisation meta-model follows. In OJAzzIC, an organisation O is a tuple representing a first class entity created at run time.

$O : <G*, R, Re, Contract, A, C, P, \sum>$

Some of the components of the organisation may be specified at design time to populate the organisation instance created at run time. The core components of the organisation are:

G: Goal Tree, including ordering tasks where necessary. This defines the organisation goals and how they could be decomposed into sub goals and ordered tasks; A task is a leaf goal, that cannot be further decomposed. A task may use a resource and can be achieved by an agent who possesses the capability associated with fulfilling that task.

R: the Role Model including a set of domain roles, relationships between roles and context based role definitions. This may also include coordination roles. Role relationships define authority between roles.

Re: a dynamic Resources list defining objects in the environment that can be used to help perform tasks;

Contract: The Contract contains a *social contract* and an *information contract*. The social contract comprises SharedPlans created by agents. The SharedPlan outlines the current selection of tasks to achieve a goal and the explicit agreement and commitments between agents thus far assigning responsibilities for tasks to individual agents. When agents at run time agree to work together and require a coordinated plan in order to successfully coordinate individual goals and actions a SharedPlan is created. The information contract is a set of agreed policy obligations and commitments to intentions to ensure consistency of beliefs within the agent organisation. The information contract is responsible for the creation of run time messages between agents regarding β the relevant beliefs set that includes beliefs about the environment, including resources.

A: set of Agents. Agents can enact roles defined in the organisation if they possess the capabilities required to do so. Agents may also adopt individual goals to fulfil tasks if they possess the required capabilities and if policies permit this;

C: set of Capabilities. Capabilities can be associated with fulfilling tasks or enacting roles;

P: set of policy constraints to apply to all members and to the adoption/allocation of tasks; *Authority* Policies explicitly define a process for adoption/acceptance of allocations by an agent and *Coordination* Policies explicitly help agents coordinate multi-agent plans dynamically. These social policies are used in OJAzzIC to specify obligations regarding knowledge cultivation, coordination and creation of dynamic organisations.

$\sum$: environment domain meta-model used to specify environment objects and relationships

The domain model and resources available in the environment, external to agents are domain specific. It is necessary for an interface to exist between agent and the environment as part of the environment meta-model so that agents can receive percepts from the environment. Agents need to have a model of the environment in order to understand and assess what goals are valid to pursue. Aside from mentioning that there needs to be a model of the environment internally for agents and the interface between agents and their environment, we do not specify anything further about the environment meta-model.

Optional components of the organisation, useful to assess or create assignments of agents to goals include:

oaf: organisation assignment function measures the utility of a particular SharedPlan assignment of Agents to Roles to Tasks; This could be useful to help an agent evaluate from a number of potential plans at run time. However, it may not be optimal to require agents to consider all possible plans and evaluate these in order to select the 'best' plan. Such design decisions are not specified as part of the OJAzzIC meta-model

achieves: function defining role adoption or assignment — how effective the behaviour of roles can be to achieve task T or goal G in a SharedPlan. This could be used if agents have

the reasoning abilities to revise a default plan or if a default plan cannot be found for a context. The Goal-Tree can be used to derive a plan. In the absence of defined achieves functions, it can be presumed that all mappings of capabilities are equal. This means that if an instance is found to map a capability to a requires function for role or task and an agent is available possessing that capability, then the agent can adopt the goal associated with that role or the goal associated with achievement of the specified task.

requires: defines the capabilities required to play a role R or task T; and

possesses: function defining the quality of an agent's capability for a particular Task. To decide how well an agent can play a role, the requires and possesses functions are combined into a function: capable. To decide how well an agent can play a role to achieve a goal, the capable function and achieves function are combined as a function: potential.

Supporting evaluation functions *oaf, achieves, requires, possesses* are optional and are based on similar functions defined in OMACS (S. A. DeLoach 2009). These are not central to the requirements addressed by OJAzzIC however, could be incorporated in future work to allow for dynamic planning. The definition of function achieves is extended to include tasks and SharedPlans. These support functions may be used if an agent needs to evaluate from a number of potential possible plans at run time, or if a central coordinator/system is allocating goals to agents. These are not explored further in this thesis.

5.3.2 Organisation run time execution model

In this section, the run time execution model for agents in OJAzziC organisations is described. Agents start by belonging to a large organisation responsible for the main high level goal (e.g. the entire system). When two or more agents need to coordinate, if they are not already in an organisation, then a new organisation is formed and within that organisation an explicit contract dictates policies, obligations, agreed goals and agreement to coordinate with others in that organisation. Policies can be specified at design time. The run time contract is created by agents making commitments to each other regarding sharing of beliefs, intentions and outcomes. Default policies can be adopted, or specific policies designed for particular types of organisation can be separately specified at design time. Coordination within the organisation relies on appropriate communication to share relevant information and share plans with others in the organisation. The new organisations that form overlap with existing organisations, so that agents can belong to multiple organisations in the distributed system. In one system organisation, multiple smaller organisations are created. These organisations each need to be explicit so that appropriate coordination can be established within each. Each organisation instance created is based on the OJAzzIC organisation meta-model

structure.

We adopt the use of contracts to manage obligations to ensure agents will share domain knowledge as well as structural and coordination knowledge within each organisation. Collective obligations can be implemented as policies to govern joint activity and teamwork (Van Diggelen, J. Bradshaw, Johnson, Uszok, and P. Feltovich 2009). Social Policies are used in OJAzzIC to specify obligations regarding knowledge cultivation, coordination and creation of dynamic organisations. The designer can anticipate a number of organisation(s) as part of the design process and allocate agents as members of long term organisations. It is also possible that agents may need to form an organisation at run time.

Figure 5.4 shows a high level algorithm for the creation of an instance of an OJAzzIC adhocracy organisation. This adhocracy would be created with a particular member group of agents, triggered by the need to establish an adhocracy in order to coordinate to achieve a particular goal. In this algorithm, the first step in the creation of a contract involves the creation of mutual agreement on potential intentions between members regarding a high level plan to achieve the goal. These intentions would establish that members in the organisation agree that they have potential agreement toward adopt a specific SharedPlan to reach a particular goal. Agents in the adhocracy also agree to adopt policies. Policies define obligations and ensure that appropriate beliefs and organisation knowledge is shared within the (ad hoc) organisation.

When two agents need to coordinate and create a SharedPlan, they need to be aware of the importance of associated communication and commitments to each other. At design time it is necessary to identify beliefs relating to the plan and define policies to obligate agents to create commitments at run time to share these beliefs. If these two agents are not already part of an existing organisation, then they can create an organisation to assist coordination between themselves. This is an instance of an adhocracy, a short term organisation. Creating an organisation would then obligate each agent (as defined in the social contract) to appropriately share information such as a SharedPlan to facilitate coordination between these agents. Multiple organisations could be created dynamically as agents elect to work together to achieve goals. When one agent is part of multiple organisations, this overlap allows for relevant information (e.g. location of injured, status of rescue) to be propagated across the network across organisational boundaries.

The sharing of relevant knowledge within and across organisations is important as knowledge may be distributed. Establishing a shared mental model (SMM) of the situation between agents within the organisation is important. Obligations to ensure

```
CREATE_ADHOCRACY_ORG(member_list)
  if (adhocracy_trigger) then
        NEWORG = create_org_structure()
        NEWORG.add_members(member_list)
        NEWORG.generate_goal_tree(goal)
        if (NEWORG.create_contract (goal)) then
              return SUCCESS
        else
              return FAIL_TO_CREATE_ORG

CREATE_CONTRACT(goal)
  thisORG.agree_high_level_plan
   establish mutual agreement of a potential intention of a SharedPlan at a
  high level (intention-that)
  thisORG.agree_obligations()
   create intentions to ensure mutual knowledge of beliefs
  thisORG.agree_beliefs()
  thisORG.agree_roles()
   agree on coordination or other roles needed
  thisORG.generate_SharedPlan()
   create more elaborated SharedPlan with mutual intentions agreeing opera-
  tionally how to achieve shared tasks
```

Figure 5.4: Pseudocode for creating an adhocracy contract for coordination, *comments italicised*

appropriate knowledge-sharing about SharedPlans between individuals (B.J. Grosz and Sidner 1990) and groups (B. Grosz and Hunsberger 2006) has been well defined and we also adopt this intention based approach to planning. In the next section the individual agent meta-model is described.

5.4 Agent meta-model

In this section, focus is given to the individual agents who are members of an OJAzzIC organisation and in particular, the execution model for agents. We adopt agents as goal-based individuals who have autonomy to adopt individual goals and possess capabilities to act in order to reach outcomes associated with goals. However, agents do not act in isolation, but with awareness of the community or organisation to which they belong. It is necessary for agents to perform individual deliberations to make decisions regarding their individual behaviour in the knowledge that they are in relationship with others and with the organisation to which they belong. Membership of the organisation brings with it associated constraints that are either imposed upon or adopted by an agent. Our focus is on cooperative BDI agents (A. S. Rao and Georgeff 1991) who wish to work toward reaching a common organisational goal. A BDI agent has a representation

of individual beliefs, desires, intentions and plans.

5.4.1 Agent run time execution model

OJAzzIC organisation agents have the following behavioural properties and relationships:

- An agent is autonomous, within the constraints imposed by the organisation(s) to which it belongs
- An agent may belong to one or more organisations
- An organisation instance contains a number of agents
- An agent may enact zero or more roles within an organisation
- An agent may adopt an individual goal based on individual priorities
- An agent will possess capabilities that enable it to perform tasks in order to act on the environment or communicate with other agents in order to achieve a desired outcome
- An agent will have either an ability to dynamically select a set of actions in order to achieve a goal or to select from predefined plans specified at design time
- An agent will have an ability to negotiate with other agents dynamically in order to agree on a SharedPlan where multiple agents need to coordinate their behaviour. Each agent will create an individual plan that is consistent with other agents' plans so that each individual's behaviour is coordinated
- When agents need to work together, they create an adhocracy and form an organisation contract. A contract contains a (set of) commitments between agents and also a set of policies that govern each agent's behaviour
- An agent enacting a particular role, defined within an organisation may, as part of this role, inherit multiple capabilities and responsibilities
- The organisation can impose obligations upon a member agent

As Corkill (Corkill, Durfee, Lesser, Zafar, and C. Zhang 2011) and others argue, agents within an organisation, need also to consider organisational objectives in addition to their own goals. Agents must ensure that individual mental attitudes are managed so that they are not inconsistent with organisational attitudes. Corkill describes *organisationally adept agents* that use adjustable preferences to value self, others and the organisation to different degrees whilst evaluating utility functions in selecting actions to perform. Management of mental attitudes in the individual agent is not developed further at this stage. For simplicity, it is assumed that each agent has individual attitudes that will be consistent with others in the organisation because of the interactions between agents. An agent is assumed to fully trust information received

from another agent.

Our approach is based on the BDI architecture (A. S. Rao and Georgeff 1991) that enables goal-directed behaviour in the presence of explicit deliberation about changes in the environment. Traditional BDI agents deliberate based on a self-interested cycle similar to that shown in Figure 5.5 (A. S. Rao and Georgeff 1995). We explicitly include reasoning considering organisation goals as part of the agent reasoning cycle. Others have modelled extensions to BDI agents toward agent groups or teams that include agents modelled with mutual or shared attitudes e.g. (G. Tidhar 1999). The organisation-aware agent deliberation cycle is shown in Figure 5.6, so that agents within an organisation might not only deliberate individually, but as an organisation. Agents therefore would be capable of individual goal or utility based (self-interested) reasoning, other-centred reasoning — with *awareness of* others, as well as organisational reasoning *with* others (G. Tidhar 1999; Kinny, E A Sonenberg, Ljungberg, Tidhar, A S Rao, and Werner 1994).

```
repeat
        perceived-events := event-selector(event-queue);
        update-attitudes();
        plan-options := option-generator(perceived-events,current-goals);
        selected-plan-options := deliberate(plan-options);
        update-intentions(selected-plan-options);
        execute();
end repeat
```

Figure 5.5: Individual agent BDI deliberation cycle

OJAzzIC is a meta-model that specifies features in an agent organisation to facilitate flexible, adaptive or improvised behaviour by agents. The meta-model is created for BDI agents who are organisation aware. This means that the agents have a reasoning cycle that has been extended to include organisation reasoning. Agents have the ability to deliberate and select actions to achieve individual goals whilst considering the organisation goals. Agents can also coordinate their actions by using individual commitments to create agreement regarding collaborative activity. This collaborative activity does not need to be specified at design time with detailed scripts or plans outlining exactly how it will be achieved, but agents at run time can work out the lower level details (such as which agent will do which task). In our approach, the obligations to update individual mental attitudes are made explicit in a social contract that defines congruence between organisational attitudes and individual attitudes to create mutual knowledge that is consistent across the organisation.

In the extended deliberation cycle, an agent perceives environmental input, up-

```
repeat
    perceived-events := event-selector(event-queue);
    process-individual-and-organisational-attitudes(current-social-contracts);
    plan-options := option-generator(perceived-events, current-goals);
    selected-plan-options :=
                deliberate(plan-options, other-agents-and-organisations);
    update-intentions(selected-plan-options);
    execute();
end repeat
```

Figure 5.6: Organisational agent BDI deliberation cycle

dates individual mental attitudes but, before selecting an intention the organisational agent will deliberate *with others in the organisation*. This organisational deliberation is defined by obligations and policies in the social contract within the organisation. This deliberation includes:

- Communication to share beliefs and intentions
- Communication regarding implementation of a SharedPlan
- Updating individual mental attitudes following communication from others in the organisation
- Revision of or mutual adjustment of individual plans considering others

The basic logic behind an organisation-aware agent *processing individual and organisational attitudes* is: if there is a objective for the organisation that is valid in the current context (scene) for a role that I am enacting, then I should adopt the consideration of this objective as shown in Figure 5.7. Following the first stage of organisational deliberation and any needed adjustment to individual attitudes, the agent continues the deliberation process *considering others* — when the agent may revise individual intentions and plans to ensure that they are not intending anything that will hamper others and possibly to add new intentions to help others.

It is up to a designer to ensure that agents have a mechanism to reason about how to prioritise and adopt goals, especially when there is the possibility that an agent will have some conflict regarding goal selection. When agents perform actions, they select actions to take based on resolving to see to it that their adopted goal is achieved. Authority policies are used to define priorities for agents regarding goal consideration and adoption. Each agent possesses a set of capabilities that enable it to achieve certain tasks. Tasks can be aligned to achieving goals that may be described in terms of sub-goals or tasks. When agents work together, they do not need to have detailed scripted plans that specify exactly how goals are to be achieved, but instead, agents

If I believe (org.objective) then	*I believe I am responsible for org.objective*
adopt (consider(agent_me, org.objective, scene))	*adopt a goal to consider that objective myself*
else	
update org.objective ← new objective	*find valid new organisational objectives that I am responsible for (based on my role). Update my beliefs to add objectives*

Figure 5.7: High level logic for organisation aware agent processing organisational attitudes (pseudocode)

adopt goals and using a commitment-based approach agents communicate with each other and commit to the achievement of individual goals that will together contribute to the achievement of organisation goals.

The term "Multi-Agent Plan Coordination" describes a class of coordination problems where agents' individual plans need to be coordinated to avoid interference/conflict and to enable cooperation/assistance where possible (Cox and Durfee 2009). This class of problems is suited to the deliberation cycle we describe. When it is required (e.g. due to interdependencies between goals), agents will coordinate their individual plans to fit in with others. A SharedPlan is created to establish this commitment. The SharedPlan is not a separate external artifact, but is a set of commitments created between agents to make explicit their intentions. If a group of agents decide at run time to work together and need to coordinate their behaviour, the OJAzzIC approach would require that if these agents are not already in an existing organisation, then these agents create an adhocracy (a short term run time organisation instance). The presence of the organisation instance has a consequence of enabling agents to have a level of organisation awareness so that appropriate knowledge sharing (knowledge cultivation) occurs. Agents working together need to follow organisation guidelines regarding what information should be shared (e.g. situational beliefs, intentions). These guidelines are specified using social policies.

OJAzzIC is based on the premise that organisations of agents exist at run time in order to provide a mechanism for coordination of individual agents within a multi-agent system. Agents may belong to multiple organisations simultaneously. Agents within the MAS are autonomous and individually adopt goals based on their individual priorities for goal adoption. Agents are organisation-aware, in the sense that the organisations created are first class entities and inspectable by agent members. This means that an agent is aware of organisation(s) that it belongs to and are aware of other agents who are members of the same organisation. Agents behave as individuals, but due to their

membership of organisations and awareness of organisation goals, agents can prioritise the individual goals that they adopt in order to achieve organisational goals. In the next section, the representation of organisation goals is described in more detail.

5.5 Functional meta-model: goal trees and role model

In order to achieve our aim of flexibility, we choose to keep separate the goals to be achieved by an organisation and the available roles that may be used to define (or allocate) responsibility to achieve these goals. This means that there does not need to be a 1-1 direct mapping between goals and roles that will be responsible to achieve the goal. Using careful goal design according to a goal decomposition tree, goals can be ordered so that goal dependency is represented in the goal model, as well as alternative options, so that if one set of goals cannot be satisfied, an alternative set may be chosen. The agents do not plan a solution in terms of deliberating to choose how to reach a goal, rather, agents at a point in time, select a valid set of sub-goals that can be satisfied to reach the adopted goal. Each agent works autonomously to adopt goals for which they are responsible. This is non-deterministic as agents have individual autonomy to adopt goals. If an agents adopts a goal that requires coordination with another agent, then before the agent adopts any sub-goals toward reaching this goal, the agent will first create a SharedPlan with a collaborating agent to agree that they will together achieve this goal.

Figure 5.8 shows the organisation structural components used to represent a problem in the OJAzzIC organisation meta-model using UML notation. This shows how abstractions for capabilities and tasks relate agents to roles and roles to goals respectively. This allows for flexibility in agent behaviour at run time because available agents can adopt goals to lead toward reaching a solution.

In this section, we use the term goal to refer to a particular objective or state of the world that an agent may take action to reach. During a cycle of deliberation, when a BDI agent adopts a goal, then the agent is signaling its intention to take action in order to achieve that goal. If the goal is not a leaf goal, there may be multiple options in terms of paths of sub-goals to adopt leading to that goal.

Agents can adopt a small goal and enact a plan for this sub-goal that contributes toward a higher level goal. They then take action toward achieving this goal and subsequently, if necessary, change or adapt plan details. This adaptation can involve dropping a goal in order to adopt a different goal to fit in with other agents and/or to adjust to a different solution. This is similar to agents adopting and committing to a

partial plan in SharedPlans and also is related to delayed agent commitment for BDI agents (Dam, T. Zhang, and Ghose 2013).

In Chapter 4, some background and discussion of the OMACS meta-model is provided. The OMACS meta-model has informed the organisation structure used in the OJAzzIC meta-model significantly, however OJAzzIC incorporates additional abstractions in order to achieve flexibility.

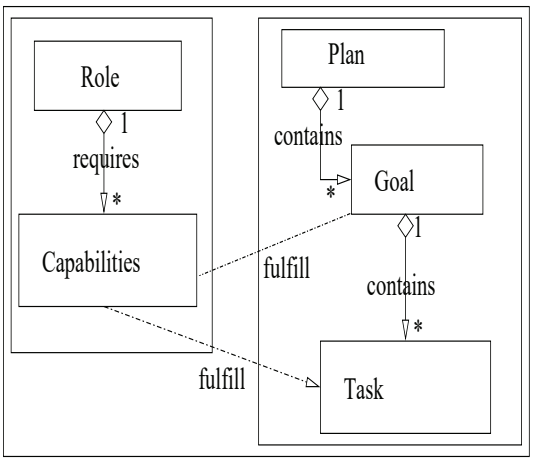


Figure 5.8: Problem representation relating roles and tasks in an OJAzzIC organisation

The OMACS meta-model assumes that one role will achieve one goal. In OJAzzIC, an extra level of separation is introduced between roles and goals as indicated in Figure 5.8 and Figure 5.9. In OJAzzIC, Agents can adopt goals based on responsibilities for Roles *or* based on Capabilities to fulfil Tasks. In OJAzzIC, the Goal Tree is extended to include Tasks as a possible decomposition of Goals.

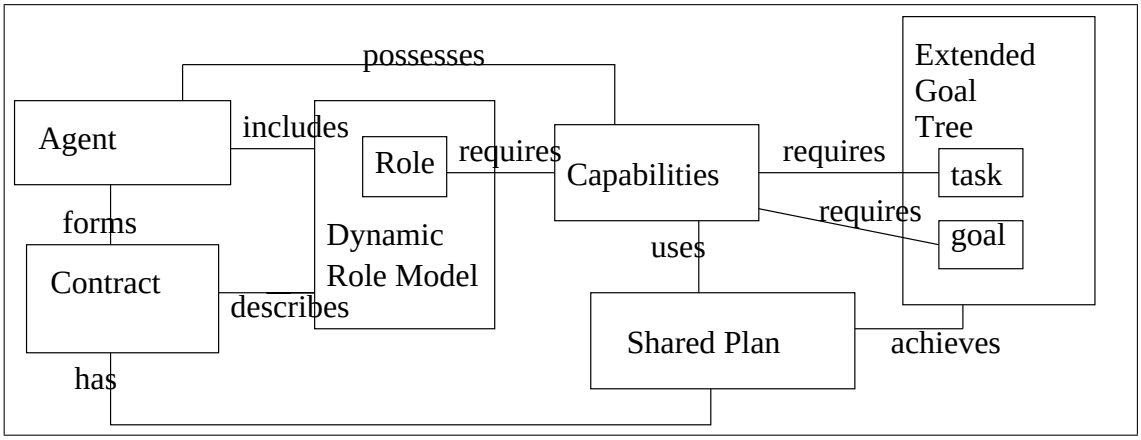
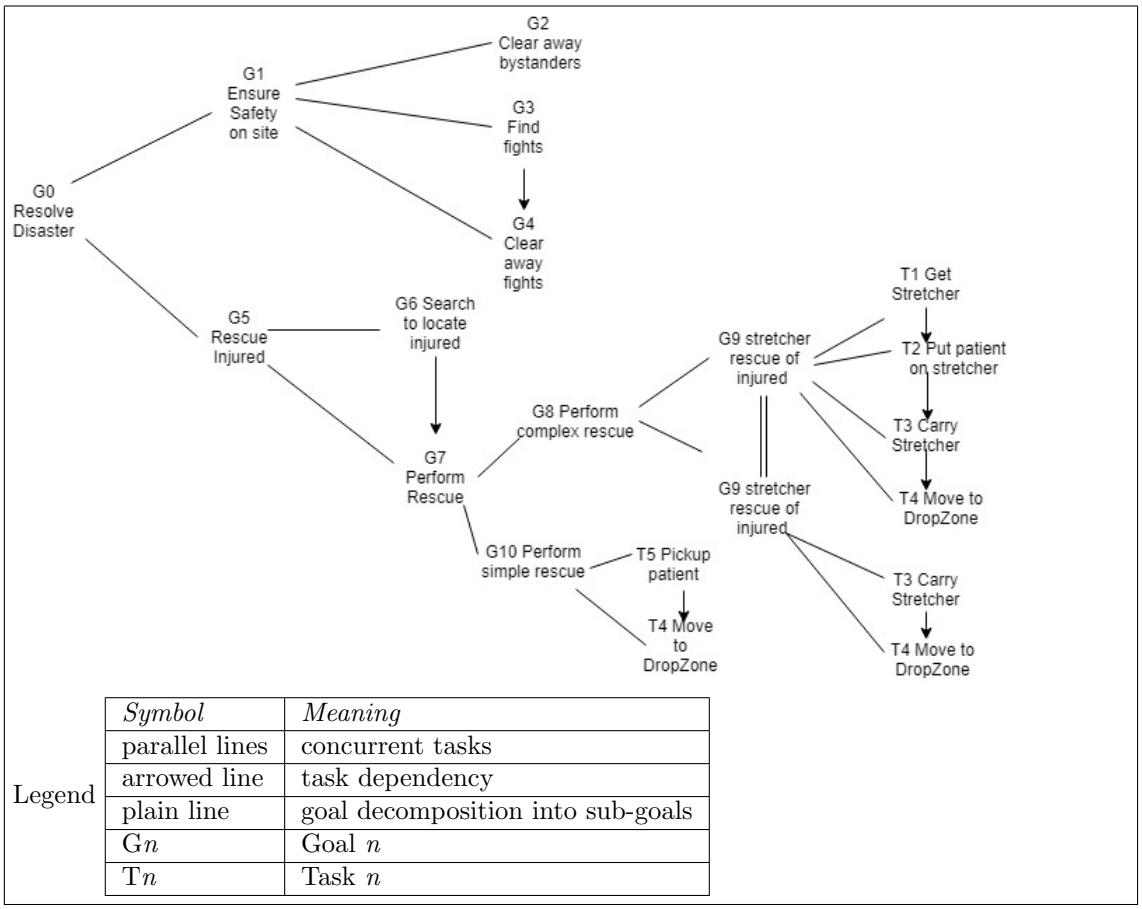


Figure 5.9: Agent-Goal Abstraction in OJAzzIC

The Role Model in OMACS is a fixed relationship of predefined roles (represented as 'Role' in Figure 4.1) whilst in OJAzzIC (Figure 5.9), the Role Model is dynamic, it is created based on context and represents the roles instantiated by agents in the or-

ganisation. The main distinguishing feature in the agent-goal abstraction for OJAzzIC compared to OMACS is the use of tasks and the use of individual capabilities in agents (perhaps outside a role) to decide upon the ability of an agent to adopt a goal.



Legend

Symbol	*Meaning*
parallel lines	concurrent tasks
arrowed line	task dependency
plain line	goal decomposition into sub-goals
Gn	Goal n
Tn	Task n

Figure 5.10: Example OJAzzIC Goal Tree

In Figure 5.10 an example OJAzzIC Goal Tree for the emergency management scenario introduced in Section 1.2.1 is shown. The goal tree encapsulates knowledge about goal decompositions that could be used at run time to link to capabilities or roles of particular agents for goal adoption. Each goal may be aligned to a responsible role defined within the organisation in the role model. A task is a leaf goal that can be mapped to an individual capability possessed by an agent. It is not essential to break goals into separate tasks, however in doing so, more flexibility is achieved at run time as tasks can be adopted by individual agents if an agent is not available to enact a role associated with a goal. The goal tree can be thought of as a plan recipe library. The goals are connected using lines to indicate where a goal is decomposed into multiple sub-goals. If goals are concurrent, they are connected by parallel lines. Where there is some dependency between goals such that one goal must be completed before another, an arrowed line indicates this. The arrow points to the dependent goal. This example shows two different task breakdowns for the goal G9 stretcher rescue. Two

agents will coordinate individual behaviour using different sets of individual tasks in order to achieve the stretcher rescue. This will be accomplished using a SharedPlan. In OMACS, problems are defined in terms of a flexible decomposition tree of sub-goals involving alternative goal plans that could be adopted (S. DeLoach and M. Miller 2010). Deloach names this run time representation as Goal Model for Dynamic Systems (GMoDS) (S. DeLoach and M. Miller 2010). The OJAzzIC goal tree is inspired by this. OJAzzIC differs because the coordination of concurrent goals is not embedded in the definition of how agents will achieve the goals, but coordination is achieved explicitly using SharedPlans.

In OJAzzIC, the Role Model may be defined based on context and represents the roles enacted by agents in the organisation. This allows for flexibility in terms of certain roles being introduced according to context. For example, when a situation becomes more complex, it may be that different roles are required and/or different responsibilities are associated with a role (based on domain requirements). This design is motivated by reality as discussed in Chapter 4. In real situations, a role description may change based on context and roles might need to be shared. Potential organisation structures, role relationships and role definitions are explicitly defined at design time.

The role model defines organisation roles and explicitly aligns these roles with goals in the goal tree. For each role defined, a set of capabilities associated with this role would be listed. Any agent adopting this role would necessarily possess or be granted these capabilities. Roles defined for the example goal tree shown in Figure 5.10 could include Medic role and Officer role. The Medic role could be responsible for the goal: Rescue Injured and the Officer role could be responsible for: Ensure safety on site.

Synchronized Tasks If a goal cannot be achieved by one role, then multiple roles or agents can combine to achieve a goal by working together in a coordinated way. We describe goals as composing synchronised sub-goals. Sub-goals that can be performed by agents with the appropriate capabilities and have no further sub-goals are tasks. This abstraction is introduced to enable flexible and dynamic planning by agents to establish a coordinated SharedPlan to work together to achieve a goal. We choose to split goals into separate synchronised sub-goals rather than split roles, as intuitively, this abstraction fits with our observations from real life examples in the EM domain. For example, consider a goal to move an injured patient on a stretcher that might be achieved by two personnel — each in the role of 'side stretcher carriers' on the left and right, additionally, one of these stretcher carriers also takes responsibility for getting the stretcher and ensuring the patient is safely on the stretcher. The goal to move the patient can be described as 2 carrying stretcher goals that occur simultaneously and

each goal is accomplished by one individual, acting in the role of 'side stretcher carrier' who adopts one of the described goals.

As shown in Figure 5.10, a goal may comprise multiple sub goals. Goals may also be decomposed into tasks. Tasks and goals can be ordered. This abstraction is to enable the splitting of goals to share between multiple agents. When a goal is split, performing sub goals or tasks can require coordination between the agents. In Figure 5.10, goal G0 is decomposed into sub goals G1 and G5. G6 must be performed before G7. G10 is described in terms of tasks T4 and T5. In some cases, one goal can be achieved by an agent enacting one role directly, for example G10 may be achieved by an agent in a role of 'Medic Agent'. Similarly, an agent in the role of medic could adopt the goal G9 to perform a stretcher rescue. This goal G9 can be decomposed into multiple tasks. In the absence of an agent allocated to medic role adopting G9 for whatever reason, an alternative is to split goal G9 into separate tasks T1, T2, T3 and T4. Agents need to communicate to share knowledge about the completion of these tasks. If necessary, these tasks then can be adopted by or transferred to individual agents based on individual capabilities. If it is necessary to coordinate tasks when goals are split and shared then the agents need to coordinate their behaviour using a SharedPlan. In Figure 5.10, a SharedPlan would be necessary to coordinate agents working on the simultaneous goal G9. Two agents need to coordinate their behaviour so that one agent will get the stretcher and put the patient on the stretcher while the other agent must be ready to support with carrying of the stretcher. Using a SharedPlan allows that agents can negotiate roles and responsibilities dynamically at run time to ensure coordination between the agents.

In order to allocate or have agents adopt goals from this goal tree, agents need to know which roles are associated with capabilities for each goal. Agents also need to be aware of their own capabilities and any roles currently being enacted. Importantly, agents need to have an ability to coordinate their actions when working together and coordinate sharing of relevant information within the organisation. Coordination of information - beliefs and individual plans between agents in the organisation is important and will be discussed in the next section.

5.6 Coordination model

In this section coordination within OJAzzIC is described. This coordination includes explicit run time interactions between agents to ensure appropriate sharing of beliefs and construction of SharedPlans. The run time execution model for the OJAzzIC

organisation uses knowledge cultivation policies that explicitly define processes for establishing and cultivating mutual knowledge and sharing of intentions between agents within each organisation. We outline an approach for explicitly defining coordination mechanisms within agent organisations using *social commitments* and *social policies* to govern obligations between agents and to facilitate appropriate processes for knowledge sharing, coordination and improvisation.

Agent social commitments are explicitly managed using interactions so that for example, agents at run time explicitly commit to adopting a SharedPlan in order to behave in a coordinated way. A SharedPlan is a set of social commitments between two or more agents to ensure that a particular goal is achieved by the agents. The social commitments regarding the SharedPlan ensure that agents commit to the creation of mutual beliefs within the group so that the group forms explicit agreements regarding:

- commitment to potential intention-that (identified mutual desire to achieve a goal, but not yet adopted)
- commitment to potential intention-to adopting a plan (proposal to adopt a SharedPlan)
- commitment to adopting a plan (intention-that the proposal will become true)
- commitment to fulfilling the plan (intention-to achieving an intention-that previously adopted, commitment to adopt relevant goals to enact)
- commitment to establishing mutual knowledge relating to this plan (e.g. not changing the plan unless agreed)
- commitment to satisfying mutual beliefs relating to completion of this plan (communication of intentions)

Capabilities to enact these commitments can be embedded in the agent's individual capability set so that agents have the ability to adopt goal(s) to achieve these commitments and satisfy the achievement and coordination of tasks that require coordination. The satisfaction of mutual beliefs ensures that agents create intentions to communicate about their intentions and capabilities to do actions. As discussed in Section 4.4, in addition to maintaining mutual beliefs about SharedPlans, it is also necessary to have commitments between agents to cultivate knowledge (beliefs) about the current situation state. According to the SharedPlans theory, cultivate is the process of complex means-ends reasoning that will bring about actions to achieve intentions-that (B. Grosz and Kraus 1999).

Social policies externalise obligations adopted at run time by agents in an organisation. When creating commitments, organisational agents are guided by these policies. Social policies can define when commitments should be created regarding the establish-

ment of mutual knowledge. Social policies are similar to norms that define permissions and obligations. The obligations specified in social policies can be used to make explicit the mutual beliefs that agents should be cultivating and maintaining as mutual.

Social policies can define policies for authority, permissions, obligations (for example regarding knowledge cultivation, coordination of goal selection, awareness and creation of SharedPlans and contracts) and to define triggers for when new adhocracies should be created. When an adhocracy is created, the organisational members create social and information contracts to define roles, responsibilities and policies that define behaviour within the organisation.

In OJAzzIC, in order to address our requirements for organisational awareness and knowledge cultivation (discussed in Section 4.4), it is proposed that at a minimum, each OJAzzIC organisation adopts social policies regarding awareness and knowledge cultivation and each agent commits to these policies in a social contract. Additionally, agents must adhere to policies regarding sharing of knowledge in information contracts. Experimentation will be used to validate if it is helpful to adopt these policies.

5.6.1 Contracts

Contracts are a dynamic set of agreements (commitments) between agents and an associated set of policies adopted within the organisation. Intentional commitment and agreement between agents has been used to form mutual intentions toward a common plan or establishing common mental attitudes (Panzarasa and N. R. Jennings 2001). Using SharedPlans, agents can deliberate together to elaborate on plans dynamically.

Agents in an organisation must comply with the social contract policies that define interaction within the organisation. At the time a new organisation is instantiated in OJAzzIC, agents explicitly adopt the organisation contract that specifies rules to govern their behaviour. Each organisation exists for the duration of time in which there is a need for that group of agents to be coordinated. The organisation contract comprises a social contract that defines the social structure of the organisation and an information contract that defines how information is shared within the organisation.

The *social contract* defines role descriptions and agreed role allocations as understood by the agents. These may be created at design time and inserted into an agent's fixed knowledge base or dynamically inserted into each agents' belief base. Role descriptions may be abstractly defined at organisational design time and adopted dynamically. Roles are defined with associated capabilities, authority levels and obligations. Role relationships are also described. These are made explicit in the social contract to enable

revision if necessary. Having an explicit social contract provides the ability to predict others' behaviour and flexibly adapt individual goals in anticipation of others' needs and behaviour. The social contract may also define any agreed model for command, control and coordination. Authority relations and Coordination Roles such as Leader, Resource Manager, Knowledge Manager and Contract Manager could be identified and allocated if needed.

Based on the need to cultivate knowledge sharing, *information contracts* include policies to obligate members in an organisation to all adopt joint intentions to cultivate mutual knowledge within the organisation. This guides agents to interact and share knowledge updates appropriately. Obligations to share information are limited to the agents within each individual organisation that forms. As an agent can belong to multiple organisations, the overlap enables relevant information to be dispersed across a wider network as necessary.

5.6.2 Social policies and commitments

In this section, we demonstrate how policies are used to specify the behaviours of agents working in and across organisations in a complex, dynamic, distributed setting. Guided by social policies, agents can create social commitments to achieve coordination of knowledge and behaviour. We demonstrate such policies, drawing on the requirements for knowledge cultivation in an EM scenario.

Inspired by others, we use policies as guidelines within an organisational context to govern the creation of social commitments to ensure that we achieve appropriate coordination of knowledge and behaviour (Carabelea and Boissier 2006; Telang, Meneguzzi, and M. P. Singh 2013; Van Diggelen, J. Bradshaw, Johnson, Uszok, and P. Feltovich 2009). There are various extensions and adaptations of the basic notion of social commitment (Torroni, Yolum, M. P. Singh, Alberti, Chesani, Gavanelli, Lamma, and Mello 2009) including probabilistic commitments, time constraints and sanctions as part of the definition, e.g. (Carabelea and Boissier 2006; Chesani, Mello, Montali, and Torroni 2013; Dalpiaz, Chopra, Mylopoulos, and Giorgini 2011; Martinez, Kwiatkowski, and Pasquier 2011; Q. Zhang, Durfee, S. Singh, Chen, and Witwicki 2016). These do not contribute to the focus of our use of commitments currently, so we do not use these. Our focus is to describe obligations for coordination that would be adopted during the active life-cycle of an organisation and particular to each organisation.

The selection of social policies implemented for an organisation is a design decision. A minimal set of social policies would address:

- authority relationships and rights (e.g. if a role has authority over another role, what are the associated delegation rights)
- knowledge cultivation guidelines (e.g. policies for belief sharing, intention sharing, goal completion sharing)
- awareness of organisation goals and a prioritisation policy for adopting goals (For example, policies could suggest a priority ordering the adoption of goals associated with enacting organisation roles first, then adopt goals to complete an active SharedPlan with another organisation member, then adopt goals to fulfil a request by another agent, adopt goals matching roles that I am assigned in the organisation next, and lastly adopt goals matching my capability set)
- improvisation guidelines (e.g. policies to allow agents to adopt tasks outside their adopted role)
- triggers for creation of an adhocracy

In the current work, we assume that agents will follow the policies adopted in an organisation. Norms, obligations and sanctions between agents in the context of a society, institution or organisation are important and others give attention to this. We are not looking to contribute in this area.

Carabelea and Boissier (Carabelea and Boissier 2006) proposed how social commitments could be used to represent contracts between agents defining expected behaviour of agents in an organisation. We use an adaptation of their notation for social policies and commitments to explicitly define social obligations for agents in an OJAzzIC organisation. This commitment notation is consistent with Singh's (Telang, Meneguzzi, and M. P. Singh 2013), however where Singh has a social policy as a higher order, meta social commitment, Carabelea introduces a separate notational syntax for social policies and includes the status and pre-conditions explicitly. Similar to Carabelea, we use separate names SC to indicate a social commitment and SPolicy to indicate a social policy. We do not contribute to notational syntax for commitments or policies. We adopt notation for social commitments in organisational context based on Carabelea's notation. Carabelea use the context of IA, Institutional Agent to witness the commitment, however in OJAzzIC, the organisation is used as context.

Social Commitment

$$SC(debtor, creditor, Org, object, status[, condition]) \tag{5.1}$$

expresses that a social commitment is created between debtor agent to creditor agent in the context of the organisation Org, regarding object, with the given current status and held to be valid when the given condition holds (the latter term is optional).

Valid commitment status' are (Telang, Meneguzzi, and M. P. Singh 2013): null, active, pending, conditional, detached, violated, satisfied, terminated, expired. In all our examples, the status of the SC is active, so we omit this for brevity in our equations. The individual agent commitments in OJAzzIC are relationships between multiple agents, captured in the interactions between those agents. There is not a separate first class artifact or entity in OJAzzIC representing these commitments. If it is desired that agents share their intentions based on when they create commitments, then policies would be created to obligate agents in this way.

We use social policies to externalise obligations placed on agents in an organisation. When creating commitments, organisational agents are guided by the policies. Policies make agents' behavioural choices or obligations explicit rather than having them implicit within agent action plans. A social policy is adopted by a given agent x within an organisation, *Org* and satisfying the given constraints, *const*. The policy specifies the trigger conditions, precond, when the specified commitment, SC is obliged to be created. The social policy applies under the condition *spcondition* if provided and is expressed as follows: The SPolicy is similar to the create() predicate defined by Chesani (Chesani, Mello, Montali, and Torroni 2013). The SPolicy defines the conditions when a particular commitment should be created. The notation used is as defined in Carabelea (Carabelea and Boissier 2006):

$$SPolicy(x, org, const : precond => create(x, SC(...)), status[, spcondition]) \tag{5.2}$$

There are four main areas that social policies can address in an OJAzzIC organisation contract. Social policies govern the creation of run time social commitments between agents to enact obligations regarding authority, knowledge cultivation (belief updates, sharing intentions), Coordination (of individual and collective plans) and Creation of Organisations. We define the following notation for a social policy that results in the agents creating beliefs around creating a new organisation:

$$SPolicy(x, org, const : precond => createorg(x, y, org, obj), status[, spcondition]) \tag{5.3}$$

Coordination policies can also be used to define priorities for agents in terms of goal

adoption. Policies can also be used to specify when improvisation might be possible. Policies regarding improvisation enable the designer to specify possibilities at design time and then allow agents the autonomy at run time to choose their action plans based on the context. The policies are used by the designer to give agents guidance at run time regarding improvisation. This is so that the designer can have some control to specify any run time limitations but that agents can be given as much run time autonomy as the designer chooses. These policies for guiding improvisation could be considered as authorisation policies that give agents permission to behave in a certain way at run time. In other more sophisticated policy systems, policies are categorized as either authorisation or obligation (including coordination) policies (J. M. Bradshaw, Uszok, Breedy, Bunch, T. C. Eskridge, P. J. Feltovich, Johnson, Lott, and Vignati 2013).

5.6.3 Demonstration social policies for the OJAzzIC organisation

We now demonstrate the use of policies by providing ten social policies. These policies highlight how social policies can be used to express organisational guidelines, for example, to create appropriate commitments to ensure coordination and sharing of knowledge. The meta-model does not specify exactly the content of the policies for an organisation, however does specify that knowledge cultivation and improvisation policies should be defined for an organisation. Social policies are specified at design time, based on domain requirements. If the designer wishes for agents and organisations to dynamically select policies based on environmental conditions at run time, then these policies would need to be specified with a relationship to a particular context. The policies provided in this section form a demonstration set of policies to help an organisation to coordinate successfully.

Social policies 1 and 2 define policies for authority; social policies 3 and 4 define policies for knowledge cultivation; social policies 5 through to 8 define coordination of goal selection, awareness and plan coordination; and social policies 9 and 10 define triggers for when adhocracies should be created. In Section 5.10, we select six of these policies to describe in the specification of a solution for the scenario introduced in section 1.2.1. In chapter 7, we test the effectiveness of these policies in an implementation of a MAS based on OJAzzIC.

The predicates and definitions used in our social policies are shown in Figure 5.11. We refer to the org as *OA* as a reminder that the organisation can be thought of as a first class entity that may itself behave as an agent adopting a role in another organisation.

Create a commitment regarding obj between agent x to agent y in the context of organisation: org and under conditions: cond.	*create (x, SC(x, y, org, obj, cond))*
Create an organisation, org including agents x and y with the shared goal: obj, and status given when condition spcondition is true	*createorg (x, y, org, obj), status [, spcondition])*
Request from agent y to agent x to create a commitment to obj in org	*request (y,x,org, sc(x, y, org, obj))*
Role r1 has a relationship type with role r2 in org OA. Type is one of : *authority, peer, subordinate*	*rolerelation(r1,r2,rtype,OA)*
Agent x is enacting role r in organisation org with status of: valid (capable of playing this role), active (currently in role)	*agentrole(x,r, org, status)*
Agent x is a member of organisation org (excluding org SystOrg)	*member(x, org)*
Role r1 in org OA has a set of capabilities, C, associated with it. Agents in this role will have these capabilities	*rolecapabilityset(r1, C, OA)*
Goal g1 requires capabilityset C	*goalrequires(g1, C)*
Agent x possesses a capability, c	*possesses(x, c)*
Task t requires capability, c	*taskrequires(t, c)*
Agent x has a set of beliefs, B comprising for each belief attributes: name, value, status and type. Belief types include: domain, situation, orgstructure. status is either current or expired	*beliefset(x, [B: bel(name,val,status, type)])*
Add a belief b with value v, type t to agent x's current belief set.	*add(x, bel(b,v, current, t)*
Agent x has an active commitment in organisation OA to update belief b with a new value v	*SC (x, z, OA, update(x, b, v), active)*
Agent x sends a message to others in organisation OA if condition is true	*inform (x, OA, message, condition)*
There is some contention because a resource, *re* is being shared by agents *x* and *y*	*contention (x,y,re)*
Identify responsibility for management of a resource, *re* with a particular organisation OA	manage (re, OA)
goal g for agent, x with pg is precondition of goal, s is success condition and f is fail condition	*goal(g,x,pg,s,f)*
function returns a number representing the value of an allocation of role r to achieve goal g	*roleachieves(g,r, pg) : goal(g)*
action a achieves task t	*achieves(t, a)*
task or goal synchronisation: obj1 must be performed before obj2	*precedes(obj1, obj2)*
task or goal concurrency: obj1 must be performed concurrently with obj2	*concurrent(obj1,obj2)*
interference: obj1 cannot be performed concurrently with obj2	*interferes(obj1,obj2)*

Figure 5.11: Predicates and definitions

SPolicy1-Role-Authority-Request If an agent x acting a role *R1* in org OA is requested by another agent y in a role *R2* that is higher in hierarchy than *R1*, then agent x is obliged to commit to any request from y so long as x is capable of fulfilling the request and is available.

$$\forall\, y : agentrole(y, R2, OA, active) \wedge available(y)$$
$$\forall\, obj \in capable(x, obj) \wedge rolerelation(R1, R2, subordinate, OA) :$$
$$SPolicy(x, OA, request(y, x, SC(x, y, OA, obj))$$
$$=> create(x, SC(x, y, OA, obj)), active, agentrole(x, R1, OA, active))$$

where agent x is capable of enacting *obj* if *obj* is in the capability set of the agent or in the capability set of role r that agent x is acting in:

$$capable(x, obj) <= possesses(x, obj) \vee (\exists\, r\ \exists\, C : obj \in C \wedge$$
$$rolecapabilityset(r, C, OA) \wedge agentrole(x, r, valid))$$

SPolicy2-Role-Authority-RoleRequest If an agent x is capable of acting in role *R1* in org OA and is requested by another agent in OA to fulfil this role, then agent x is obliged to commit to adopting this role if possible[3].

$$\forall\, y : member(y, OA) :$$
$$SPolicy(x, OA, request(y, x, SC(x, y, OA, agentrole(x, R1, OA, active)))$$
$$\wedge available(x)$$
$$=> create(x, SC(x, y, OA, agentrole(x, R1, OA, active))), status,$$
$$agentrole(x, R1, OA, valid) \wedge available(x))$$

SPolicy3-Update-OrgBeliefs If agent x, a member of org OA or in a currently valid role in org OA, has an active commitment (in any organisational context) to update its own belief set with a new or updated belief that is relevant to the organisation (for simplicity, we reduce the relevance to be: the belief is in the current organisation belief set), or agent x is requested to update its own belief set with a new or updated belief that is related to the organisation, then agent x is committed to requesting an update to the organisational agent to update the organisation beliefs to be consistent with the update.

[3]This policy may be somewhat contentious as it obliges an agent to accept a request to perform a role, however, this could be representative of command and control organisations within some domains. It is possible to declare weaker policies regarding such obligations.

$$SPolicy(x, OA, \forall\, bel \in B1 \cap B2 : beliefset(OA, B1) \wedge beliefset(x, B2) \\ \wedge \exists\, SC(x, a, g, update(x, bel, newval)) \\ => create(x, SC(x, OA, OA, request(x, OA, update(OA, bel, newval))), status, \\ member(x, OA) \vee agentrole(x, r, OA, valid))$$

In the case that agent, OA is an organisational agent, then the belief set of OA is the organisational belief set.

SPolicy4-Update-AgentBeliefs if agent x is in organisation OA and OA has a commitment to update a belief, then agent x is obliged to also commit to update individual beliefs to be consistent with OA organisation beliefs.

$$SPolicy(x, OA, \forall\, bel \in B\ beliefset(OA, B) : SC(OA, z, OA, update(OA, bel, newval)) \\ => create(x, SC(x, OA, OA, update(x, bel, newval)), status, bel \in B : beliefset(OA, B) \\ \wedge member(x, OA) \wedge (member(z, OA))))$$

SPolicy5-RoleResponsibility if agent x is in a role, r then agent will commit to goals in the responsibility set for that role if requested by any other agent in the org

$$\forall\, y \in member(y, OA) \wedge agentrole(x, r, OA, active) \wedge rolecapabilityset(r, C, OA) \\ \forall\, goal(obj, x, pg, s, f), pg(obj), goalrequires(obj, C) : \\ SPolicy(x, OA, request(y, x, sc(y, x, OA, active(obj))) \\ => create(x, SC(s, y, OA, active(obj), status, agentrole(x, r, OA, valid))))$$

where a goal, g for agent, x with pg is precondition of goal, s is success condition and f is fail condition is formulated as: *goal(g,x,pg,s,f)* (Telang, Meneguzzi, and M. P. Singh 2013).

Goals are represented in a hierarchical network or tree like decomposition (Telang, Meneguzzi, and M. P. Singh 2013). We decompose goals into tasks so that where an agent is not available to adopt a role and completely achieve a goal, individual agents can coordinate activity on individual tasks within the goal to achieve the same result. A function *roleachieves(g, r, pg) : goal(g)* returns a value representing the value of

an allocation of role r to achieve goal g. The functions and predicates in Figure 5.11 include predicates relating to ordering of goals.

SPolicy6-CoordinationCooperation Agents are obliged to commit to 'help' by committing to concurrent tasks if requested by any agent, if they are capable of doing so.

$$SPolicy(x, OA, \forall\, x, y \in OA\, \forall\, obj1 \in capable(x, obj1) : request(y, x, SC(x, y, OA, obj1))$$
$$=> create(x, SC(x, y, OA, obj1, active, \exists\, obj2 : active(obj2) \wedge concurrent(obj2, obj1)), status))$$

SPolicy7-CoordinatePlan if agents are working together on a common goal, they will create a commitment to ensure coordination of the shared organisational plan toward reaching that goal

$$SPolicy(x, OA, \forall\, x, y \in OA, SC(x, OA, OA, g1) \wedge SC(y, OA, OA, g1) \wedge goal(g1)$$
$$=> create(x, SC(x, OA, OA, SP)), valid, active(g1) \wedge SC(x, OA, OA, g1))$$

where SP represents the Organisational Plan. We consider the Organisational Plan to be a SharedPlan (B. Grosz and Hunsberger 2006) between members of the organisation, using policies in the organisational contract to ensure that agents keep each other informed of changes to the plan. The organisational plan outlines the set of actions and goals to be enacted and at least a partial allocation of agents to each.

SPolicy8-CoordinateAgents if an agent is busy in an organisation with goal *g1*, that agent informs the other organisations that it is member of, that it is unavailable.

$$\forall\, org, member(x, org) :$$
$$SPolicy(x, OA, sc(x, OA, OA, g1), goal(g1), org \neq OA$$
$$=> inform(x, org, unavailable, sc(x, OA, OA, g1, active)),$$
$$status, active(g1) \wedge sc(x, OA, OA, g1))$$

SPolicy9-CreateOrg-Trigger-Goal if multiple agents from different organisations have a common goal, trigger creation of an adhocracy.

$$SPolicy(x, SysOrg, \exists\, goal(g1, x, pg, s, f) \wedge \exists\, goal(g1, y, pg, s, f) \wedge \nexists\, org1 : member(x, org1) \wedge member(y, org1) => createorg(x, y, neworg, g1), valid, active(g1))$$

The createorg function will instantiate a new adhocracy, neworg with member agents: *x,y* and the goal *g1*.

SPolicy10-CreateOrg-Trigger-Res if multiple agents from different organisations have contention of a shared resource, then they need to trigger creation of an adhocracy to manage this contention

$$SPolicy(x, SysOrg, \exists\, resource(re) : contention(x, y, re) \wedge \nexists\, org1 : member(x, org1) \wedge member(y, org1) => createorg(x, y, neworg, g1), valid, goal(g1, x, contention(x, y, re), manage(re, neworg),) \wedge contention(x, y, re))$$

5.7 Environment model

In OJAzzIC, the environment is external to the agent, however an interface is required to enable agents to perceive events in the environment as well as act upon the environment. This interface is not specified as part of the meta-model. It is assumed that at an agent level, it will be specified how agents can perceive changes in the environment and within the agent reasoning cycle, these perceptions can be converted to beliefs at the individual agent level.

5.8 Demonstrating the use of policies

In this section, a use-case scenario used to demonstrate the use of policies to create commitments between agents and to obligate agents to share information appropriately.

5.8.1 Demonstration use case description

In this section, a use case scenario is described. This scenario occurs within the rescue simulation described in Section 2.3.2 with actors that are one of 3 possible agent types: Medic, Officer and Bystander. Each actor has individual capabilities and can poten-

tially enact a role defined within an organisation. The table in Figure 5.12 specifies the capabilities allocated to each actor and each role in the organisations created initially. The default role for an actor named medic$x$ is to be in the role of a medic agent within the MedicOrg. As part of this role, the medic agent has the capabilities to locate injured and perform rescues to move the injured to the ambulance drop zone. The officer role, in the OfficerOrg has the capability to clear bystanders away from the disaster area so that the medics can enter rooms for search and rescue operations. Officer1 has an additional capability in addition to the default capability set of the officer role, so that officer1 can assist with a stretcher rescue if requested.

	Role	**Role Capabilities**	**Organisation**
	medic	locate injured, perform rescue, perform stretcher rescue	MedicOrg
	officer	clear bystanders	OfficerOrg
Actor	**Role**	**Additional Individual Actor Capabilities**	**Organisation**
medic1	medic		MedicOrg
medic2	medic		MedicOrg
medic3	medic		MedicOrg
officer1	officer	perform stretcher rescue	OfficerOrg

Figure 5.12: Roles and Capabilities of agents in rescue scenario

At the beginning of the scenario, there are a number of injured parties located in 5 rooms in the simulation space (see Figure 2.2). The 3 medic agents begin to conduct a search of the area in order to locate the injured. Medic1 finds an injured person in room A3 and identifies that the level of injury is serious and will require a stretcher rescue. Medic1 informs Medic2 and Medic3 immediately that an injured person is located in A3. Medic1 realises that she will need assistance so sends a message to medic2 to ask if he is available to help with the stretcher rescue of the seriously injured patient in room A3. Medic2 responds to say yes he could potentially help with this rescue. Medic2 and Medic3 are also conducting a search of the area and identify and share beliefs regarding the location of injured people found in other rooms. All medic agents keep each other informed regarding injured people found during the search. The medic agents prioritise stretcher rescues of seriously injured patients. Medic3 sends a message to medic2 regarding a rescue plan and asks if medic2 is potentially able to help. Medic2 hasn't yet committed to the rescue proposed by medic1 so agrees in principle that he could also potentially help medic3 with the proposed rescue. Upon receiving confirmation from medic2 of a potential shared intention, medic1 suggests that medic2 and medic1 now create a commitment to the potential rescue in A3. Medic1 and medic2 both commit and agree to go ahead with this plan and intend to meet at the door of room A3. Medic1 remains in room A3 with the patient.

Three (incomplete) scenarios are now explored to complete the rescue of the patient

in room A3. These pathways describe situations requiring plan elaboration/revision, improvisation or mutual adjustment of individual plans to fit in with others. Agents are also required to keep each other informed of relevant beliefs and intentions.

Possibility 1: En route to room A3, medic2 finds another injured person in room C4 who needs urgent medical assistance, so medic2 stops to help and sends a message to medic1 to say "I am no longer available". Medic2 is now relieved of his prior commitment to medic1 and gives assistance to the injured person in C4. Medic1 now needs to find a new agent to help with the stretcher rescue planned in room A3. Medic1 leaves the room and notices that medic3 is nearby and asks medic3 to help carry the stretcher with her. Medic3 negotiates with medic1 regarding how the rescue will be completed (they agree to adopt specific distinct roles of stretcher pickup (medic1) and stretcher carrier (medic3)). Medic1 informs medic2 that medic3 will now be helping with the rescue. The stretcher rescue of the patient from room A3 is then completed by medic1 and medic3. As soon as the rescue is completed, medic2 is informed that it was successfully completed. Medic3 then turns attention to the previous potential plan to rescue the injured person in room C4 with Medic2 and they both commit to this plan and begin that rescue.

Possibility 2: Medic1 is waiting in room A3 and notices that officer1 is nearby so realising that time is important, decides to ask officer1 to help carry the stretcher instead of waiting for medic2. Medic1 sends a message to medic2 to indicate that she is no longer needing medic2 and instead medic1 delegates the stretcher carrying role to officer1. The stretcher rescue of patient from room A3 is then completed by medic1 and officer1. As soon as the rescue is completed, medic2 and medic3 are informed that the rescue was successfully completed. Medic2 and medic3 have already committed to and begun the previously identified rescue in room C4.

Possibility 3: Medic1 waits in room A3 until medic2 arrives. Medic3 proposes a rescue plan with medic2 to rescue the injured in A3. Medic1 sends a message to medic3 informing of the intention of medic2 and medic1 to complete the rescue in room A3. Medic2 arrives at the front door to room A3 and can see that medic1 has the stretcher ready, so in order to complete the plan, without further discussion regarding roles, medic1 adopts the support role of stretcher carrier. Then medic1 and medic2 complete the stretcher rescue of the patient from room A3. As soon as the rescue is completed, medic3 is informed that the rescue was successfully completed. Medic1, Medic2 and Medic3 all begin to propose, negotiate and commit to plans for subsequent stretcher

rescues that are needed.

The above scenarios are used to conceptually demonstrate our use of OJAzzIC to ensure coordination and knowledge sharing within and between organisations. We create multiple organisations, each autonomous but acting in a context of awareness of others in the network. The entire global management system organisation has the goal to resolve the disaster - rescue the injured, give first aid, transport to hospital. The main agencies are allocated sub goals in the high level plan, based on their service for example the goals of the MedicOrg agency are to provide immediate medical care and casualty transport to hospital; the goals of the OfficerOrg agency are to manage hazards, protect the site and clear away bystanders.

Figure 5.10 provided a high level goal tree for the disaster rescue scenario. This would be used by agents to adopt goals according to the roles initially allocated to agents at design time. Goal G5 Rescue injured is associated with agents in the role of a Medic. Goal G1 Ensure safety on site is associated with agents in the role of an Officer. Following the initiation of the scenario, the initial response would involve each agency dispatching a number of available agents and emergency response vehicles to the disaster zone. Goals may be further described as comprising one or more tasks. Tasks may then be adopted by agents in a SharedPlan or Goals may be adopted and enacted by an agent or an organisation that is fulfilling an agreed role. For example, Goal G7: Perform rescue can be decomposed to include perform complex rescue or perform simple rescue. In a complex rescue, the Medic agents would need to establish who/how these tasks are actioned by creating a run-time SharedPlan in order to establish the detailed plan and make explicit agreements and commitments regarding the execution of the plan.

On initiating the scenario, 4 agents are created as indicated in Figure 5.12 and allocated to belong to the main disaster management system organisation (SysOrg). Each specialised agent in the environment is also allocated to a default fixed organisational agency based on their agent type. The specialised agents cannot leave their parent agency organisation. The organisational agencies include: medicalAgencyOrganisation (MedicOrg), securityAgencyOrganisation (OfficerOrg).

As the scenario progresses, an agent may choose to form/join new organisations of agents representing short term adhocracies that will occur to help with communication, coordination and control. Following the creation of an adhocracy, this organisation can exist until the end of the life of the main system organisation (SysOrg), or explicit policies could be created to dismantle the organisation based on a trigger to suggest it is no longer useful. In the example that follows, we presume the former and do not

describe an explicit dismantling policy.

The system organisation, SysOrg is the super organisation and contains two sub-organisations: MedicOrg and OfficerOrg. This means that all agents in MedicOrg and OfficerOrg are also in SysOrg. In practice, the communication flow is such that there is one nominated leader or spokesperson agent in each of the sub-organisations who will attend to formal communication between organisations at the super-organisation level and then be responsible for passing on relevant knowledge into each sub-organisation. At the superorganisation level - SysOrg, the high level plan is known and each agent(sub-organisation) is aware of its allocated goal in the goal tree. This means that an agent in OfficerOrg will be aware of the more detailed plans agreed within the MedicOrg organisational but also aware that there is a high level plan at a higher level in SysOrg.

Initially, there is no intersection or direct formal communication between individual members of MedicOrg and OfficerOrg other than within the Sysorg plan allocation of high level goals to MedicOrg and OfficerOrg. As the scenario progresses, situations will arise requiring individual members to agree to coordinate to achieve goals and/or share resources. When that occurs, the individual members create new short term organisational adhocracies. Each adhocracy has an identifiable common interest -e.g. shared resource, shared knowledge goal, shared coordination goal. Explicitly representing each organisation as a first class entity provides a mechanism for agents to identify which other agents are involved and thus facilitates appropriate knowledge sharing communication. In the next section, we do a paper walk through of part of the unfolding scenario to illustrate the management of coordination and how policies ensure appropriate knowledge sharing and cultivation of relationships between organisations.

5.8.2 Using policies and commitments

Roles are defined within the organisation using relationship definitions to reflect the structural hierarchy and define permissions and authority regarding which agents may request other agents to accept allocated tasks. For example *rolerelation (medic, officer, authority, SysOrg)* defines that within the SysOrg, an agent in the role of medic has an authority relationship with an agent in the role of officer. When an organisation is created, agents will instantiate a social contract and an information contract. The information contract defines the organisation's relevant current agreed beliefs and obligations to share these within the organisation. The social contract includes a list of member agents, roles, relationships and the organisational plan.

In order to demonstrate by example, we choose a snapshot in time during the scenario to focus upon. We will highlight some of the social policies introduced in

section 5.6.2 and how they ensure appropriate interactions between agents.

Because medic1, medic2 and medic3 are all in the MedicOrg, when medic1 finds the injured person in room A3 initially, Spolicy3 and Spolicy4 result in medic1 sharing the information about the injured in A3 with medic2 and medic3. Then when medic1 requests if medic2 might be available to help with a stretcher rescue, Spolicy5 obligates medic2 to consider this as part of the medic role. Spolicy5 also obligates medic2 to consider helping medic3 with the second rescue request. When medic1 and medic2 both commit to the plan to work together on the stretcher rescue in room A3, sPolicy7 is responsible for each agent committing explicitly to this SharedPlan.

Considering possibility 1, medic2 changes his individual plan fulfilling his role as a medic when he discovers the injured party in room C4. Spolicy8 obligates medic2 to inform medic1 and medic3 that he is no longer available. As a result, when medic1 requests help with carrying the stretcher from medic3, medic3 is obligated by Spolicy5 and then Spolicy7 to agree and create a plan to perform this rescue. When the rescue is complete, each of medic3 and medic1 are obligated by Spolicy3 to share their new belief with others in the organisation. In this way, medic2 is informed and updates his own individual beliefs to be in line with the organisation belief that the rescue has been completed.

Considering possibility 2, Spolicy6 obliges officer1 to accept the request from medic1 to help with the stretcher, Spolicy9 will oblige medic1 and officer1 to create an adhocracy and then Spolicy8 obliges the agents to agree and commit to a SharedPlan regarding the stretcher rescue. Spolicy7 obliges medic1 to inform medic2 that he is no longer needed. When the rescue is complete, Spolicy3 ensures that this information is shared with medic2 and medic3. If it had been the case that medic3 had been considering adopting a goal to do this rescue, the new information update would mean that medic3 would reconsider his goals and drop this as a goal to consider.

Adhocracies enable the flow of information across the wider network of organisations. For example, if the scenario had involved more than one officer agent, then when officer1 was aware of a new updated belief, such as a rescue being completed, officer1 would be obliged by Spolicy3 to inform other officers who would accept this new information and update their own individual beliefs due to Spolicy4.

This scenario discussion does not demonstrate the use of a dynamic role model as agents are allocated to roles at the beginning of the scenario (at design time). It also does not demonstrate a need for policies 1 and 2 as these social policies relate to a situation where agents might be allocated to roles as part of the execution model rather than at design time.

In Section 5.6.2 default policies for knowledge sharing in the information contract are provided. The OJAzzIC meta-model does not specify how strict these policies need to be. It is up to the designer of the system, based on requirements to define obligations as policies to suit the situation and the domain. In Section 5.10, a MAS based on the OJAzzIC meta-model is specified and policies are described for this system relating to the emergency management scenario used in 2.3.2.

5.9 Evaluating OJAzzIC against requirements

In this section, the user stories introduced in section 2.3.2 and section 2.3.3 are used to validate the OJAzzIC meta-model by traceability to requirements. Figure 5.13 summarises how OJAzzIC addresses the requirements specified by the agent stories in section 2.3.3.

Agent User Story	Ability to address this requirement
1 adopt goals, work with others	agents can adopt goals based on roles or capabilities, SharedPlans can be created
2 be aware of others and mutually adjust to fit in with others	agents can adjust on the fly, relying on goals defined at design time and policies to guide improvisation and obligations regarding sharing beliefs and commitments regarding SharedPlans
3 prioritise organisation goals	can be achieved by policies or could be hard coded at an agent level
4 rescue injured	with hard coded action plans defining agents capability to achieve goals individually or as part of a role
5 coordinate to perform stretcher rescue	coordination based on goal design and creating SharedPlans to coordinate individual actions
6 perform tasks autonomously	agents perform actions based on a role or based on individual capabilities; at design time, define how agents achieve tasks, at design time can specify multiple goal paths
7 help others if I can	agents are allocated roles based on capabilities, individual improvisation possible if policy written at design time
8 share information about my intentions and actions	needs to be hard coded into agent actions or specified in policies at design time

Figure 5.13: Agent User Story Requirements addressed by OJAzzIC

The OJAzzIC meta-model includes an organisation model that does explicitly:

- identify goals - individual or organisation goals
- allow for multiple solution paths to be defined at design time to enable adaptation at run time
- identify who is in the organisation

- capture information about the structure of the organisation including roles and responsibilities
- specify capabilities of individual agents and which roles agents can enact and which roles can achieve goals

OJAzzIC uses policies written at design time to provide mechanisms to:

- create organisations at run time
- obligate agents to share information regarding the organisation structure
- obligate agents to share (update) information regarding beliefs (e.g. location of injured, completed rescues)
- identify which agents need to be informed of updates regarding the situation and regarding an agent's intentions
- allow multiple agents to dynamically plan and commit to interdependent coordinated plans to achieve one goal
- allow agents to improvise beyond the organisation design (for example, help out with a task outside of role)

OJAzzIC adopts SharedPlans created based on explicit commitments between agents at run time so that agents can agree to adopt a goal and:

- commit to goals that contribute toward an incomplete plan
- revise plans by changing adopted goals if the context changes

In section 5.11, OJAzzIC is evaluated further by comparison with a number of other meta-models. In Section 5.10, the OJAzzIC meta-model is described using an example model specification for the emergency management simulation system used in section 2.3.2.

5.10 Simulation MAS Specification

In this section, a MAS is specified, based on the OJAzzIC meta-model. This system specification is used to link the meta-model to an implemented simulation system. This MAS system is specified with a focus on identifying the policies and operations, based on OJAzzIC that would be implemented in a executable system. This system specification describes a MAS capable of responding to the emergency management situation described in Sections 2.3.2 and 5.8.

Functionally, the organisation goal of the system is to resolve a disaster. This goal is decomposed into two sub-goals: ensure safety on site and rescue injured. There are 2 main organisations: one for medic agents and one for officer agents. Agents enacting the

medic role (Medics) in the medic organisation are responsible for achieving the objective to rescue the injured and the agents in the role of officer in the officer organisation are responsible for the objecting of ensuring safety on site. Medic agents have the capabilities to control virtual robots to pick up and rescue an injured 'person' in the simulation. If the injured is seriously injured, then two Medic agents work together to perform a stretcher rescue.

There are 6 agents in this specification as shown in Figure 5.14. There are three different types of agents: Medic, Officer and Bystander agents. By default, Medic agents enact the Medic role and Officer agents enact the Officer role. All Medic and Officer agents have the capability to move to a location, perceive (observe) injured in their immediate environment and to communicate with other agents by sending messages.

Details regarding operations of bystander agents are not provided. Bystander agents are not in any organisation and do not enact roles. When the system is implemented, a number of Bystander agents can be created. Bystander agents have limited capabilities. Bystander agents randomly move to a location and receive messages from Medic or Officer agents informing them to move to and remain in a new location. Medic agents have authority over Officer agents and therefore can delegate tasks to an officer agent.

This specification is partially specified formally using the Z notation (Spivey 2001) in Appendix A. The operations relating to individual agent capabilities are not specified, but assumed to be included in the specifications of individual agent types. The specification provides an overview focusing on the main operations needed in an executable MAS based on OJAzzIC.

Organization	Agent names	Agent Type	Role enacted
sysOrg, medicOrg	medic1, medic2, medic3	Medic agents	Medic
sysOrg, officerOrg	officer1, officer2, officer3	Officer agents	Officer
	feye1, feye2, feye3, ajax1, ajax2, ajax3	Bystander agents	

Figure 5.14: Organisations and Agents in simulation system specification

The Z schema provided in Appendix A shows some of the main operations that would be necessary in order to run through the rescue response scenario using the OJAzzIC meta-model. Agents need to be able to perform operations such as *create_Shared_Plan*, *process_individual_and_organisational_attitudes*, *enact_goal*, and *Create_a_Commitment*. Agents also need to be able to adopt and act upon organisation policies (*Adopt_Org_Policy*)

when they are obliged to, for example, to inform other agents when they adopt a goal if that is an organisation policy.

This scenario and solution schema declares agents in the OJAzzIC-based organisation are capable of organisational reasoning including awareness considering organisation goals and adopting organisation policies. Agents adopt goals to enact based on possessing individual capabilities and/or enacting a role associated with the goal. The specifications also shows that agents collaborate by creating a SharedPlan to work together to achieve a goal that requires coordination.

In the *init_policies* operation, six policies are specified. These policies are specific implementations of policies selected from the policies in section 5.6.2 to be applied in the rescue scenario. The following list describes each policy in the specification:

- spolicyA specifies that an officer agent will be obligated to commit to a medic agent if requested to help with a task that the officer is capable of achieving. This implements SPolicy1.
- spolicyB specifies that any agent in an organisation is obligated to share new beliefs with other agents in that org. This implements SPolicy3.
- spolicyC specifies that when an agent is in an organisation and creates an commitment to achieve a particular objective in that organisation, then that agent is obligated to inform others in the organisation of their intention. This is a policy regarding sharing beliefs about intentions, drawn from both SPolicy3 and SPolicy7.
- spolicyD obligates an agent, agent1 in an organisation to commit to another agent in the organisation who requests help with a particular objective where agent1 is enacting a role responsible for the objective. This implements SPolicy6.
- spolicyE states that when an agent has an existing commitment in an organisation and is busy, then that agent is obligated to inform other agents in that organisation that it is currently busy. This is implemented by creating SharedPlans to make commitments explicit between agents. This is based on SPolicy7 and SPolicy8.
- spolicyF is a policy to create a new adhocracy, a new organisation if there are 2 agents with a shared goal who are not already in an existing organisation. This implements SPolicy9.

The *enact_goal* operation shows that when an agent can partially achieve a goal with another agent who can partially achieve that goal, then these agents will create a SharedPlan and if they are not already in an organisation, create a new organisation. The *adopt_Goal* operation shows that if the policyB exists in an organisation, then the

agent will inform others in the organisation when they have adopted a goal.

Figure 4.10 described a number of predicates at a conceptual level relating to an organisational group. These are implemented in the demonstration MAS with some modifications. The operations are not presented with reference to a temporal value for time, so the *Create_org* operation aligns with the predicate $RM(gr, t)$ defined in Figure 4.10 without any representation of the current time, t. The predicate $AReltoB$ is implemented using the operation : *member_Agent_in_Org* to confirm that each of A and B are in an organisation. Alternatively, the operation $agent_is_member_of(A)$=$agent_is_member_of(B)$ can be used to test that each agent is a member of the same organisation. The predicates $Add(x, gr)$ and $Update(x, RM)$ are both referring to an agent, x being added to an organisation. In OJAzzIC, there is not a separate concept of a group. Groups for coordination and collaboration are organisations in OJAzzIC. This is implemented using the operation *Add_Agent_to_Org*. The predicate $Focus(b, GR)$ represents an operation to establish that a belief b is related to group GR. In the demonstration system, this is not implemented. Rather it is assumed that that all beliefs can be shared to all organisations that the agent is a member of, so this can be implemented using the operation *member_Agent_in_Org* to establish the list of organisations that an agent is a member of, and then generate a list of all agents who are members of each organisation using the operation *member_Agent_in_Org*. This is a simplification that would need to be addressed in a more complex simulation where there were a greater number of organisations.

5.11 Comparing OJAzzIC to other models

Some of the major design decisions in the OJAzzIC meta-model were to incorporate an organisation meta-model, an agent meta-model, a functional meta-model, and a coordination model specified using policies. An environment model is considered to exist outside the OJAzzIC meta-model with an interface to provide percepts to agents within the organisation.

In section 4.2, requirements from our simulation system scenario, introduced in Chapter 2.2.5 were used to examine how different comparison meta-models could be used to attempt to design a multi-agent system to engage with this scenario. We now compare these with OJAzzIC. Figure 5.15 summarises the ability of each meta-model to address the main agent user stories described in section 2.3.3.

Challenges for each meta-model include how to establish agent awareness and how to enable agents to coordinate their individual behaviour at run time to achieve the

Agent User Story	OMACS	JaCaMo+	OperA+	SharedPlans	OJAzzIC
1 adopt goals, work with others	p	p	p	p	y
2 be aware of others and mutually adjust to fit in with others	p	p	p	y	y
3 prioritise organisation goals	y	y	y	p	y
4 rescue injured	y	y	y	y	y
5 coordinate to perform stretcher rescue, improvise if necessary	p	y	p	y	y
6 perform tasks autonomously	y	y	y	y	y
7 help others if I can (even if outside my role responsibilities)	n	n	n	p	y
8 share information about my intentions and actions	p	n	p	p	y

Legend	Extent that User Story requirement is met by model
p	partially: by hardcoding into roles or action scripts, not explicit constructs
y	yes
n	no

Figure 5.15: Agent User Story Requirements addressed by each meta-model

completion of tasks that require multiple agents to work together (without relying on scripts defined at design time). JaCaMo+ and OperA+ provide features based on social contracts between agents and separate agent internals from an interaction layer that is specified externally. The interaction layer is used to specify protocols to direct agents in their run time behaviour.

In JaCaMo+, it is not possible for agents to negotiate an agreed plan including coordination of individual agent plans in order to reach a goal involving interdependent activities. SharedPlans provides for dynamic coordination of agent plans to enable dynamic planning. This level of coordination is not provided in the other meta-models under examination. SharedPlans incorporates an explicit level of interaction around intentions and sharing of intentions, however SharedPlans does not incorporate any organisational knowledge. Interactions in JaCaMo+ are managed through an artifact, external to the agent. The artifact in JaCaMo+ is referenced based on protocols specified at design time and agents rely on the system to notify them rather than receiving direct communication from another agent.

OperA+ and OMACS both use capabilities as a way of allocating agents to roles, but both assume that it will always be possible to find an agent possessing all the capabilities required to enact a role fully. Improvisation outside of a role description is

not a feature of any of these models.

OJAzzIC separates the role descriptions from the functional description of organisation objectives into goals and sub-goals. OJAzzIC assigns capabilities to individual agents based on their type. This enables agents to adopt roles based on their capability. Agents in OJAzzIC can also adopt individual intentions to reach a goal based on capabilities (outside of role allocated). This is a distinguishing difference between OJAzzIC and other meta-models. This provides some flexibility at run time that is not available in other approaches.

OJAzzIC uses explicit commitments similar to JaCaMo+ and similar to contracts in OperA+. Commitments make it possible for agents within the organisation to have some awareness of others. Additionally, policies in OJAzzIC are used to obligate agents to generate appropriate commitments to share information with other agents regarding intentions and beliefs. OJAzzIC enables agents to have awareness of the organisation structure and intentions of others so that agents at run time can mutually adjust and change their own individual goals to fit in with others. JaCaMo+ offers flexibility to agents at run time, but JaCaMo+ agents do not have awareness at an individual level in the same way that OJAzzIC agents would because the information is stored in artifacts external to the agent.

The analysis presented in this chapter contributes to validation of the OJAzzIC meta-model. The requirements articulated earlier in the thesis have been used to predict behaviour. Based on these predictions, a number of meta-models have been examined in terms of the components included in the meta-model and thus, their ability to achieve this behaviour in an implemented model. In the following chapters, further scrutiny is applied to the OJAzzIC meta-model by describing the design and implementation of a MAS using the meta-model. In the next chapter, the OJAzzIC design process is described.

5.12 Conclusion

In this chapter, the OJAzzIC meta-model is described including the meta-model components as well as a specification of an execution model based on OJAzzIC. We propose that based on the components included in this meta-model, MAS can be created so that at run time agents have some flexibility in their behaviour. The meta-model specifies the components in an OJAzzIC organisation and specifies how agents deliberate, guided by policies specified in the organisation model. Policies can be specified to obligate agents to share information within the organisation and to allow some agent

improvisation at run time.

The use case scenario and user stories provided in Section 2.3.2 and Section 2.3.3 highlight the requirements for agent behaviour that we are trying to address with the OJAzzIC meta-model. In particular, we are concerned to address requirements such as autonomy, flexibility in goal adoption, awareness, sharing beliefs and intentions, coordination and improvisation. These requirements have been highlighted in section 5.8 and considered in the choice of entities in the meta-model.

In Figure 5.13, there is some identification of components of OJAzzIC that address these requirements. In Section 5.11, the same requirements in 2.3.3 were used to compare a number of meta-models.

We have proposed that using the OJAzzIC meta-model, it is possible to design a MAS that can behave with some flexibility at run time. In Section 5.10, a specification is provided for a MAS model based on OJAzzIC. In chapter 6, the design process for creating a MAS using OJAzzIC is demonstrated using the scenario and specified demonstration system and then in chapter 7 an implementation of a MAS based on this specification is discussed. We examine the performance of the implemented system against the predicted behavioural requirements being addressed by the meta-model. Our experimentation also provides an opportunity to test the benefit of suggested policies.

Chapter 6

OJAzzIC Design Process

6.1 Overview

In this chapter we focus on the design process. We discuss design issues and then describe the process adopted for the design of organisation-centered MASs using the OJAzzIC meta-model. We begin in section 6.2 by presenting a methodology for the design process[1]. We use an adaptation of O-MaSE (S. A. DeLoach and Garcia-Ojeda 2010), an organisation-based multi-agent software engineering methodology, applied to the OJAzzIC meta-model introduced in chapter 5. We use a running case study to show examples of each model that would be created in the OJAzzIC design process in section 6.3. In chapter 7, we elaborate and demonstrate the design process by discussing an implementation of a MAS based on this process.

6.2 Adopting a design methodology

To highlight our design considerations and illustrate our design process, we use a simulated emergency incident response scenario based on the scenario introduced in Section 2.3.2 and elaborated upon in Sections 5.8 and 5.10. Mindful of the requirements of adaptability, flexibility and improvisation, we adopt steps in the design process based on an adaptation of the O-MaSE methodology.

[1]The methodology outlined in Chapter 6 with an example of how OJAzzIC could be used to represent, design and implement a simulation system was described and presented at the COIN workshop held at PRIMA in 2014 K. Keogh and E. A. Sonenberg (Dec. 2014b). "Designing Organization-Aware Agents using OJAzzIC". in: *Proceedings of 16th International COIN Workshop, PRIMA 2014*. Ed. by A. Ghose, N. Oren, P. Telang, and J. Thangarajah followed by publication in 2015 K Keogh and E A Sonenberg (2015). "Designing for Planned Emergence in Multi-agent Systems". In: *Lecture Notes in Computer Science* 9372. Ed. by A. Ghose, N. Oren, P. Telang, and J. Thangarajah, pp. 97–113

6.2.1 Addressing requirements in the design process

Considering the dynamic nature of domains under consideration and our desire for flexibility to improvise at run time, a number of issues must be considered at design time. The particular requirements that influence the design issues that we have considered include how we achieve the desired flexibility in terms of coordination, mutual adjustment of plans and goals, improvisation and knowledge sharing. As part of the design process adopted, a number of questions can be highlighted for consideration when models are being developed at design time. The answers to these questions would influence how much is specified at design time and how much flexibility at run time is anticipated.

Design questions
Does the specification at design time allow for flexibility and improvisation later?
How is coordination between agents achieved and at what level (e.g. task specific, role specific, or policies within an organisation, or at a system level)?
What type of adaptation is required in the system?
How complete and adaptable are roles specified during design?
How much autonomy can be given to agents in terms of choosing tasks outside of a role specification?
Which potential organisations and adhocracies can be identified at design time?

Mindful of these questions and the questions raised by Fornara (Fornara and Balke-Visser 2018), listed in section 3.4.1, the adopted design process is focused upon creating models to specify the goals, the organisation membership, agent types and capabilities, define roles and specify policies within the organisation.

6.2.2 The design process

The OJAzzIC design process is outlined below with five main tasks. These tasks result in a specification or refinement of each of the meta-models defined in the OJAzzIC system[2]. The design tasks are:

- Define the functional goal model
- Define the organisation model
- Define the agent capabilities model
- Define the role model for the organisation
- Establish social policies in the coordination model

Methodologies for the engineering of MAS are discussed in section 3.4. The O-

[2]The environment meta-model is not specified as part of the design process as this is external to the OJAzzIC system, although when designing agent actions, there would need to be an interface between agents and the environment so that actions can act on the environment and perceive inputs from the environment.

MaSE methodology has been proposed as a customizable approach to adaptive MAS design (S. A. DeLoach 2014; S. A. DeLoach and Garcia-Ojeda 2010). O-MaSE provides a set of agent development tools and a customisable approach to building complex, adaptive MAS. Figure 6.1 lists the O-MaSE tasks adapted to produce models based on the OJAzzIC meta-model.

O-MaSE task	**OJAzzIC component produced**
Model goals	goal decomposition tree
Refine goals	task specification - synchronisation, concurrency, dependencies
Model domain	domain level tasks and interfaces
Model organisation interfaces	organisation models and organisation instances
Model roles	role model (relationships and authority)
Define roles	role descriptions, capabilities required and responsibilities
Define role goals	role-goal responsibilities
Model agent classes	agent types, capabilities possessed, authority relationships
Model protocols	autonomy, improvisation and control policies
Model policies	social policies
Model plans	verified plan design as paths in goal tree
Model capabilities	agent capabilities model
Model actions	action rules at agent level
Generate code	code

Figure 6.1: O-MaSE key tasks applied to produce OJAzzIC components

As the OJAzzIC design methodology is adapted from O-MaSE, we name the equivalent task in O-MaSE in brackets for comparison purposes in the descriptions provided below. Tasks in O-MaSE map well to produce corresponding OJAzzIC components and are consistent with our approach. The design approach is iterative so refinement and review may result in repeating steps. There are differences between O-MaSE and OJAzzIC in the use of goals, roles and agent capabilities at run time. In O-MaSE, goals are used to define the objectives of the organisation, whilst roles are used to define abstract positions within the organisation that can achieve a given goal or set of goals. In O-MaSE, unlike OJAzzIC, there is no provision for splitting roles. In OJAzzIC, the design of a problem solution includes into two distinct design components: the problem design represented as a set of goals and tasks and the resources available described in terms of agent types and organisations. By keeping these distinct, we aim for more flexibility at run time. In OJAzzIC, a direct relationship between roles responsible for a goal and agents available to adopt roles is not presumed. If there is not a direct match between the goals and the available agents' assigned roles, then the matching of goals to agents can emerge at a lower level based on agent capabilities and the capabilities required to achieve a task or sub-goal. In Section 6.3, we describe each stage in more detail using our case study example.

OJAzzIC Design Methodology Tasks

1. Define the Functional Goal Model (O-MaSE:Model goals, Refine goals, Model domain, Model plans, Model protocols)
   - Create a high level goal decomposition of system objectives.
   - Break objectives into tasks that may be achieved by agents individually.
   - Where possible identify multiple alternatives (plans) to achieving an objective[3].
   - Identify dependencies between tasks and objectives, paying attention to requirements of synchronisation - e.g. before(task1,task2), concurrent_tasks(task1,task2).
   - Identify autonomy and control associated with each objective or task. Identify for each task or objective if it must be associated with any particular role.
2. Define the Organisational Model (O-MaSE:Model organisational interfaces, Model roles)
   - Identify long term organisations (including sub-organisations) agents may belong to.
   - Define default agent types and domain roles associated within each organisation.
   - Identify any inter-agent relationships.
3. Define the Agent Capabilities Model (O-MaSE:Define roles, Model agent classes, Model capabilities)
   - List capabilities to be given to particular agent types.
   - Identify capabilities required to achieve each task and thus required to fulfil each domain role.
4. Define the Role Model (O-MaSE:Define role goals)
   - Identify roles that agents of a particular type may be able to adopt within each organisation (may include domain roles and structural roles). e.g. Medic, Leader
   - Identify responsibilities associated with roles within each organisation. Map organisational roles to objectives they are responsible for.
   - Identify role relationships (e.g. dependency, authority, right to delegate etc.).
5. Define the Coordination Model: Establish Social Policies to be adopted within the run-time organisational contract (O-MaSE:Define protocols, Model policies)
   - role adoption responsibilities. e.g. Medic will prioritise locating injured then rescuing injured

[3]SharedPlans created at run time to signal an agreement between agents to achieve an objective can be based on these plans.

- knowledge sharing obligations. e.g. Medic will tell other Medics when an injured agent has been located or a rescue has been completed
- organisational adhocracy creation triggers e.g. in rescue domain, if inter-agency coordination is required, a new adhocracy will be created to ensure appropriate communication and coordination occurs.
- obligations between agents to establish shared organisational plans for coordinated tasks before goal actions are adopted.

6.3 Demonstrating the design process

In this section, we reference the incident response system introduced in section 1.2.1 and discussed in sections 2.3.2, 5.8 and 5.10. The incident response system is used to demonstrate decisions made during the design process and illustrate by example the models created. In chapter 7 we expand on the design to transform it into an implementation. In this section, where it seems helpful, some code examples are used from the implementation in GOAL discussed in more detail in the next chapter.

6.3.1 Define the functional goal model

The first[4] stage in our adopted methodology is to define the goal model. This involves the following steps:

- Create a high level goal decomposition of system objectives and where possible, break objectives into tasks that may be achieved by agents individually. This goal decomposition can initially be represented visually as a tree with the main objective as the root of the tree. Goals can be decomposed into small sub-goals and leaf goals are tasks that can be completed by an agent;
- identify if multiple alternatives plans exist to achieving an objective, these should be visible as alternative paths in the tree;
- identify dependencies between tasks and objectives, paying attention to requirements of ordering and synchronisation. Where ordering of tasks is important, specify this in the design visually in the goal tree so that it can be translated into code later appropriately (For example, using parallel lines to join tasks to indicate that they occur in parallel and using an arrow between tasks to indicate precedence);
- translate the tree design into predicate-like specifications for each task or goal where ordering is important. For example if task 1 must be completed before

[4]It is possible to revisit any of these design tasks in a process of iterative refinement

task 2, this can be represented as: *before(task1, task2)*. If task 1 and task 2 must be completed concurrently, then this can be represented as: *concurrent (task1, task2)*.

- identify autonomy and control associated with each objective or task. Think about any policies that might apply to this task (particularly regarding improvisation);
- identify for each task or objective if it must be associated with any particular role(s) exclusively;
- based on the objectives and tasks, design agent level specifications for how to achieve these.

In Figure 5.10 a goal decomposition tree for the incident response scenario was provided. There are two major goal objectives in our system - ensure safety on site and rescue injured. Ensure safety can be broken into three tasks: clear away bystanders, find fights and clear away fights. The clear away fights task is dependent on the find fights task that must occur beforehand so *before(findfights, clearfights)* is true. The rescue injured goal may be reached by either of two tasks: perform simple rescue (one Medic agent works alone to pick up the agent and move them to the ambulance, or perform complex rescue (two Medic agents work together to use a stretcher to move an injured agent to the ambulance). This means that *concurrent(rescueinjuredOnStretcher(agent1), rescueinjuredOnStretcher(agent2))* is true.

A design decision should be made regarding autonomy and initiative. The question should be asked for each task - can it be actioned by any agent with the necessary skills (capabilities) or must it be adopted only by one specific type of agent or an agent fulfilling a particular role? In the case of the clear blocking bystanders task we can allow such initiative. The system may be configured to treat the task of removing blocking bystanders as a task that is only allocated to Officers, either as part of their role, or as a task delegated by request from a Medic, or as a task that Medic agents may also adopt by initiative if they are available. In our system, the feature allowing a Medic agent to clear bystanders using their initiative if available to do so may be turned on or off.

The goal tree paths can each correspond to a plan of actions to reach a landmark objective. We adopt the term landmark to refer to each high level objective goal states, following the approach used in OperA (Weigand, V. Dignum, J.-J. Meyer, and F. Dignum 2003)[5]. The subgoals and tasks in the pathway can be completed in order to

[5]The implementation of the demonstration MAS incident response system was built based on extensions to an existing system that was initially designed with OperA+, so the objectives are referred to as landmark (Lmk) objectives in code.

achieve that objective. Each objective landmark has an associated task list defined as states that must be accomplished in order to reach the landmark state. The capabilities required in order to achieve these tasks are also associated with the objective. This can be represented as: *landmark(Objective, SubTaskList, CapabilitiesRequired)*. For each objective, it should be possible to identify at least one individual action plan for how that objective can be achieved by an agent *plan(planID, landmarkId-Objective, subTaskStates, taskList)*. The plan contains a list of states that must be achieved toward the final objective and a list of tasks or goals that will lead to successfully reaching the objective state.

Objective	Capabilities Required	sub-goal states
injuredLocatedLmk	$at(Ag, _)$ $injured(Ag)$	checkedRooms, injuredLocated
injuredRescuedLmk	$holds(Ag)$ $injured(Ag)$ $at('DropZone')$	injuredLocated

Figure 6.2: Example objectives with details of required capabilities and sub-states

Figure 6.2 shows two objectives in our incident response system, the capabilities required in order to reach this landmark and the sub-states that are achieved whilst reaching each landmark objective. In the examples provided, capabilities are described in terms of beliefs. at(Ag,Location) is used to represent that a particular Agent is capable of holding a belief that Ag is at a particular location. By extension this represents that the agent is therefore capable of creating such a belief. at(Location) represents the belief that the agent itself is at Location, so the agent is capable of taking action to make at(Location) true ; holds(A) indicates that an agent believes it is holding a particular item A. This means that the agent must have the capability to pick up an item; injured(A) represents that the agent believes that A is injured. Figure 6.3 shows similar information expressed more formally in code[6] for a number of landmarks in the incident response system. Landmarks are defined using a rule of the form *landmark(aLmk, [list of sub-goals that lead to the landmark], [list of capabilities required to achieve this landmark])*[7] state.

Plans are associate with reaching a landmark objective and are defined in terms of high level landmark states that lead to reaching that objective, defined in GOAL code as follows: *plan(planID, landmarkId-Objective, subTaskStates, taskList)*. For example, the following rule specifies that the plan to located injured, *injuredLocPlan* is associ-

[6] This is GOAL code taken from the implementation discussed in Chapter 7

[7] We adopt the term landmark to refer to each high level goal state, following the approach used in OperA (Weigand, V. Dignum, J.-J. Meyer, and F. Dignum 2003)

$landmark(injuredLocatedLmk, [checkedRooms, injuredLocated], [at(Ag, _), injured(Ag)]).$
$landmark(injuredRescuedLmk, [not(injured(_))], [holds(Ag), injured(Ag), at('DropZone')]).$
$landmark(blockingBystanderRemovedLmk, [blockingBystanderRemoved], [not(at(_, _))]).$
$landmark(fightLocatedLmk, [checkedAreas, fightloc], [fight(_)]).$
$landmark(fightStoppedLmk, [not(fightloc)], [fight(X), not(at(_, X))]).$
$landmark(rescueOnStretcherLmk, [onStretcher(Ag), in('DropZone')],$
$[onStretcher(Ag), carryStretcher(Ag), at('DropZone')]).$
$concurrent_tasks(onStretcher(Ag), carryStretcher(Ag)).$
$before(injuredLocatedLmk, injuredRescuedLmk).$
$before(fightLocatedLmk, fightStoppedLmk).$

Figure 6.3: Selected landmark objectives defined for rescue scenario, shown as GOAL statements

ated with the landmark goal state, *injuredLocatedLmk*, which is comprised of achieving states: *checkedRooms* and *injuredLocated*. This is achieved by the agent adopting the goal *injuredLocatedGoal* and requires capabilities to move to be at a particular block: at(Ag,_) and recognise that the block is an injured: injured(Ag). *plan(injuredLocPlan, injuredLocatedLmk, [checkedRooms, injuredLocated], injuredLocatedGoal).* Within our test system the plan for locating injured involves checking all rooms for injured and at least locating one injured agent.

6.3.2 Define the organisation model

The next design questions are regarding the organisational structure. It is necessary to identify long term organisations agents can belong to and the agents associated with each organisation. Also, it is important to consider adhocracies that can form and anticipate these and incorporate these at design time. The OJAzzIC meta-model allows for the dynamic formation of adhocracies at run time if triggers are specified with policies at design time. However, in the current design stage, the focus is on defining long term organisations.

In our example, the initial design time organisations are clear. There are three organisations to begin with. These are the System organisation as a whole - including all agents with the shared high level objective to resolve the disaster. The Medic organisation includes all Medic agents. A third organisation is the Officer organisation. Based on the initial goal decomposition, high level objectives are allocated to particular organisations. For example, the overall goal objectives allocated to the Medic organisation are to rescue injured and transport injured to hospital using ambulances. The Officers organisation has responsibility to ensure the safety objective landmark is reached.

Within each organisation agent types can be identified. These types can have a

set of associated related roles they are capable of enacting. Do all agents have the same capabilities or are some more specialised? For each agent type identified, what capabilities does that agent type have? The answer to this design question may impact upon flexibility in the final system when adapting to changes in agent availability.

In our system, all Medic agents have the capabilities required to locate injured and enact a basic rescue of an individual injured agent. Medic agents also have the capability to remove blocking bystanders. We can also allocate to particular Medic agents the capability to perform a rescue on stretcher. Based on the capability set, the agents can be allocated to roles in the run time model or allocated (or adopt) responsibility for specific tasks.

When basic organisations have been identified, the designer needs to think about adhocracies that may form during a simulation. Adhocracies are organisations that may need to be created in order to facilitate coordination between agents across organisational boundaries such as when Officer agents and Medic agents work together to achieve an objective. For example: if an Officer improvises and helps out a Medic with a stretcher rescue, or if an Officer and two Medic agents need to work together to clear a safe exit for a complex rescue in an area of the rescue zone where access is limited due to a room collapse [8]. This complex rescue may require multiple coordinated activities and so an agreed plan for action and communication between these agents is required.

Forming an adhocracy organisational structure is beneficial to ensure that agents are coordinated with knowledge sharing and action plans. This is a feature of OJAzzIC, agents can dynamically form organisations and by doing so, agents can set up a boundary defining the set of agents who need to be coordinated and share information. At design time, if such an adhocracy can be anticipated, then triggers for the creation of an ad hoc organisation with members from both Medic and Officer agents based on the anticipated particular domain situation can be specified.

Adhocracies emerge dynamically during a scenario and can cross existing organisational boundaries. These organisations persist over some time to assist with coordination of particular objectives and to facilitate inter-organisational coordination. The motivation for the creation of an adhocracy includes the need for coordinated behaviour or knowledge sharing between individuals. During design, anticipate some of the situations when adhocracies that may form and the triggers for when they should be created. Social policies are used to define the triggers for creating adhocracies.

In our demonstration run-time system, three organisational structures are created

[8] room collapse is not implemented in the demonstration system.

at design time: medicOrg, officerOrg and combinedOrg[9]. For each organisation, a list of objectives, a list of member agents, a set of roles and plans are identified. The organisation's belief set is initialised to include the name of each org. The syntax *org(Org, Objlist, Memberlist, Rolelist, CurrPlanID, BeliefSet)* defines an organisation.

An example from our demonstration system to define the organisation: combinedOrg is as follows:

org(combinedOrg, [blockingBystanderRemovedLmk], [medic1, officer1], [medic,officer], [blockingBystanderRemovedPlan], [orgname(combinedOrg)])

The combinedOrg is a organisation created based on Medic agent medic1 and Officer agent officer1 both being in an organisation responsible for removing blocking bystanders. The presence of this organisation means that when these agents perform the task of removing blocking bystanders, they will be obligated by social policies within the organisational structure to keep each other informed about the progress of removing the bystanders. Without such obligations, the agents' knowledge would not be coordinated and so their behaviour with respect to blocking bystanders would be more random. For example, if medic1 and officer1 both noticed a group of blocking bystanders, and then medic1 cleared the group and did not tell officer1, then officer1 may have adopted the goal to also clear the bystanders before noticing that the task is no longer needing to be completed and so then dropping this goal. This may delay officer1 from performing other tasks.

6.3.3 Define the agent capabilities model

We have identified organisations and agent types in section 6.3.2. The agent type is a mechanism for categorisation of the particular types agents that might be instantiated in the system at run time. For each agent type, we list the capabilities possessed by agents of each particular agent type. The capabilities granted to each agent can be used to match agents to roles and tasks in the system. A set of capabilities can be associated with a role that is responsible for particular objectives. In our example, the agent types are based around existing roles (e.g. Medic, Officer), however, this need not be the case.

The agent capabilities model provides knowledge to enable organisation aware reasoning in terms of goal selection. In our example system, Medic agents have capabilities to locate injured, rescue injured and remove blocking bystanders. Officer agents have capabilities to find fights, clear fights and remove blocking bystanders.

[9]By definition, all agents belong to the sysOrg

In addition to the domain capabilities, organisation aware agents have reasoning abilities to consider the organisational objectives when choosing to adopt a goal. Social policies specify the priority process agents use for selecting goals. In the demonstration simulation system, an agent will first consider adopting an active landmark objective if it is an organisational objective and the agent is in a role that is responsible for that particular objective. Second, the agent will consider adopting an active landmark from within an active scene in which the agent is involved (no organisations involved). Third, the agent will consider adopting an objective if the agent is capable of fulfilling all tasks in an objective (apart from role allocations). Fourth, the agent will consider adopting a task that is part of a current objective if the agent is capable of achieving that task.

When an agent has a list of considered objectives, the agent will select a goal to adopt based on a prioritisation of these objectives. For example, the Officer agent will prioritise locating fights over stopping fights and lastly removing blocking bystanders. These priorities are decided at design time explicitly in the ordering of predicates in the program module or by defining priorities in meta-rules or policies.

6.3.4 Define the role model

The role model describes roles and responsibilities (objectives) associated with each role within the organisation. Each landmark objective has an associated capability set that defines one or more capabilities required in an agent to achieve that landmark. An agent can adopt or be allocated a role. Each role is defined with a roleID and an associated set of objectives that the role is responsible for. This means that an agent enacting that role will be responsible to adopt goals to reach those objectives. A role is defined with a roleID, and a list of landmarks that agents enacting that role are responsible for: *role(roleID, [list of landmark objectives])*

When specifying the role model, it is necessary to identify roles that agents may be able to adopt within each organisation. These may include domain roles such as Medic or Officer and structural roles e.g. Leader. For each role, identify the responsibilities that should be associated with that role in terms of which objectives that role is responsible for achieving. For example, the medic role is responsible for four landmark objectives: injuredLocatedLmk, injuredRescuedLmk, blockingBystanderRemovedLmk and rescueOnStretcherLmk.

Following the definition of roles, then role relationships can be identified (e.g. dependency, authority, right to delegate etc.). For example, Medic agents can delegate to Officer agents to clear blocking bystanders as there is a hierarchical dependency

between the Medic and the Officer, defined as follows:

dependency(medic, officer, [blockingBystanderRemovedLmk], hierarchical).

The OJAzzIC meta-model specifies that agents may, if appropriate use initiative, to dynamically adopt responsibility for tasks outside role-specific definitions where appropriate to achieve system goals. So, as part of the agents' decision-making process the agent should consider adopting the organisation's goals including those the agent is responsible for in their current role, however if the agent is not in a role, or there are no relevant organisation goals requiring attention, the agent should be capable of adopting a goal based on possessing the appropriate capabilities to reach the associated landmark.

Autonomy and initiative allowed in agents' organisational reasoning needs to be determined. For example, the task of clearing away bystanders can be fulfilled by any agent type within the vicinity using their initiative without being specifically asked/assigned a role. However, the task of rescuing an injured agent and moving them to the ambulance can only be adopted by a Medic agent type enacting the Medic role as delegated by the Medic in charge. Alternatively, perhaps there is no specific 'Medic in charge', but the Medic agents agree amongst themselves who is rescuing each injured agent, or the medics just go about selecting a rescue to perform without first agreeing who will do each rescue. These decisions can be guided by policies considered during design. Such questions around leadership roles or domain specific roles and responsibilities should be addressed as much as possible at design time. Then the system design can be configured with flexibility where it is anticipated that agents may need to dynamically revise objectives and agent allocations to roles or tasks.

We also require that agents improvise to form adhocracies at run time in order to facilitate an awareness and context for coordinated behaviour. These requirements are supported by the OJAzzIC organisation meta-model discussed in Chapter 5, representing agents as individuals with particular capabilities and relationships separate from role specific definitions. Agents can adopt or be assigned predefined roles, however roles can be split and agents can also be matched to potential tasks using individual capabilities. Policies can be used to specify situations where agents can adopt goals based on capabilities rather than roles.

6.3.5 Define the coordination model: establish social policies

At this stage, we establish social policies to be adopted within the organisational contract. As we are focused on social policies to facilitate coordination between cooperat-

ing agents, we are not concerned with defining sanctions to impose on non-compliant agents although clearly in a broader open application, defining such consequences may be essential to controlling an agent society.

During the process of developing policies, it is important to consider the run time adaptation requirements, including mutual adjustment of individual goals and awareness between agents. We aim that our agents can negotiate at run time to adjust their plans to fit in with others. Considering our case study as an example, we would want our design to enable medic agents working on the joint task of carrying a stretcher to dynamically adopt complementary roles as part of the stretcher carrying task (one medic as the block carrying medic and the other medic holding the door open). Also, in the case that a medic asks an officer agent who is not in the same organisation as the medic to engage with the stretcher carrying objective, then it must be possible for these 2 agents to both establish mutual agreement that they form an adhocracy so that they can coordinate their actions and knowledge relating to the rescue task.

As discussed in section 5.3.1, policies in OJAzzIC would address authority relationships, knowledge cultivation, goal selection, improvisation and the creation of adhocracies. Social policies can make explicit the priorities to aid agents in their reasoning, selection and adoption of goals. Priorities for goal adoption e.g. Medic agents low priority to remove blocking bystanders; priorities for role adoption e.g. Medic agents can be allocated the Medic Role; and priorities for communication - e.g. within an organisation, inform all others of task progress.

Meta-policies could be defined at design time and used to dynamically create policies at run-time. However, policies can be identified at design time based on a set of suggested policies, such as those broadly described in section 5.6.2. These can be combined with domain specific policies identified during analysis. The follow social coordination policies demonstrate organisation policies that could be implemented (not domain specific).

- An agent A can delegate a task to agent B in order to achieve an objective if the agent A is playing a role with authority to delegate to role that B is enacting.
- If an agent A completes a task which another agent B is dependent upon, then agent A should tell agent B the task is completed.
- If agent A and agent B share an objective and agent A completes the objective, then agent A should tell agent B it has been completed.
- If agent A and agent B are both involved in the same scene, then when an objective in that scene is completed, then the agent, A should inform other agents in the scene, B that it has been completed.

- If agent A and agent B are both members of an organisation O, then when an objective for that organisation is completed, then the agent, A should inform all other agents (B) in the organisation that the objective has been completed.
- If agent A and agent B are both members of an organisation O, and agent A adopts an intention to perform actions to reach a goal that is an objective of organisation O, then agent A will inform agent B of the intention.
- If agent A needs help in order to complete a task and agent B is available and capable, then agent A can request help from agent B regardless of whether agent B is in a role responsible for that task.

Domain specific social policies can be created at design time, to suit the particular model, based on these generic social policies. In Section 5.6.2 eleven demonstration policies were provided and then in section 5.10, a selection of six of these policies were chosen for the incident response scenario. Six specific domain examples are included in the formal description of a solution to the incident response scenario. With reference to the incident response system, domain specific illustrations of behaviour arising from the social policies corresponding to these six domain examples are described:

spolicyA, authority guidance regarding improvisation

Domain example: If an Officer is nearby with stretcher carrying capabilities and a Medic is needing to do a stretcher rescue, the Officer can help carry the stretcher on request by the Medic.

spolicyB, knowledge sharing obligations

Domain example: Medic will tell other Medics when an injured agent has been located and if a rescue has been completed;

spolicyC, obligations to share intentions

Domain example: As soon as a Medic has created an intention to perform a rescue, that Medic will inform other Medics of this intention;

spolicyD, role adoption responsibilities

Domain example: Medic will prioritise locating injured then rescuing injured, stretcher rescues are prioritised over individual rescues. When a Medic is asked to consider adopting a plan with another Medic for a stretcher rescue, then the Medic will prioritise this above individual goals;

spolicyE, obligations between agents to establish SharedPlans and inform relevant others if they cannot fulfil commitments

Domain example: Medics will perform stretcher rescues by first negotiating a SharedPlan with another capable stretcher carrying agent. If an agent becomes unavailable, that agent will inform the agent with whom the SharedPlan was created that they are now busy;

spolicyF, organisational adhocracy creation triggers

Domain example: If inter-agency coordination is required, (i.e. a Medic and an Officer are intending to work together) an adhocracy will be created to ensure appropriate communication and coordination occurs;

6.4 Conclusion

We have proposed a number of questions to be considered during the design stage of development of an organisation-oriented MAS. In particular, we suggested explicitly considering flexibility, coordination, adaptability, autonomy and adhocracies that could be created. The process of making explicit choices about elements of the system that can be specified at design time helps to clarify the requirements of the system overall. In particular, in trying to identify where flexibility and potential emergence can be anticipated and planned at design time, we can create a framework for run time instantiation of organisations. Each organisation defines a context for agents regarding knowledge sharing and coordination.

We have used the O-MaSE methodology adapted for the OJAzzIC meta-model. Similar to O-MaSE and OMACS we use goals to represent the system objectives. We differ from O-MaSE and OMACS in the way goals, roles, capabilities and agent classes are used at run time. In order to gain more flexibility at run time, we allow agents to adopt goals based on matching capabilities to sub-goals or tasks. Therefore during the design process it is important to consider the need for high level plans to support agents in creating SharedPlans and to articulate policies to guide agents with goal selection. We have also proposed the value in creating policies to guide agents regarding when they might be permitted to improvise for example, by adopting a goal outside of a role allocated or by handing over a task to another agent. It is important that during the design process, adaptability, flexibility and autonomy of agents is considered and where possible decisions made at design time to enable more flexibility and improvisation in agent behaviour at run time. The choice of policies is very important.

In Chapter 7, we demonstrate the application of the OJAzzIC design methodology to describe an implementation of a test MAS system based on this methodology. We use this implemented executable model to observe the impact of policies that could be adopted to support agents.

Chapter 7

Implementation of a MAS organisation based on OJAzzIC

7.1 Overview

In this chapter, the OJAzzIC meta-model introduced in Chapter 5 and the design process described in Chapter 6 are exercised by creating a demonstration MAS system for the scenario introduced earlier in the thesis. The behaviour of agents in the MAS[1] is observed in order to establish the benefits of adopting the meta-model as part of the design of a MAS. The operations needed for the scenario are specified in Chapter 5, section 5.10 and more formally in Appendix A.

This MAS simulation system is a 'proof of concept' model to demonstrate that adopting the concepts and features included in the OJAzzIC meta-model can facilitate flexibility at run time. The simulation is run with various scenario settings used to compare agent's run time behaviour with the behavioural requirements introduced earlier in the thesis and to test the policies suggested for coordination. By varying internal parameters such as policies adopted in the simulation system, and varying complexity in scenario settings, it is possible to demonstrate the impact these parameters and inputs have on the agents' behaviour. This is important so that we can make generalisations about the meta-model used to create the MAS.

In this chapter, an outline of the simulation approach used and the design of our scenarios is presented in sections 7.2 and 7.3. An overview of the design and implementation of the system, built based on the OJAzzIC meta-model and associated design methodology is then provided in section 7.4. Results follow in section 7.5. Some

[1]The implementation is available in a public Github repository: https://github.com/OJAzzIC/OJAzzIC

observations are discussed to highlight the distinguishing features in OJAzzIC with a discussion of the findings complete this chapter in section 7.6.

7.2 Demonstration MAS system and scenarios

In this section, we discuss the approach used for selection of scenarios and the implementation of the MAS.

7.2.1 Motivation for scenario testing

Following from the hypothesis proposed in section 1.6, we expect that adopting OJAzzIC as a meta-model accompanied by the design process in Chapter 6, answers the first part of the research question in section 1.3:

"How can we design organisations of software agents that coordinate the sharing of knowledge and plans, and demonstrate run time improvisation based on the design?".

Specifically, we seek to show the agents in the MAS, created based on the OJAzzIC meta-model, will then successfully achieve the following observable behaviour (repeated from section 1.6 with agent stories added). This list of observable agent behaviour aligns to the specific agent stories relating to the scenario that were introduced in Chapter 2.2.5, section 2.3.3 as indicated in parentheses:

i Agents will be able to improvise at run time to adopt goals if they have capability even if they are not assigned to relevant roles (agent stories : 1, 7)

ii Agents will be able to demonstrate mutual adjustment of individual plans to fit in with others (agent story : 2)

iii Agents will behave autonomously but prioritise organisational goals (agent stories : 3, 4, 6)

iv Agents will be able to coordinate joint activities at run time even if they are not already in an organisation with their collaborators (agent story : 5)

v Agents will be able to negotiate and create plans with other agents to coordinate joint action at run time (agent story : 5)

vi Agents will demonstrate appropriate knowledge sharing within the organisation in order to behave efficiently (agent story : 8)

We test this hypothesis with various runs using the simulation system. The scenario settings are described in the next section.

We examine the performance of the implemented MAS against the predicted behaviour. The outcomes observed from each of the scenario runs is discussed in sec-

tions 7.5 and 7.6.

We focus on the requirements of agent flexibility at run time and improvised coordinated behaviour. We test the capability of agents within organisations to improvise regarding role adoption at run time. In particular, we evaluate if the meta-model allows the appropriate level of specification of tasks, capabilities, roles and policies at design time to facilitate improvisation at run time. Of course, a solution is possible by 'hard-wiring' behaviour at design time with detailed synchronised plans to match possible run time contexts. We are interested in agents having some autonomy to choose the operational details at run time. We explore the use of design time specification of plans and policies to enable run time flexibility so that agents create commitments to adopt SharedPlans to coordinate behaviour at run-time.

Two particular situations are selected for the scenarios. The first situation tests role improvisation. This involves a scenario where an agent has the capability to complete a required objective, but is not directly allocated this objective based on role allocations. The agent can however improvise and adopt this objective when the need arises. The need may arise when another agent with the required capability is not available. We establish policies at design time to specify the tasks that may allow improvisation around how they are adopted and by which agents. The expectation is that at run time, agents will if required, improvise to adopt these tasks when faced with a situation where they have the capability but are not allocated to a role.

The second situation tests coordination between agents. It involves a joint rescue task that requires that two agents coordinate at run time to achieve one rescue task simultaneously. This situation is similar to agents combining to fulfil a role. The agents in this situation are expected to create run time commitments to each other to adopt complementary individual tasks simultaneously and then work together to achieve a joint rescue. We investigate the ability for agents implemented using the OJAzzIC meta-model to appropriately share information, coordinate and improvise. We measure agent success in completing rescues using a number of different scenarios with varying levels of complexity. Within our trials, we vary the internal MAS parameters: number and type of organisational policies provided at design time and measure the impact of this on success. We include policies providing varying levels of organisational support for agents, including a situation where agents have no organisational support at all.

The policies tested include obligations for sharing of beliefs, intentions and actions with other agents in an organisation. We also enable some flexibility with a policy to allow an agent involved in a collaborative joint task to handover its partner's task to another agent nearby.

7.2.2 Simulation system MAS implementation

The simulation system is based upon the problem scenario introduced in Section 1.2.1 and discussed in Section 2.3.2 as a blocks world implementation with BW4T. Consistent with the specification in section 5.10, the simulation system was built and a MAS designed based on the OJAzzIC meta-model. Agents in the system engage with the simulation and search for injured requiring rescue, then pick up and deliver the injured to the ambulance drop zone. In Section 2.3.1, the simulation environment is introduced. The simulation uses a blocks world environment in which agents control robots who are responsible for searching for and rescuing injured coloured blocks that are in rooms, picking them up and transporting them to a special drop zone.

In our rescue responder system, we deliberately specify roles at a high level and do not specify detailed plans for joint tasks. For example, two agents involved in a joint stretcher rescue, must at run time negotiate which particular role they will adopt when enacting a shared rescue together. One agent must adopt the role of support (waiting at the door and then 'carrying the stretcher') whilst the collaborating partner agent adopts the role of picking up and holding the injured block[2]. This run time behaviour requires that the agents mutually adjust to adopt the correct role to fit in with their partner agent. The stretcher rescue task involves two agents first agreeing that they have a mutual SharedPlan to adopt that rescue task, then both dynamically enacting (and perhaps adjusting) plans to ensure the rescue is completed. The task requires that the two agents communicate and collaborate to complete the joint task.

The process of deciding who will perform a rescue involves communication between the rescue agents. First one agent proposes to another that they might consider collaborating on a rescue. As each medic agent is behaving autonomously, there is no control over which agent may propose a rescue. An agent may instigate a rescue proposal or may receive a rescue proposal from another agent. When two agents agree that they both have the same potential rescue in mind, then they agree that this potential intention can become a mutual intention that this rescue be completed together. There is also a third possibility which enables an agent to directly invite another agent to accept and create a mutual intention-that regarding a rescue. When agents have an agreed mutual intention that they work together on a rescue, the then create a SharedPlan to enact a rescue.

[2]This fits in well with our aims because each agent must adopt a different role. It also satisfies the BW4T constraint that only one agent enters a room at a time.

7.3 Scenario design settings

In this section, we describe the scenario settings used during the evaluation of the MAS system.

7.3.1 Overview of different scenarios used

Our evaluation is conducted with reference to our requirements for our organisations of agents as discussed in Chapter 2.2.5, section 2.3.2, meta-model requirements developed in Chapter 4 and policies discussed in Chapter 5, section 5.10. The following variables were adjusted in different simulations:

1. varying number of agents, number of injured, state of injured and number of rooms. We used 3 main simulation settings:
   (a) 3 or 4 medic agents searching 4 rooms for 4 injured blocks requiring individual rescues
   (b) 3 or 4 medic agents searching 4 rooms for 4 injured blocks involving 2 individual rescues and 2 stretcher rescues
   (c) 5 medic agents searching 12 rooms for 10 injured blocks, 5 of which require stretcher rescues.

   This was varied in order to adjust the complexity of the tasks and the requirements for agents to coordinate. The simplest case involves individual rescues only and requires only coordination of agent beliefs. The cases involving stretcher rescues require agents to coordinate on joint tasks.
2. varying configurations with different levels of organisational support and organisation membership. We toggled on or off the organisational settings below to observe the impact (implications for each configuration shown in brackets after each setting):
   (a) full organisation support (medic organisation, officer organisation, sharing of beliefs and intentions within organisations, sharing of result of task completion within organisation, i.e. when a rescue is completed, implementing policies spolicyB, spolicyC, spolicyD)
   (b) no organisation defined at design time (no organisations, so no sharing of beliefs or intentions, no organisation policies to help agents, no adhocracies created)
   (c) full organisation support with addition of a combined organisation of selected medic agents and officer agents defined at design time as well as the medic and officer organisations (supports extra communication sharing be-

liefs between medic and officer organisations, enables collaboration between selected medic and officer agents, implementing spolicyA, spolicyB, spolicyC and spolicyD across organisations)

(d) partial policy implementation - an organisation policy specified at design time obligating agents to share beliefs within the organisation (explicit sharing of beliefs within organisation regarding situation assessment, implementing spolicyB regarding observed situational beliefs)

(e) partial policy implementation - an organisation policy specified at design time obligating agents to share rescue completion results within the organisation (explicit sharing of rescue status following a completed rescue with all other rescue agents so agents can act to drop existing plans if they are out of date, implementing spolicyB following updating of beliefs resulting from actions taken by agents)

(f) changing organisation membership - limiting the membership of the medic organisation defined at design time so that in some runs, a particular medic agent was not in the medic organisation (reduce the number of medics available for stretcher rescue collaboration and limiting knowledge transfer to exclude that agent)

(g) changing organisation structure - removing roles allocated to selected agents in the organisation at design time, so that some agents were forced to select goals based on capabilities not roles (tests the use case that agents can select goals based on capabilities)

3. varying levels of task complexity, focusing on coordination of joint tasks to increase complexity, see Figure 7.2. This was achieved by setting some of the injured to be seriously injured (orange blocks) requiring a stretcher rescue and some less serious (red blocks) can be rescued by one individual medic rescue agent.
4. varying flexibility given to agents by defining different improvisation policies (detailed in section 5.10) within an organisation

   (a) enabling an agent to handover (delegate) an agreed joint task to another agent, (medic agent can handover stretcher carrying task to another agent implementing spolicyA and spolicyD) or

   (b) find a new partner if original partner is unavailable (medic agent can ask another agent for help, rather than wait, if original stretcher carrying partner sends an unavailable message, implementing spolicyA, spolicyE and spolicyF)

The impact of adopting different social policies was tested by running the system with various different policies for sharing knowledge about the location of injured, rescue

intentions and task completion when a rescue is successfully achieved. The impact of adopting these social policies is discussed in sections 7.5 and 7.6.

In all scenarios, there were 3 officer agents in an officer organisation and there were 6-8 bystander agents each acting alone with no communication with others other than to receive messages from officer agents instructing them to move away.

In the first set of tests, we ran our simulation with either 3 or 4 medic agents using the 4 room map shown in Figure 2.2 multiple times with a number of different configurations of agent organisations (described in next section) and two different block configurations (baseline: 4 rooms with 4 red injured blocks or complex:4 rooms with 2 red injured blocks and 2 orange seriously injured blocks). The *baseline* map configuration for 4 rooms has 4 injured red blocks, 2 in room A1, 1 in room A2 and one in room B2. There are no seriously injured orange blocks. The *complex* map configuration for 4 rooms has 2 injured red blocks one in room A1 and one in room A2, along with 2 seriously injured orange blocks located in room B2 and room A1 respectively.

In the second stage of testing, we focused on more complex tasks using the 12 room configuration with 5 stretcher rescues, 5 individual rescues, 5 medic agents and 3 officer agents. The purpose of the second stage of testing was to test if there were situations where there was any benefit in agents having policies enabling them to perform handover tasks. We hypothesized that as the situation became more complex and agents had more competing potential joint tasks, there may be some increased efficiency in performance by granting agents flexibility to improvise.

7.3.2 Organisation configurations

Figure 7.1 provides a categorisation of the different settings used in various test runs. The focus in each simulation is the performance of the medic agents to complete rescues. Each configuration is categorised with an organisational structure index denoting of the level of organisational support afforded to agents. The index value increases as more organisational support is given to agents. Organisation support means that medic agents share beliefs, intentions and rescue outcomes with each other within the medic organisation. Organisation index 4 and above have full organisation support in the medic organisation. In Organisation index level 4 simulations, an extra combined organisation with membership of selected medics and officers was defined at design time. This had the effect that information was shared between the medic and officer organisations.

A higher organisation index value indicates that the agents were given more organ-

isational support with more policies to support the organisation. When a configuration is marginally better in support than another configuration, an index value increment of 0.5 is used. Otherwise, each configuration is given a unique integer between 1 and 9. Configurations with index values below 4 have had some changes made to the organisation membership or to remove policies for sharing of beliefs, intentions or rescue outcomes. Configurations with index values 4 or above have organisations for medics and officers and policies for sharing beliefs, rescue intentions and rescue completion. Configurations 5,6,7,8 and 9 have full organisation support with various policies for handover toggled on or off to observe the impact.

organisation structure index	number of rooms	medic org exists	combined org exists	sharing beliefs and rescue intentions	sharing rescue completion	stretcher handover
0	4	✗	✗	✗	✗	✗
0	12	✗	✗	✗	✗	✗
1	4	✓	✗	✗	✗	✗
1	12	✗ γ	✗	✗	✓	✗
1.5	4	✓	✓	✗	✗	✗
2	4	✓	✗	✗	✓	✗
3	4	✓ δ	✓	✓	✗	✗
3.5	4	✓ υ	✓	✓	✗	✗
4	4	✓	✓	✓	✓	✗
5	4	✓	✗	✓	✓	✓
6	12	✓	✓	✓	✓	✗
7	12	✓	✓	✓	✓	✓ α
8	12	✓	✓	✓	✓	✓ ω
9	12	✓	✓	✓	✓	✓ β

Legend	
γ	all medics are allocated to medic role even if there is no medic org
δ	one medic agent is not in medic org, but is capable of stretcher rescue
υ	one medic is not in the medic org, but is allocated a medic role
α	stretcher handover from medic to officer only occurs if officer is nearby
β	the stretcher handover only occurs in response to a busy message sent by the original partner
ω	the stretcher carrying delegation to an officer is automatic but will only occur in 3 rescues
✓	yes
✗	no
4 rooms	4 rooms, 0-2 stretcher rescues, 2-4 individual rescues, 3 or 4 medic agents
12 rooms	12 rooms, 5 stretcher rescues, 5 individual rescues and 5 medic agents

Figure 7.1: Organisation structure index

7.3.3 Task complexity during the rescue response scenario

The scenario adopted provides challenges for agents in terms of coordination of behaviour and information. Sharing of beliefs regarding the location of the injured blocks is helpful so that agents do not each have to do a complete search of the area. The stretcher rescue tasks require coordination between agents. Additionally, if medic agents share information regarding rescues as they are completed, then other medic agents can update their individual beliefs and intentions so that any out dated intentions can be dropped.

At the beginning of a run, the officer agents will find bystander agents and instruct them to move away to a different location. Following a situation assessment search phase, looking for injured at the beginning of the scenario, the medic agents then consider enacting rescue operations. The medic agents are capable of performing two types of rescue:

1. A simple rescue (of a red block) involves the agent entering the room, moving to the block, picking up the injured (block) and taking it to the dropzone and putting it down - representing delivery of the injured to the ambulance.
2. A stretcher rescue (of an orange block) requires two medic agents working together, one takes on the support role and waits at the door to the room whilst the other enters the room and picks up the orange block, using a virtual stretcher, both agents then together move to the drop zone with the carrying agent holding the block. The carrying agent enters the dropzone to put the injured block down, whilst the support agent waits at the front of the dropzone.

Task	Complexity	coordination required	autonomy
search for injured	low	share beliefs	high
individual rescue	low	share intention, share beliefs	high
clear bystanders	low	share beliefs	high
stretcher rescue	high	agree on shared intentions, adopt roles, coordinate joint task	low
stretcher carry handover	low	handover of role	low

Figure 7.2: Tasks and their complexity

We categorise the key simulation scenario tasks and associated complexity, coordination needs and individual task autonomy in Figure 7.2. The complexity is low when the task does not involve interdependence with other tasks, agents or resources. The complexity is high when there is some interdependence requiring agents to coordinate

their behaviour with others. The autonomy is high when an agent can act independently with full autonomy and no need to consider others, low otherwise. In all cases, agents have full autonomy to select which particular rescue goals to adopt.

Following the initial search of rooms, the medic agents need to first establish which rescue they will adopt - which block is to be rescued and with which agent to collaborate on this task. When agents have agreement that they will work together on a joint task (a mutual intention-that they will see to it that the task will be completed), they create an exclusive contract referred to as a SharedPlan. If the 2 agents are not part of an existing organisation, they also create an instance of an adhocracy to which they are both added as members. This then ensures that they follow organisation policies to share related information appropriately with each other. The stretcher rescue is an organisation goal that requires coordination and communication between agents and is representative of more urgent medical rescues. Our agents prioritise the organisation's goals ahead of individual goals[3], so in our responder system, the agent prioritises the stretcher rescues (orange blocks) above individual rescues (red blocks). If an agent is aware that other agents are intending to or in the process of completing all existing stretcher rescues, then that medic agent will move on to consider individual rescues.

The plan for enacting a stretcher rescue requires that at run time, the 2 collaborating agents need to mutually agree to adopt a SharedPlan, then individually adopt different roles - i.e. support and pick up roles. SharedPlans are discussed in Chapter 4, section 4.4 and in Chapter 4.2 and relate to mutual agreement between agents regarding the adoption of an intention to achieve a joint task and how they will enact a joint task. In OJAzzIC, within an organisation's contract, obligations regarding the sharing of beliefs, intentions and task completions are specified in social policies. When a SharedPlan is adopted, the two agents involved need to agree on (at least) a partial recipe for how they will reach their agreed intention-that. This involves agents adopting a plan, establishing roles and then enacting this plan. As each agent deliberates based on individual beliefs about the world, so that agents can coordinate effectively, efficient communication between agents regarding relevant beliefs is required. In our rescue scenario, agents need to appropriately communicate to share information about the problem space such as location of blocks that require rescuing and when a block has been successfully rescued. Agents communicate regarding their coordination to achieve the interdependent tasks involved in a joint stretcher rescue between 2 agents: who is doing this rescue, which roles are being adopted by whom. Agents also need to communicate regarding the completion of tasks so that other agents can adjust outdated

[3]Prioritising organisational goals does not remove agent autonomy in terms of selecting which stretcher rescue to complete

beliefs remove intentions to perform an already completed task.

Complexity is present because multiple medic agents need to coordinate which 2 agents will take responsibility for each joint rescue task. There is a possibility that 2 different agents could propose 2 different rescues with a 3rd agent simultaneously, so, to avoid a deadlock situation, whilst allowing agents full autonomy, it is necessary for agents to each consider and propose possible rescue situations as potential intentions and then have a mechanism for agents to select and agree on a proposed rescue to become an actual intention. This is consistent with the SharedPlans concepts of potential intentions and intentions. Another complex situation arises following 2 agents having agreed on the need to create a SharedPlan, they then need to negotiate or allocate which role each will adopt. In other words, they need to elaborate and agree on the more specific details of the plan. One approach to role coordination would involve hard-coding the plans for stretcher carrying roles so that each agent would automatically assume a particular role, however this lacks the flexibility that we are trying to achieve, so we have deliberately required the agents to negotiate and adopt roles dynamically. This provides an opportunity for agents to demonstrate mutual adjustment.

The stretcher rescue requires coordination between two medic agents as follows:

1. agents create a number of potential intentions regarding the rescue(s) of a particular block with other agent(s)
2. an agent proposes a potential rescue to another medic agent (or one agent directly requests assistance with a rescue)
3. two agents agree to collaborate on a potential rescue of a particular block (each individual agent agrees and creates individual intentions-that they will perform the rescue together)
4. two agree to adopt and enact a particular rescue now (create a SharedPlan : mutual beliefs of a partial recipe to achieve the rescue of a particular block. Both agents create intentions-to perform the rescue)
5. agents begin to adopt particular roles - one support agent and one pick up agent, each agents adopts an individual role
6. agents communicate and take action to enact a stretcher rescue by adopting individual goals to reach this. If necessary, agents mutually adjust their individual actions to fit in with each other

Following the creation of the SharedPlan commitment, agents engage in a process of agreement on the individual intentions-to adopted by each agent. In some cases, this requires a process of mutual adjustment. Mutual adjustment is needed when two agents clash intentions by both simultaneously adopting the intention to enter the room to

pick up the injured block. Agents communicate to inform each other of their intentions and so, when the conflict is realised, one agent then needs to adjust their individual plan to then wait at the door and adopt the role of the stretcher carrying agent.

There is also complexity based on the number of rooms to be searched and the number of medic agents. The baseline configuration used in our testing involves 4 rooms, 3 or 4 medical agents, all in the medic org with either 2 stretcher rescues and 2 basic rescues or 4 basic rescues. The complexity in the adoption of rescues by agents in the baseline scenario with 2 stretcher rescues and 3 medic rescue agents (medic1, medic2 and medic3) indicates that the solution space of potential stretcher solutions is 3x2 as 2 agents are required for a stretcher rescue and there are 3 potential combinations of agents for each stretcher rescue (medic1 + medic2, or medic2 + medic3, or medic1 + medic3).

In a more complex scenario with 5 stretcher rescues and 5 medic agents, there are 5x2x5 unique solution combinations possible. There is no hierarchy of medic agents and no central coordination, so the medic agents need to dynamically determine who is going to perform each rescue. Once a rescue plan is adopted, then other medic agents can be informed (if a suitable social policy exists to oblige agents to share this information) and so will no longer consider adopting the same rescue, however until a SharedPlan is adopted and others are notified of this, all medic agents may consider this as a potential rescue to be performed.

To ensure that matches would be found between agent proposals and to avoid making this deliberation too intensive, especially as the number of agents increases, agents do not consider all possible rescues before adopting one to act upon. An agent could directly suggest a proposed intention to a potential partner and request that they directly adopt this intention to achieve a rescue and then following commitment, proceed to create a SharedPlan to achieve this together.

An agent will not propose a potential rescue if the agent has been informed that other agents have intentions to do this rescue. During the medic agent's deliberation process, if an agent is informed that other agents have adopted intentions to complete a rescue, the agent will drop any potential intentions or intentions to complete that same rescue.

This design decision to prioritise stretcher rescues ahead of individual rescues could easily be changed, however we felt that it was a good indicator of success and agent awareness that agents need to believe that all the organisation's goals requiring coordination are complete before proceeding to adopt the individual rescue tasks.

7.3.4 Policies

In Section 6.3.5, policies for the demonstration scenario were described based on the policies in the specification in section 5.10. The policies implemented in the MAS are provided in Figure 7.3.

spolicyA	an officer agent will be obligated to commit to a medic agent if requested to help with a task that the officer is capable of achieving.
spolicyB	any agent in an organisation is obligated to share new beliefs with other agents in that org.
spolicyC	when an agent is in an organisation and creates an commitment to achieve a particular objective in that organisation, then that agent is obligated to inform others in the organisation of their intention.
spolicyD	obligates an agent, agent1 in an organisation to commit to another agent in the organisation who requests help with a particular objective where agent1 is enacting a role responsible for the objective.
spolicyE	states that when an agent has an existing commitment in an organisation and is busy, then that agent is obligated to inform other agents in that organisation that it is currently busy.
spolicyF	is a policy to create a new adhocracy, a new organisation if there are 2 agents with a shared goal who are not already in an existing organisation.

Figure 7.3: Social Policies implemented in demonstration MAS system

7.3.5 Metrics

In order to measure performance under the various circumstances and allow comparison between simulation runs, we used the following performance metrics:

1. Success : did the agents find and rescue all injured? (yes/no)
2. Awareness : were all agents aware that the rescues were completed? (yes/no)
3. Time : time taken to complete simulation.
4. Efficiency :
   (a) number of rooms entered by each agent; and
   (b) idle time for each medic agent.
5. Communication : number of messages sent between agents regarding
   (a) situation awareness - location and status of injured;
   (b) situation awareness - rescue completion; and
   (c) goal adoption (stretcher rescues).

A clear performance measure is time. However, the number of rooms entered is important as this is an indicator of the combined total effort involved in the solution

amongst all agents involved. In the complex room trials, the focus was on differentiating between different social policies for if/when to have a medic handover a stretcher carrying task to an officer. The handover activity requires more communication to drop the SharedPlan with the original medic partner and delegate the stretcher carrying task to the new officer partner. In order to compare the behaviour and performance of agents in the complex 12 rooms scenarios, the idle time of agents was also examined to get a measure of how efficient each agent's activity was during the run.

7.3.6 Scenarios

Figure 7.4 summarises the scenario runs that were executed. Scenarios 1-3 are baseline experiments with low complexity and no need for agents to coordinate their plans or actions. Scenarios 4-17 compare agent behaviour in conducting rescues in the 4 room scenario with 2 stretcher rescues and 2 simple rescues. These scenarios demonstrate the agents ability and performance differences with various different levels of organisational support provided at design time (membership and roles) and policies for knowledge sharing of beliefs, intentions and rescue results. Scenarios 18-22 are based on a more complex situation with 12 rooms and 5 stretcher rescues. The objective of these experiments was to explore and differentiate agent performance with different policies for handover of tasks. The organisation structure index in Figure 7.4 relates to those defined in Figure 7.1.

The scenarios attempted with index values less than 4 were used to observe agent performance without the potential organisational benefits involving sharing of information between agents. When medic agents share their beliefs regarding where the injured are located, this saves all agents needing to individually conduct a search of the rooms. When medic agents also shared their rescue intentions with others in the organisation, then if multiple pairs had adopted the intention to perform the same rescue, the agents would react when informed that another agent pair has intentions that the rescue be completed to change their intentions to adopt a different rescue. If medic agents are not informed by others about rescues that have been completed, or the rescue intentions of others, then it is possible that a medic agent will have an outdated intention to perform a rescue until the agent notices that the rescue has been completed.

In the simulations conducted with the more complex 12 room configuration, full organisation support with sharing of beliefs, intentions and rescue completions was provided. Additionally, a combined organisation including selected medic and officer agents was defined at design time.

Three different policies for handover of the stretcher carrying task were used to

compare performance. The policies allowed a medic agent, under some circumstances, to decide to delegate a stretcher carrying role to a nearby capable agent rather than wait for the medic agent who had originally planned to perform this role.

Scenario	Organisation structure index	Complexity	No. medic agents	No. rooms	No. stretcher rescues	No. simple rescues	No. runs
4 room rescues, 4 red blocks 3 or 4 agents							
1	1	low	3	4	0	4	4
2	4	low	3	4	0	4	4
3	4	low	4	4	0	4	4
4 rooms, 2 red 2 orange blocks, 3 medic agents, stretcher rescues, different organisation support							
4	0	high	3	4	2	2	4
4	0	high	4	4	2	2	4
5	1.5	high	3	4	2	2	3
6	2	high	3	4	2	2	3
7	3.5	high	3	4	2	2	4
8	4	high	3	4	2	2	4
9	5	high	3	4	2	2	5
4 rooms, 2 red 2 orange blocks, 4 medic agents, stretcher rescues, different organisation support							
10	0	high	4	4	2	2	11[a]
11	1.5	high	4	4	2	2	4
12	2	high	4	4	2	2	3
13	3	high	4	4	2	2	4
14	3	high	4	4	2	2	4
15	3.5	high	4	4	2	2	4
16	4	high	4	4	2	2	4
17	5	high	4	4	2	2	6
12 rooms, 5 medic agents, stretcher rescues, some handover, different policies							
18	0	high	5	12	5	5	5
19	6	high	5	12	5	5	5
20	7	high	5	12	5	5	5
21	8	high	5	12	5	5	4
22	9	high	5	12	5	5	5

[a] 6 runs were unsuccessful

Figure 7.4: Scenario configurations

Agent Configurations varied in terms of membership and the amount of organisational support afforded the agents. The number of agents varied, as did the membership of the organisation: medicOrg. In some configurations, one medic agent was taken out of the medicOrg. In that case, fewer agents were able to collaborate on the shared rescue task. The excluded medic agent was still able to act autonomously to perform an

individual rescue task. In one configuration, one of the medics in the medicOrg was set up to always handover a stretcher rescue task to an officer following the establishment of a SharedPlan. The magnitude of the configuration category index increases with more organisational support. Configuration with index value of zero has no organisational support. Configuration with index value of four has full organisational support with sharing of beliefs, intentions and completion of tasks with others in the organisation.

In the higher organisational categories (6 and above), tests were conducted using the complex room configuration with 12 rooms, 5 medic agents and 5 stretcher rescues. The potential number of agent interactions and collaborations to complete a stretcher rescue is greater. Additionally, there are more stretcher rescues to be completed, and more rooms to search. The objective of these tests was to distinguish behaviour and outcomes regarding stretcher handover policies.

In configurations with a policy for handover, one medic in the medicOrg may handover a stretcher rescue task to an officer following the establishment of a SharedPlan or following a busy message received from their partner.

When a handover of the stretcher carrying task occurs, a new adhocracy is created dynamically including the carrying officer and the original medic agent involved in the stretcher rescue. The presence of this adhocracy will oblige the officer and medic agent to share information about completion of the rescue with each other without the need to explicitly create rules in the plan about this communication. If the medic agent is also in an existing (separate) organisation with other medics, then this same information will then be passed on to the other medics in the medicOrg. In all simulations with organisational support, officerOrg includes all officer agents and the combinedOrg includes medic1 and officer1. When we were testing agent behaviour without organisational support, some of these organisations were eliminated.

The behaviour of agents during each simulation run were saved to log files for inspection. The baseline configuration with 4 rooms involves 3 or 4 medical agents, all in the medic org with either 2 stretcher rescues and 2 basic rescues or 4 basic rescues. The simulation was run under four altered configurations of agent organisation to observe any impact. The alterations were:

- removing one of the medical agents from the agent organisation, but keeping the removed medic agent allocated to capabilities of the role ‘Medic’;
- allowing one of the medic agents to delegate an existing stretcher carrying task to an officer agent;
- turning off the policy to share beliefs (location and status of injured) within an organisation;

- turning off the policy to share beliefs within an organisation but adding a specific rule to inform others in the organisation when a rescue has been completed.
- turning on an extra capability for officers that they can carry a stretcher and enabling one medic to handover and delegate an adopted stretcher rescue tasks mid way to an officer.

The objective of running the simulation under various configurations was to verify and demonstrate the usefulness of implementing the multi-agent system using the OJAzzIC meta-model and to see the impact of adjusting different social policies. It was our hypothesis that agents within an organisation would perform better due to the social policies within an organisation leading to better sharing of information within the organisation. As shown in the specification of the organisations in Figure 7.8, it is possible to easily change the members allocated to an organisation. So it is possible to run the simulation in different configurations for instance with the *medicOrg* only containing 2 medics, medic1 and medic2 and thus medic3 would be an individual medic with all the capabilities of a medic, but excluded from the organisation instance *medicOrg*. It is also possible to remove individual role allocations (see Figure 7.11).

7.4 Rescue response system design

In this section, a design for the simulation system is presented.

7.4.1 Overview of implementation platform for agents

The agents in our rescue response system were implemented using the GOAL programming language (K. V. Hindriks 2009)[4]. GOAL is a high level language, based on the BDI paradigm that enables agents to make rational choices about the goals they will adopt based upon their mental attitudes such as beliefs and knowledge about the world. The GOAL language provides an interface to interact with BW4T and enables the agents to communicate with each other by sending and receiving messages. GOAL agents follow a blind commitment strategy so they commit to adopted goals until they are achieved or until new beliefs change the agent's decision to adopt the goal. This may occur if the conditions leading to the adoption of the goal have changed, for example, if the agent perceives changes in the environment or if another agent informs this agent that the goal has already been reached. Agents may have a number of possible action choices at any time based upon their current beliefs and goals. The goal list for

[4]The selection of GOAL to implement this system was practical rather than based on a preference for GOAL over other BDI agent languages.

an agent specifies what the agent wants to achieve at a point in time and is similar to a desire in the BDI framework (A. S. Rao and Georgeff 1991). When an agent selects a particular goal to adopt, then that goal describes a state that the agent intends to achieve. Agents choose actions from provided rule-based action rules based on their current goals. In our simulation system, If a medic agent updates a belief (for example, no longer believes that there is a particular block needing rescuing in a room) and if that agent had an existing goal adopted to perform that rescue, then the updated belief will lead the agent to drop the existing rescue goal. Additionally, if an agent is informed by another agent of an intention to perform a particular rescue, then the agent will drop an intention to perform this rescue, this has the effect of loosening the blind commitment strategy to behave more like an 'open-minded commitment' (A. S. Rao and Georgeff 1995). Ideally, rational agents should be able to reassess existing goal commitments when a conflicting higher priority goal exists (Thangarajah, Padgham, and Harland 2002). In our simulation, this is not implemented. When an agent has begun a stretcher rescue for example, unless the rescue partner agent sends an explicit message to drop the shared plan, or beliefs change to make this goal no longer valid, the agent will blindly commit to completing this rescue. The MAS system was designed using the OJAzzIC meta-model with organisations for medic agents and officer agents modelled on the solution specified in section 5.10. Policies outlined in the Z specification were implemented, with the ability to toggle these policies on and off to see the effect on agent performance.

Based on our requirements the reasoning cycle of our organisation aware agents is outlined below The code corresponding to steps 2a and 2c is shown as GOAL code in Figure 7.5.

1. Consider individual objectives (based on existing individual goals)
2. Consider organisation objectives: adopt an active landmark state as an objective if one of the following holds:
   (a) It is an org objective and I am responsible due to my role;
   (b) I am responsible (no org);
   (c) I am capable (not in an allocated role);
   (d) I am capable of achieving part of the objective.
   (e) I have been asked to help and I have the capability required
3. adopt goals based on landmark objectives
4. choose actions based on current goals
5. send/receive messages to and from other agents
6. perceive updates to environment
7. respond to any events to update beliefs and current goals

```
% consider adopting an objective based on role
if    % agent is enacting role in an organisation, and in scene,
        bel(agent(Agent,_),
        rea(Agent, Role, Scene),
        scene(Scene, Roles, Objectives),
        %role is responsible for an active objective in this org
        responsible_org(Org, Obj, Scene, Role),
        %organisation is responsible for this objective
        org(Org, OrgObjList, OrgAgentList,
              OrgRoleList, OrgPlanList, OrgBelList),
        % agent belongs to organisation responsible for this objective
        member(Agent, OrgAgentList),
        %within the currect set of Objectives, each Objective
        member(Objective, Objectives),
        % there is a plan including this landmark objective
        plan(Plan, Objective, _, _),
        % in set of plans for this organisation
        member(Plan, OrgPlanList),
        %Role is responsible for this Objective
        responsible_landmark(Objective, Scene, Role))
        % landmark is active
        active_landmark(Objective),

then
        % agent can consider adopting the objective task Objective
        adopt(consider(Agent, Objective, Scene)) +
                insert(matchedOrgrea(Objective, Org)).
```

```
% consider adopting an objective based on capabilities
if      % agent has a list of individual capabilities
        bel(agent(Agent, CapabilityList),
        % current scene has a set of Objectives
        scene(Scene,_, Objectives),
        % within the current set of Objectives, each Obj
        member(Obj, Objectives),
        % landmark state Obj has a task list defined
        landmark(Obj, _, TaskList),
        % there is a plan including this landmark
        plan(Plan,Obj,_,_),
        % the landmark is currently an active landmark goal
        active_landmark(Obj),
        % agent is capable of all tasks in this objective
        capable_all(Agent, Obj))

then
        % agent can consider adopting the objective task Obj
        adopt(consider(Agent, Obj, Scene)) +
                insert(matchedcap(Obj)).
```

Figure 7.5: Partial GOAL code for organisational goal consideration by an agent

The code provided in Figure 7.5 is GOAL code taken from the code module orgReasoning(agent) that is run by an agent in the main module when no other actions match. This code is run initially for an agent to identify some organisation high level goals to consider. Then after those goals are completed or dropped, it will be called again. The purpose of this module is to set up organisational beliefs - including what organisation they belong to, define roles etc. and then to adopt some goals to consider.

The order in which goals to consider are adopted is based on considering org goals first then if there are no org goals, consider based on role, then lastly based on capability. The agent will always perform the action generated by the first rule possible, so as soon as a goal is adopted, this module is completed and the agent cycle continues, so by placing the highest priority rule first, agents will always select the highest priority goals for consideration.

The reasoning cycle for agents in the implemented MAS is as follows (K. V. Hindriks 2014):

1. The event module (containing event or percept action rules) is run to update the agent's mental state (effectively this module handles events and messages); After the perceptual events are processed, the last step in the event module is to call the organizationalEvents module.
2. The organizationalEvents module is responsible for taking action that is immediate, sending organisational messages based on policies, updating beliefs and goals based on messages received, revising goals that are outdated and negotiating with other agent regarding SharedPlans.
3. The main module is entered and rules in that module are applied to define action selection, so the agent selects an action using the action rules (based on existing beliefs and goals). The action rules are evaluated in linear order (top - bottom), the first valid rule is selected and applied. By placing the rules in priority order in the main module, the agent will choose the highest priority valid selection (matching preconditions on the rule). If there are no existing individual goals, the agent will consider adopting organisational goals (identified in the organizationalEvents module). If there are no organisational goals for consideration yet, the agent will then call the orgReasoning module.

This reasoning cycle is based on the inbuilt reasoning within GOAL, with the addition of the organizationEvents module. This is consistent with the BDI-interpreter cycle as proposed by Rao and Georgeff (A. S. Rao and Georgeff 1995). According to Rao and Georgeff, the BDI-interpreter cycle is as shown in Figure 7.6. The agents in our simulation follow a slightly modified version of this cycle. The execute step is achieved

in the main GOAL program. The first 3 steps in the repeat loop (lines 4-6) are achieved in the event module in the simulation system code. There are also some prioritised actions and decisions to drop outdated goals embedded in between deliberation in the event module.

```
1  intialize-state();
2  repeat
3    % done in the event module in the simulation system
4    options := option-generator(event-queue);
5    selected-options := deliberate(options);
6    update-intentions(selected-options);
7    % done in the main module
8    execute();
9    % done in the event module
10   get-new-external-events();
11   drop-successful-attitudes(); %ie drop goals already achieved
12   drop-impossible-attitudes(); %ie drop goals if cannot be achieved
```

Figure 7.6: BDI interpreter cycle (A. S. Rao and Georgeff 1995)

When an agent is capable of achieving part of an objective, or agents outside a formal organisation structure agree to work together to achieve an objective, this requires coordination. Agents can form SharedPlans to agree on how they will work together, but if they also need to ensure that appropriate information is shared between the relevant agents, then the agents can create an adhocracy. An example of a situation requiring an adhocracy is when a medic agent hands over responsibility to an officer agent to carry a stretcher.

A medic agent following creation of a SharedPlan to perform a coordinated stretcher rescue could decide to handover the responsibility for the stretcher carrying task to an officer agent. This requires little negotiation, it involves a delegation and handover message so that the SharedPlan agreement is transferred to the new *officer* partner. The handover requires a sharing of beliefs and plans relating to the rescue between the *medic* and the *officer*. If the handover was completed without also creating a new adhocracy, then issues would arise in terms of communication regarding the completion of the objective. The agents completing the rescue may not know to inform the original medic that the objective has been reached successfully. This was addressed by creating an adhocracy involving the officer and medic agents involved in the rescue to ensure appropriate sharing of beliefs. Without this adhocracy, the alternative would be to specify the stretcher rescue plan with more detail to include explicit rules to specify that rescue agents will inform others (and explicitly state which others) when the rescue has successfully completed.

In the remainder of this section we outline the high level design for our rescue

response system aligned to each of the steps proposed in the design process in Chapter 6. We also provide some commentary regarding the implemented system.

7.4.2 Functional Goal model

The goal tree showing the decomposition of objectives in the rescue response system was shown in Figure 5.10. The logic for organisation aware agents in terms of considering potential objectives is partially shown in Figure 7.5. When an agent adopts an intention to reach a particular objective, that agent is intending to achieve a particular goal state.

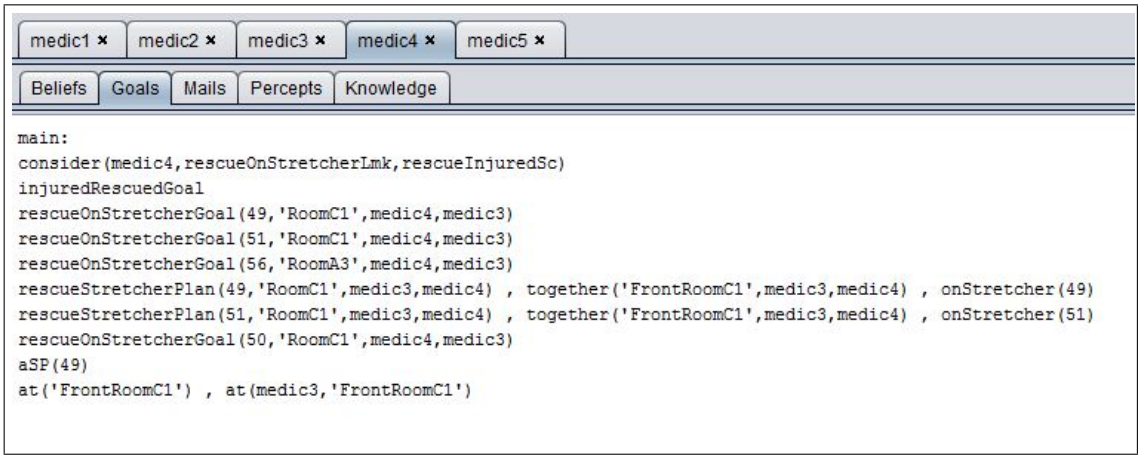


Figure 7.7: A snapshot of goals for medic agent medic4 at a particular time during a run

A snapshot of the goals for medic agent medic4 adopted at a particular time during a simulation run is illustrated in Figure 7.7. Medic4 has adopted the organisation's objective rescueOnStretcherLmk and has adopted the injuredRescuedGoal, followed by a number of potential rescue intentions. The rescueStretcherPlans indicate that the agent has identified two potential intention-that plans (a plan to rescue block 49 with medic3 and a plan to rescue block 51 with medic4) and agreed with the potential partner, medic4 that each of these are potential intentions. The goal list indicates that medic4 has then created a SharedPlan aSP(49) to rescue block 49 with medic3. The existence of this goal aSP(49) indicates that medic3 and medic4 have agreed and committed to a SharedPlan to complete this rescue. As part of the high level plan for this stretcher rescue, medic4 has then adopted the (sub) goal that medic3 and medic4 will both be at the front of roomC1. At this point, the agents haven't yet decided or agreed upon which agent will be the main rescue agent and pick up the block and which agent will be the stretcher carrying agent. However, as the SharedPlan is created, each agent knows that the commitment is shared and they will both begin to enact the first step of the plan and move to meet at the front door of room C1. Later, the agents will each adopt an explicit rescueOnStrectherGoal for this rescue and then as part of

that goal will negotiate picking up the injured, carrying the stretcher and moving to the dropzone.

7.4.3 Organisation model

Three organisations were created at design time: *medicOrg, officerOrg* and *combinedOrg*. For each organisation, a list of objectives, a list of member agents, a set of roles and plans are identified.

Each organisation's belief set is initialised to set the name of each org as shown in Figure 7.8. The syntax *org(Org, Objlist, Memberlist, Rolelist, CurrPlanID, BeliefSet)* defines an organisation. The design time specification of each org is shown in Figure 7.8.

Organisation specification

```
org( Orgname, Objlist, Memberlist, Rolelist, CurrPlanID, BeliefSet )
```

```
1  org(medicOrg, [injuredLocatedLmk, injuredRescuedLmk],
2          [medic1,medic2,medic3], [medic],
3          [injuredLocPlan,injuredRescuePlan], [orgname(medicOrg)])

4  org(officerOrg, [blockingBystanderRemovedLmk, fightLocatedLmk,
5          fightStoppedLmk],        [officer1,officer2,officer3], [officer],
6          [blockingBystanderRemovedPlan,fightLocatedPlan,fightStoppedPlan],
7          [orgname(OfficerOrg)])

8  org(combinedOrg, [blockingBystanderRemovedLmk], [medic1, officer1],
9          [medic,officer], [blockingBystanderRemovedPlan],
10         [orgname(combinedOrg)])
```

Figure 7.8: Specification of OJAzzIC organisations for the rescue response system

When a handover of the stretcher carrying task occurs, a new adhocracy (dynamic run time organisation) is created dynamically including the carrying officer and the original medic agent involved in the stretcher rescue. This is an implementation of SPolicyF described in section 7.2.2.

The GOAL code shown in Figure 7.9 stipulates that if there is an officer nearby in the scene, then that officer can be delegated to carry a stretcher by a medic and a new adhocracy (rescueOrg) is created between the medic agent, M and the officer agent O.

The adhocracy is created by creating a new belief about the org rescueOrg using the insert command on line 15. This code first delegates the stretcher carrying role to an officer agent close by. In addition to creating the new organisation adhocracy rescueOrg, it also involves dropping an existing shared plan created with another agent (OM) (line 12) and replacing this with a new shared plan created with the officer agent

(O) (line 13). The rescue goal associated with this new shared plan is adopted on line 10. This code demonstrates direct programming of the policy to create an adhocracy when a medic delegates stretcher carrying to an officer nearby.

```
1  if bel(agent(O), closeby(O,FL), avail(O),
2          rea(O, officer, rescueInjuredSc), me(M),
3          front_room(L,FL)), a-goal(torescue(Ag,L,OM,M)),
4     not(bel(carryStretcher(Ag,_,FL,Loc)))
5
6  then send(O, !delegated(carryStretcher(Ag,M, FL, Destn))) +
7         adopt(together('FrontDropZone',O,M)) +
8         adopt(torescue(Ag,L,O,M)) +
9         drop(at('DropZone')) + drop(torescue(Ag,L,OM,M)) +
10        drop(holding(Ag),in('DropZone')) +
11        insert(delegateCarryStretcher(Ag,O)) +
12        adopt(rescueOnStretcherGoal(Ag,L,O,M)) +
13        drop(rescueOnStretcherGoal(Ag,L,OM, M)) +
14        send(OM, drop(aSP(Ag,L,OM,M))) + delete(aSP(Ag,L,OM,M)) +
15        insert(aSP(Ag,L,O,M)) + insert(carryStretcher(Ag,O,FL,Destn)) +
16        insert(closeby(O,FL)) +
17        insert(org(rescueOrg, [rescueOnStretcherLmk],
18                             [M,O], [medic, officer], [injuredRescuedPlan],
19                             [myorg(rescueOrg)])).
```

Figure 7.9: GOAL Code showing creation of an adhocracy with an officer delegated to carry stretcher

Lines 1-4 in Figure 7.9 are explained as : if the agent (a medic agent, M) believes that: agent O is close by the front of room L; agent O is available; agent O is enacting the role of an officer and agent M has already a goal to complete a rescue with agent OM (another medic agent) which hasn't yet reached the carrying stage then the handover will occur (lines 6-19). Line 7 shows the medic agent M adopting new goals to complete the rescue with the officer O. Lines 9-10 show Medic M dropping existing goals that related to the original rescue plan. Line 12 shows Medic M adopting the rescueOnStretcherGoal particular to the new rescue with officer O, then line 14 shows the deletion of the SharedPlan with agent OM and on line 15 creation of a belief of a SharedPlan aSP with agent O. As the rescue was delegated, there is no need to negotiate all these steps and wait for agreement from officer O.

Using organisations enables agents to dynamically identify the other agents with an interest in knowledge, belief sharing and task collaboration. Line 15 of Figure 7.9 shows that the agent M inserting into its beliefs that the organisation rescueOrg has been created. The officer agent O would insert a similar belief into its individual beliefs. The organisation rescueOrg is responsible for the landmark objective rescueOnStretcherLmk and has two member agents: M and O. The beliefset associated with the organisation on creation is that the agent believes that myorg(rescueOrg) is true. Each organisation

has a set of objectives. The injuredRescuedLmk in the medic organisation indicates that this organisation, medicOrg is responsible for achieving this landmark state by following a high level plan: injuredRescuedPlan.

7.4.4 Agent capabilities model

In addition to the role capabilities inherited with assigned role responsibilities, each individual agent can be assigned a number of individual capabilities.

In Figure 7.10, medic3 is given individual capabilities to perform a stretcher rescue and to clear blocking bystanders. The capability to perform a strecher rescue is achieved by explicitly stating that medic3 is capable of the tasks `onStretcher, carryStretcher` and `at('Dropzone')`. Giving additional capabilities to medic3 enabled further adaptability in the system, so if medic3 is not specified to enact the medic role, medic3 could still consider adopting rescue tasks based on capability rather than role. In such a case, if medic3 is not in the medic organisation but is to be considered as a potential partner in a shared rescue, then there also needs to be a policy in place for other medic agents to consider potential partners outside the medic organisation, based on their capability rather than only considering partners who are in the medic role within the organisation. Otherwise, medic3 would only be able to perform individual rescues.

Agent Capability Specification

```
agent( agentName, [capability list] )
```

```
1  agent(medic1, [at(_), at(_, _), holds(_), injured(_)])
2  agent(medic2, [at(_), at(_, _), holds(_), injured(_)])
3  agent(medic3, [at(_), at(_, _), holds(_), injured(_),
4                 blockingBystanderRemoved,
5                 not(at(\_,\_))
6                 onStretcher(Ag),
7                 carryStretcher(Ag, O, L1, L2),
8                 at('DropZone')])
9  %
```

```
9  agent(officer1, [not(at(_,_)), fight(_)])
10 agent(officer2, [not(at(_,_)), fight(_)])
11 agent(officer3, [not(at(_,_)), fight(_)])
```

Figure 7.10: Agent capability specifications in the rescue response system

7.4.5 Role model

In our rescue response system, agents have capabilities to fulfil the role responsibilities for roles that they enact. Medic agents enacting the medic role have role based capabil-

ities to locate injured, rescue injured and perform stretcher rescues. Officer agents have role based capabilities to locate and clear fights and remove blocking bystanders. In addition to the domain capabilities, all organisation agents have the ability to deliberate considering the organisational objectives when choosing to adopt a goal.

The role specification shows the default capabilities attributed to each role. the belief rea(AgId,RId,SId) specifies that a particular agent, AgId is allocated as a 'role enacting agent' (rea) to the role RId in the Scene SId. In our system, a role is defined with a roleID and a set of landmarks as shown in the GOAL rules in Figure 7.11. To illustrate this, the rule on lines 1 and 2 in Figure 7.11 specifies that the medic role is responsible for (and therefore capable of achieving) the named objectives: injuredLocatedLmk, injuredRescuedLmk and rescueOnStretcherLmk. The rule on line 5 specifies that the agent *medic1* is enacting the *medic* Role in the *rescueInjuredSc* Scene.

Role Specification

```
role( RoleID, [Objectives] )
```

```
1 role(medic, [injuredLocatedLmk, injuredRescuedLmk,
2                              rescueOnStretcherLmk])
```

```
3 role(officer, [fightLocatedLmk, fightStoppedLmk,
4                              blockingBystanderRemovedLmk] )
```

```
rea (Agent, Role, Scene )
```

```
5 rea(medic1, medic, rescueInjuredSc)
6 rea(medic2, medic, rescueInjuredSc)
7 rea(medic3, medic, rescueInjuredSc)
```

```
8 rea(officer1, officer, stopFightSc)
9 rea(officer1, officer, rescueInjuredSc)
```

Figure 7.11: Role specifications in the rescue response system

7.4.6 Social policies

The policies chosen for adoption in the MAS implementation are listed in section 6.3.5. In this section, the policy, SPolicyB is chosen for demonstration purposes. This policy specifies that any agent in an organisation is obligated to share new beliefs with other agents in that org. New beliefs can be regarding location of injured and also status of completed rescues.

The example GOAL code in Figure 7.12 demonstrates an implementation of the policy, SPolicyB that an agent will share beliefs with others. This code is sharing

regarding the location of injured blocks with all other agents who are enacting a role with responsibility for the landmark: `injuredRescuedLmk` (line 2) or the landmark: `injuredLocatedLmk` (line 6). In GOAL, an a-goal represents an *achievement goal*. This is a goal that has not yet been reached.

The code shown in Figure 7.12 defines two rules that for all adopted goals for the agent (Me) involving considering a landmark objective (injuredRescuedLmk or injuredLocatedLmk), and for all the Agents (Ag) who are in a role R that is responsible for the respective landmark, then the agent will share its current beliefs with those agents. The beliefs shared are the locations where injured have been found (Line 12) and what rooms have been checked (line 14).

```
1  forall a-goal(consider(Me,injuredRescuedLmk,Sc)), bel(me(Me),
2          rea(Ag,R,Sc), responsible(injuredRescuedLmk,Sc,R), Ag\=Me)
3             do sharebeliefs(Ag).
4
5  forall a-goal(consider(Me,injuredLocatedLmk,Sc)), bel(me(Me),
6         rea(Ag,R,Sc), responsible(injuredLocatedLmk,Sc,R), Ag\=Me)
7            do sharebeliefs(Ag).
8
9  module sharebeliefs(Ag) {
10 program {
11          forall bel(injuredLocatedAt(A,Locn))
12                    do send(Ag,injuredLocatedAt(A,Locn)).
13          forall bel(checked(Room)), not(bel(sent(Ag,checked(Room))))
14                do send(Ag, checked(Room)).
15          }
16 }
```

Figure 7.12: GOAL code implementing policy for sharing beliefs of location of injured

The GOAL code in Figure 7.13 demonstrates the policy that when an agent believes an organisation objective has been reached in a Scene Sc, it will share this belief with all other agents, REA who are enacting roles in the scene (rea(REA,_,Sc)). Similar policies are implemented to share selectively e.g. with others in an organisation or in the same role as the agent Ag.

```
1  forall a-goal(consider(Ag,Objective,Sc)), bel(reached(Objective)),
2     bel(me(Me), rea(REA,_,Sc), REA\=Me)
3          do send(REA, :reached(Objective, Sc))
```

Figure 7.13: GOAL code implementing policy to share objective reached beliefs

7.4.7 Policies addressing our flexibility requirements: initial observations

In addition to the creation and adoption of SharedPlans between agents, appropriate knowledge sharing, guided by social policies helps the agents perform tasks efficiently. For instance, if agents have a policy to share information about the location of the injured blocks, then it is not necessary for all medic agents to enter every room in order to gather situational awareness of the rescue scene and the location of injured blocks. Agents can share their beliefs with each other, so that during the initial investigation phase each room only needs to be entered by one medic agent. This knowledge sharing requires that agents know that it is important to share beliefs relating to injured blocks and also know which other agents are interested in this knowledge. There is a trade off between communication and the time it takes for agents to send and receive messages and the time available for action. The attention of agents to focus on communication detracts from the agents' performing actions. It was our expectation that sharing beliefs about location of the injured would reduce the search actions and number of rooms entered and would improve efficiency and time taken to complete a rescue. We did not have any particular expectations regarding the impact of organisational support and the flexibility of enabling the handover of stretcher carrying tasks on time or efficiency in the solution.

Across all scenarios, situations were created to test the ability of our agents to

1. use initiative in adopting a task based on capability not role responsibilities,
2. to share and create mutual knowledge of situation and plans and
3. to mutually adjust individual plans to fit in with others.

We used situations where improvisation was possible based on defining policies to adopt tasks based on capabilities possessed rather than role adopted. Additionally, policies were implemented for situations when agents at run time decide to work together outside of an organisation (such as the officer agent carrying a stretcher with a medic). In these situations, agents create an adhocracy to ensure appropriate coordination.

The improvisation addressed is in terms of finding a set of tasks and associated goals that will achieve a particular goal state. Agents are not restricted but are guided in their selection of individual goals. It is necessary for agents to choose the order in which rescue goals can be adopted. This does not limit agents to only one possible particular goal at a time (e.g. there are multiple rescues possible). The GOAL implementation will prioritise types of goals selected based on the order in which they are found in the main module, but each individual agent may randomly choose any particular rescue of

that type to perform.

Improvisation is demonstrated with policies to specify when agents might choose to change a plan, for example a medic agent could decide to find an alternate partner for a stretcher rescue if a message is received to say the original agreed partner is busy. In the following discussion, we highlight how the design assists agents to achieve coordination and flexibility in their behaviour.

Using initiative - adopting objectives based on capability This requirement for improvisation is supported by the design of OJAzzIC in enabling agents to adopt goals based on matching capabilities. This was tested and demonstrated in our system. For example, medic3 was given the additional individual capabilities:

```
[blockingBystanderRemoved], [not(at(\_,\_))]
```

These capabilities are needed to reach the objective blockingBystanderRemovedLmk, to clear bystanders. Possessing these capabilities enabled medic3 to consider the potential goal of removing a bystander agent. This would enable medic3 when finding a bystander agent during a search to remove this bystander agent directly rather than delegating this to an officer agent. Figure 7.14 shows that medic3's goal objectives considered include blockingBystanderRemovedLmk as well as rescueOnStretcherLmk.

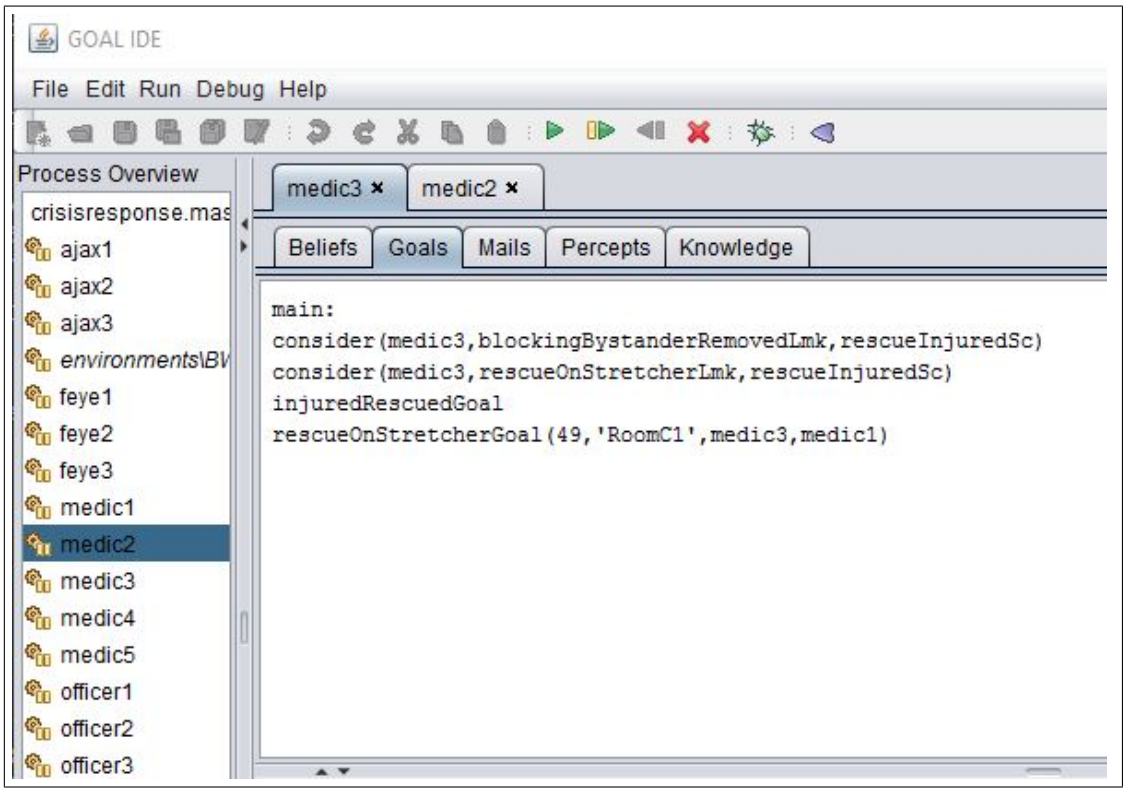


Figure 7.14: Snapshot of goals adopted by medic3 during a run

In order to run the simulation without any medic organisation, policy rules were added (shown in Figure 7.15) to enable medics to invite other agents with the appropriate stretcher carrying *capability* to engage in the collaborative rescue stretcher task for a seriously injured block when there was not a matching agent within an organisation to collaborate with. The `rescue_injured_together` module defines rules so that an agent can consider negotiating and creating a shared rescue plan with any of the potential agents in the Members list.

```
1  forall a-goal(injuredRescuedGoal),
2     bel(injuredLocatedAt(X,L), %believe injured X, located at L
3           seriouslyinjured(X),  %X is seriously injured
4           me(M)    % I am agent me (M)
5           rea(M,R,Sc), % M is enacting role R
6           % Role R is responsible for rescue landmark
7           responsible(injuredRescuedLmk,Sc,R),
8           %Members lists all other agents capable of rescue
9           agents_capableList(Members, rescueOnStretcherLmk)   )
10
11      do
12            insert(inRoleIwillWorkWithOthersCapable) +
13           %insert belief of who others are and potential rescues X,L
14           insert(rescue_injured_together(X,L,Members)) +
15           % call module: consider potential rescues with any of Members
16             rescue_injured_together(X,L,Members).
```

Figure 7.15: Policy to find rescue stretcher partner based on agent capability not role, from Medic agent

Share information It is expected due to sharing information because of policies in the organisation that organisation aware agents would be enabled to be more focused in their behaviour. It is possible to turn on/off whether an agent is in an organisation. When this was done to remove medic3 from the *medicOrg*, medic3 was not told by other agents of located injured blocks, so had to individually search and enter every room to establish complete awareness of the situation. Also, medic3 was not informed when other agents completed a rescue, so was behaving with outdated beliefs. Medic agents by default, only consider collaboration on the shared stretcher rescue with other agents in the medic organisation, so medic3 was unable to contribute to a shared rescue. The result was that medic3 went about searching and rescuing the red blocks whilst the other medical agents in the medicOrg were completing the stretcher rescues. Then, all agents had beliefs inconsistent with reality in terms of the rescues that had been completed. Medic3 was not informed that the orange blocks were rescued and Medic3 did not inform others when it had successfully rescued red blocks. So, as each agent adopted out dated goals and began to enact a rescue of a block that no longer needed rescuing, then as the agent went looking in the room and discovered no block

there needing rescuing, the agent then updated its beliefs and dropped the associated rescue goal. The simulation still completed successfully, but agents did more work to individually each establish the current situation and update goals.

If we turn off the policy to share beliefs regarding the injured amongst medic agents in the *medicOrg* organisation, the initial situation assessment phase requires that agents individually check all rooms. If an agent is not part of the organisation, that agent is not informed about where injured blocks are found by others. The search goal then requires more activity for each agent. If we turn off the policy for medic agents to share intentions regarding adopted SharedPlans for rescues, then there will be agents who are not aware of others' intentions. So, if there are more than 3 agents involved in the rescue organisation, potentially, multiple available agent pairs could then compete to perform any particular rescue.

In the following discussion, we focus on the objective: rescue injured, to highlight the knowledge sharing benefits gained by our organisation aware agents. The organisational instance defines which other agents to share with. Unsurprisingly, the organisation aware agents share information that enables them to be more coordinated in their behaviour than an agent without an organisation. In the unaware system, with no coordinated knowledge sharing, each individual medic agent, when allocated the rescue task has to first individually search the potential locations and identify where the injured agents are, before planning a rescue. However, in the organisation aware agent system, as soon as any medic agent locates an injured agent, this knowledge is shared with all other medic agents by sending a message to each. This allows the organisation aware medic agents to focus on the rescue task sooner.

When an injured agent is picked up ready for delivery to the ambulance dropzone, the environment changes and all agents are able to, upon entering a room, perceive that change, notice that the injured block is no longer there and update their beliefs. Organisation aware Medic agents, due to their social policy for sharing beliefs send a message to all other medic agents when the rescue of a particular injured agent has been completed. In this case, with access to beliefs about rescued agents, the organisation aware agent can choose to adopt goals to rescue injured agents only if they have not already been rescued. This avoids the unnecessary creation of redundant goals. Figure 7.16 shows a sample of messages sent and received by medic agent, medic3 during a run of the system. medic3 shares relevant beliefs with medic1 and medic2 because they are in an organisation: medicOrg. Medic1 and medic2 in turn update their own beliefs when informed by medic3. Social policies describe these knowledge sharing obligations.

In the implemented MAS, and in the OJAzzIC meta-model, there is no consideration of agents mistrusting another agent. As soon as a message is received with new information, such as a rescue being completed, the agent completely trusts this and updates its individual beliefs. This is a limitation of both the demonstration system and the meta-model.

medic3:messages

```
sent(medic1,rescued(23))
sent(medic2,rescued(23))
received(medic2,rescued(23))
received(medic1,rescued(23))
sent(medic1,rescued(22))
sent(medic2,rescued(22))
received(medic2,rescued(22))
```

medic3:beliefs

```
clearingRoom('DropZone')
occupiedclearRoom('DropZone')
rescued(23)
injuredRescued(23)
informed(rescued(23))
rescued(22)
informed(rescued(22))
```

Figure 7.16: Selection of messages and beliefs from medic3

Mutual adjustment of plans to fit in with others It is not possible to pre-plan in all cases. Coordination based on mutual adjustment requires agents to change their plans to coordinate with other agents dynamically so as not to interfere with other agents by their own behaviour. Meso-level policies ensure prioritisation of agent goals to fit in with the organisation's goals.

Mutual adjustment is demonstrated when 2 stretcher rescue medic agents happen to simultaneously both propose to be in the role of agent entering the room. This is seen in the communication sample between medic3 and medic2 shown in Figure 7.17. This sample is taken from an actual run of the simulation. Medic 3 and medic2 both simultaneously send messages to say 'I will go into the room'. It is not possible for both agents to enter the room. Initially, we attempted to have a negotiation phase where agents agreed on roles, but this was difficult to script in the situation where both agents send this message simultaneously. It is of course possible when one agent is proposing this before the other agent. In that case the second agent can agree and then respond to say they will wait at the door. The solution to the conflict caused when 2 agents simultaneously propose 'I will go into the room' was to enable the agents to use their initiative and have one agent decide to swap roles and wait at the door. This was a demonstration of mutual adjustment. Each agent upon receiving a message from the other agent that conflicted with their own suggested intention, then observed and adjusted their intentions (the agent in the room keeps the intention to go into the room and the agent at the door adjusts its intention to wait at the door).

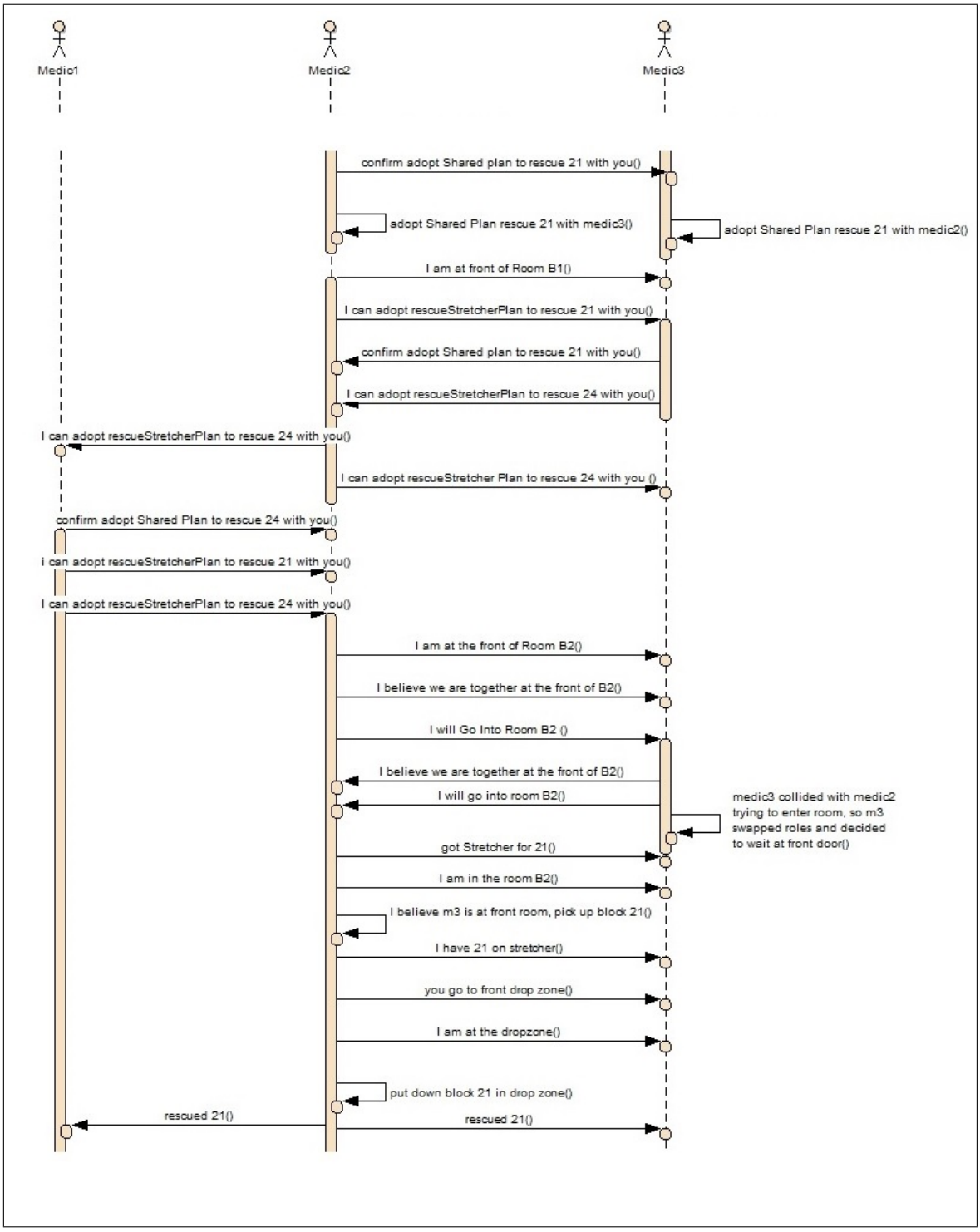


Figure 7.17: UML sequence diagram (OMG n.d.) showing agent interactions demonstrating mutual adjustment of plans

Staying open and flexible up until SharedPlans adopted Regarding the proposal and adoption of SharedPlans, we initially had agents communicating with each other when they had proposed a rescue with another agent in the org. This was done so this could impact on subsequent proposals so that agents wouldn't double up with proposal of a rescue that another agent had proposed. This design rule was initially

conceived so agents would not have competing proposals. However, it was found that all agents needed to be open to adopting a potential rescue up until a mutual SharedPlan was actually adopted. This is in keeping with the SharedPlans process of establishing mutual potential intentions. This avoided a tie-break deadlock situation where each one of the three agents had proposed a different rescue (e.g. Agent A proposed rescue with agent C, agent C proposed rescue with agent B and agent B proposed rescue with agent A). This demonstrated that agents needed to stay flexible with potential goals until details were agreed upon.

Allowing short cut adoption of a SharedPlan if invited prior to proposal Following on from the previous example, in some cases, if agents have not yet created a proposal to match with that of other agents, the cycle of propose, agree and adopt took multiple cycles of deliberation. In order to enable agents to accept a proposed SharedPlan when there is no other competing plan proposed, a rule was added to circumvent the initial proposal steps. This rule was added to help agents directly invite another agent to adopt a mutual intention-that and SharedPlan. This reduced the number of deliberation cycles for stretcher rescue plans in these cases.

Handover of tasks by delegation, creating an adhocracy The idea of having a medic agent handover or delegate the support stretcher carrying role to an officer is not of itself novel and can be achieved within the agent plan for a stretcher rescue. There are two consequences of interest in our simulation related to the handover. One is the need to create a run time organisation if one does not exist already so that agents can collaborate and coordinate without the need to insert rules into the agent plans regarding communication. The second is that by including in the design that the stretcher carrying role can be completed by any agent with the capability to do so, flexibility can be gained at run time so that agents can use initiative in order to complete the rescues in a timely way.

The simulation design was modified so that a medic agent who had established a SharedPlan with another medic agent to perform a rescue on a stretcher, could change its mind and delegate the stretcher carrying task to an officer agent. Using a policy rather than embedding this in the plan gave agents the ability to choose to do a handover at run time depending on the situation.

The handover of a stretcher carrying role required little complexity of coordination as it was a simple delegation and handover of responsibility to transfer the plan to the officer. However, in order for the officer to coordinate information and actions with the medic agent, they form an adhocracy. This behaviour at run time to enable two agents

to create an adhocracy and share the stretcher carrying role is a demonstration of the use of the organisation construct to assist agents. The deliberation of agents to create SharedPlans and adopt knowledge sharing policies is facilitated by the organisation. These same features could be used to facilitate agents deciding to split an existing role described at design time. The same actions would be required of agents: create an adhocracy, agree on roles, create a SharedPlan and (because of the existence of the adhocracy) adopt policies for knowledge sharing.

We also tested a similar handover policy when a medic agent received a busy message from their potential partner upon suggestion of adopting a shared intention-that to adopt a SharedPlan on a particular rescue. An extension of this, would be to test agent initiative where the system failed. For example, if an agent was no longer available to carry a stretcher and did not explicitly handover the task, it would be necessary for the collaborating agent to trigger/notice that the task was failing and then begin the process of finding a new partner with which to coordinate to complete the task. One approach that would work to address this situation would be to specify a policy that the departing agent would need to signal their departure/unavailability to complete the plan by sending a message to their partner. This would then trigger deleting the SharedPlan and starting again.

7.5 Results

The results presented in this section serve to demonstrate that by adopting the OJAzzIC meta-model at design time, run time adaptability and flexibility in the agent behaviour within a MAS can be achieved. This behaviour includes improvisation and coordination between agents. It is not the purpose of this section to suggest or prove that this particular implementation is optimal. Rather it is hoped that this is a proof of concept that demonstrates that the concepts and features included in the OJAzzIC meta-model facilitate flexibility at run time. Further work would be needed to claim performance benchmarking against alternative meta-models. In this section, we present results to highlight where we believe that adopting the OJAzzIC meta-model has been useful.

In Section 7.3.5, a number of metrics were introduced that will be used to compare the behaviour of agents in each of the simulations.

Figure 7.18 provides a summary of all the 4 room simulation runs involving co-ordination of a stretcher rescue. Green results are for simulation runs involving 3 medic agents, blue results represent simulation runs involving 4 medic agents. These simulations were used to demonstrate that the agents could perform as intended and

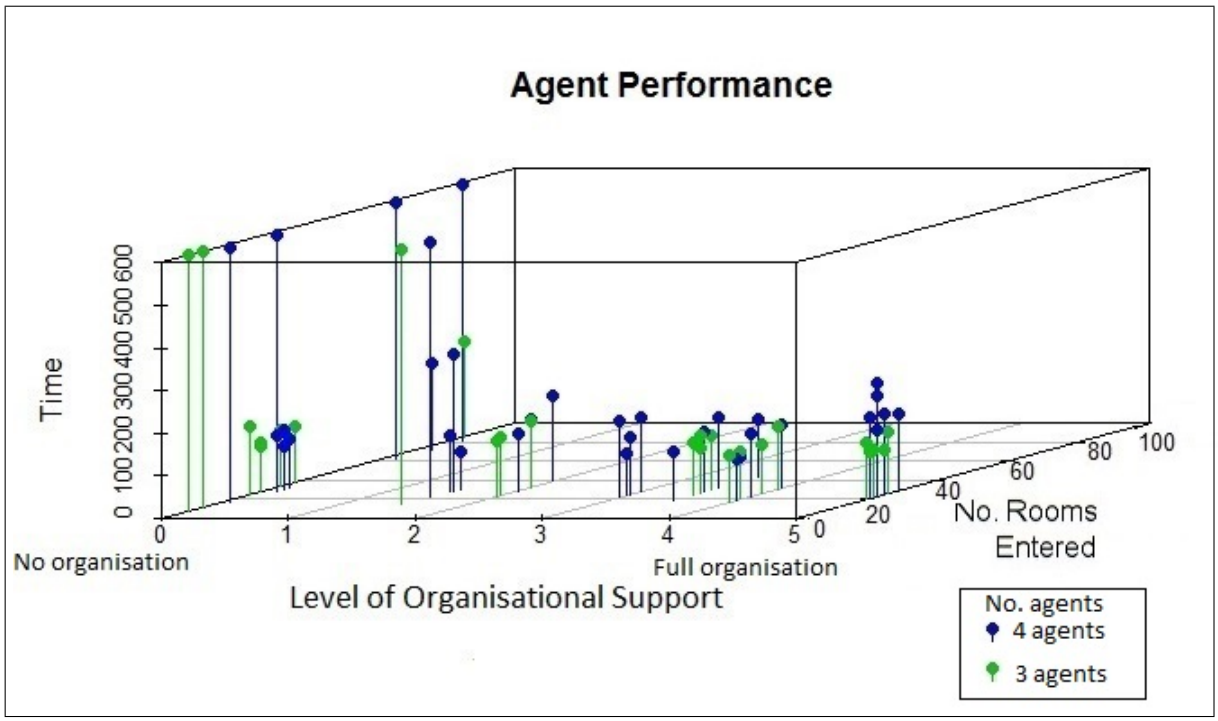


Figure 7.18: Agent performance

coordinate behaviour without detailed scripts created at design time. The Y axis shows time taken for each run and the Z axis shows the total number of rooms entered by the agents. To identify which runs have best performance efficiency, we need to focus on data with minimum time and minimum total number of rooms entered. This data is clustered primarily in the right hand front corner of the graph. The X axis separates the runs into categories representing how much organisational awareness and support was provided in the configuration. These categories are the index values listed in Figure 7.1. The index value is representative of the level of organisational support and complexity in each simulation. A value of zero represents a configuration with no organisation for the medics, a value of 4 represents full organisational support with all medic agents belonging to the medicOrg organisation with policies turned on for sharing beliefs and sharing completion status when rescues are complete. The value of 5 represents the category where the medic agents are in an organisation and where one of the medic agents when performing a stretcher rescue will handover the stretcher carrying task by delegation to an officer agent. A higher index number on the X axis indicates more organisation centred properties were afforded to medic agents in the simulation. For ease of comparison, we refer to this as the level of organisation support provided to the agents. Index values in Figure 7.1 above 5 relate to simulations in 12 rooms. The purpose of the simulations in 12 rooms was to explore agent behaviour with different handover policies and will be discussed later in this section.

The data in Figure 7.18 indicates that the agents with no organisation awareness

(zero on the level of organisational support X-axis) were not as successful as those with organisation support. The successful runs occur with category level 2 and upward. These simulations involve organisations that are communicating and performance is similar across these simulations regarding time taken. There are generally small time improvements as more beliefs are shared within the organisation. The highest level of organisation support in this group of simulations is level 4 and 5. Category 5 simulations have better performance overall on the number of rooms entered during the simulation.

In many of the runs with zero organisational support, the time is at the max of 600. When a run was not successful the completion time was arbitrarily set to 600 to indicate this. Further, in some runs at level zero, the agents completed all the rescues, but without awareness shared between agents. In these cases, some of the medic rescue agents were unaware of the fact that the injured had been successfully rescued. Some of the runs were not successful at completing the required rescues. This information regarding completeness and awareness is not shown in Figure 7.18, however will be highlighted in subsequent figures.

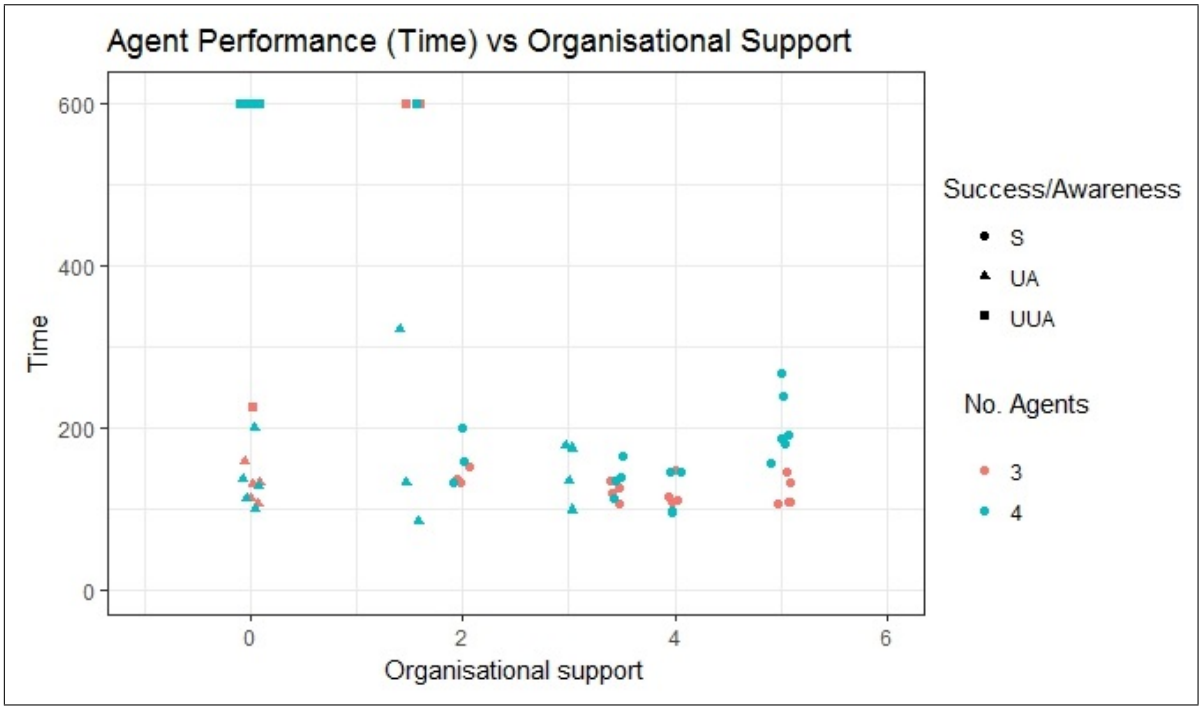


Figure 7.19: Agent performance over time

Figure 7.19 highlights the performance of agents against time for all the 4 room simulation runs involving stretcher rescues and hence requiring coordination between agents. The total number of rooms entered by agents during the simulation runs are show in Figure 7.20. In Figure 7.19 and Figure 7.20, individual data points corresponding to each index value are randomly positioned horizontally around each appropriate index value in order to distinguish each data point separately. Data points shown as

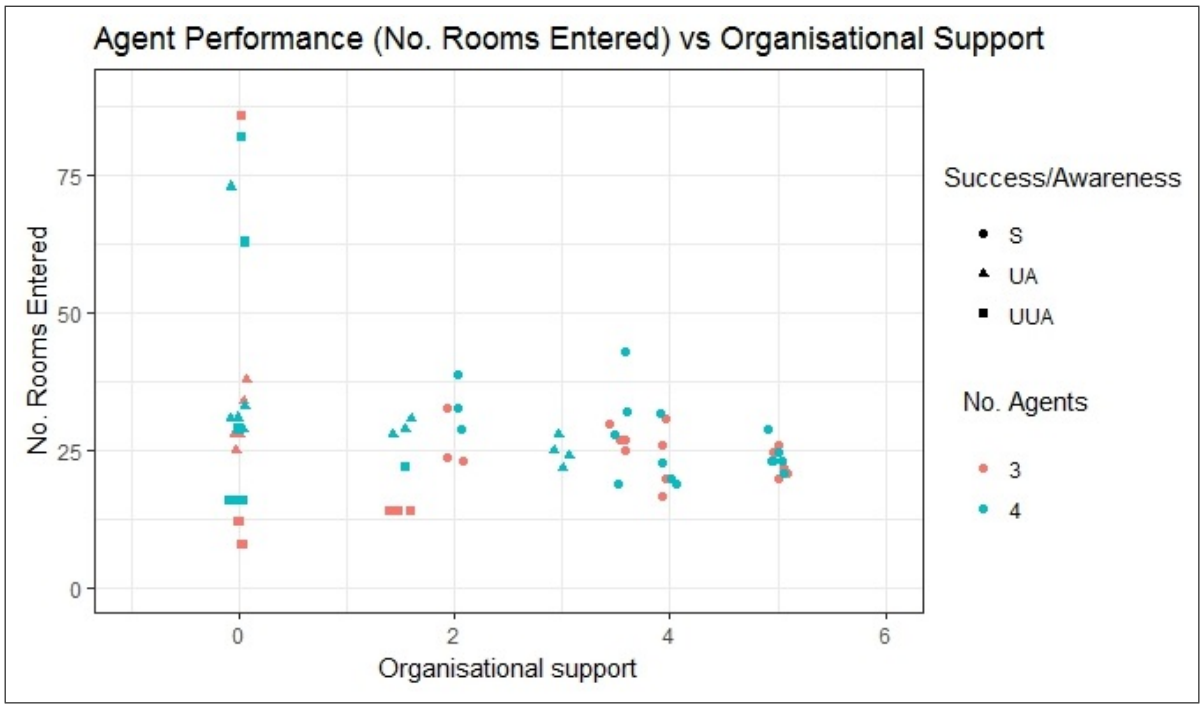


Figure 7.20: Agent performance regarding number of rooms entered

circles in the graph represent successful runs of the system when agents completed the search and rescue of all the injured and all agents were aware of this. The data points associated with runs when at least one agent was unaware of the true completion status of the rescues are indicated with 'UA' symbolised as a triangle in the graph. The square data point associated with the annotation 'UUA' indicates that the run was unsuccessful and also at least one agent was unaware of the situation in terms of rescue completions.

If the meta-model support of using an organisation with policies to support knowledge cultivation is beneficial, it would be expected that as the organisational index value increases, the performance would improve. There are some simulation runs in the 4 room stretcher rescues with low times with low organisational support index values, however, these runs are associated with situations where agents were unaware of the accurate completion status of rescues. This unawareness is due to the absence of policies to share beliefs amongst the agents who are not in an organisation.

There are some runs with good performance times clustered around the organisation structure index value of 2 indicating that they did complete, but had entered more rooms than the most efficient runs. In these cases, the organisational policy was set such that agents did inform each other about completion, but did not share beliefs around the location of injured. This required each agent to individually conduct a search of the area and so more rooms were entered during the initial search phase. Simulation runs at level 3 shared beliefs about location of injured and rescue intentions but did

not share beliefs regarding completion of rescues, so in these cases upon completion of the simulation, a number of medic agents were unaware of the accurate status of the completed rescues.

If we focus on the time factor alone and compare the performance across each category of organisational support, we see in Figure 7.19 improved performance with more organisational support. The run times seem to be best at organisation level 4. This level has full organisation support provided to agents. At level 5, when 4 agents are involved and a medic begins a stretcher rescue then hands over to an officer to carry the stretcher, this adds time to send messages to agents about this change and delegation as well as time to wait for the officer to meet the rescue partner (who in many cases had already begun to move toward the dropzone) and effectively begin the rescue again. This is a limitation in the way the agents actions were coded.

If we focus on the total number of rooms entered by medic agents during the successful 4 room simulation runs and compare the performance across each category of organisational support, we also observe in Figure 7.20 improved performance with more organisational support. Fewer rooms were entered in runs at organisational support levels 4 and 5. Level 5 is more consistent with better performance in terms of rooms entered. Figure 7.20 shows that many of the cases with zero organisation support were either unsuccessful or successful, but with agents unaware of the success.

Stretcher rescues with zero organisational support were set up with no medics in an organisation and so no sharing of beliefs or rescue status amongst medics. The medics chose partners from any other medics available based on agent capabilities. In Figure 7.19, some of the times for completion with zero organisational support look positive, however in these cases, the level of unawareness was high. The unsuccessful runs that did not complete were given an arbitrary maximum time of 600. These can be ignored. The results indicate an improvement of time performance with more organisational support. There is not a big difference between simulation times at level 4 and 5 for 3 agents. When 3 agents are involved, only 1 stretcher rescue can be done at any time as two agents are required for the rescue.

Figure 7.21 shows corresponding communication data from the 4 room rescues runs for each organisational category. As the organisational level increases, so does the number of messages sent and received by agents. This is not surprising as the organisational level increases, then obligations to share information with others in the organisation increase. Comparing this data to the overall time taken in these situations the time taken decreases with organisational support. This suggests that even though there is an increase in the messages sent between agents, this communication is effective

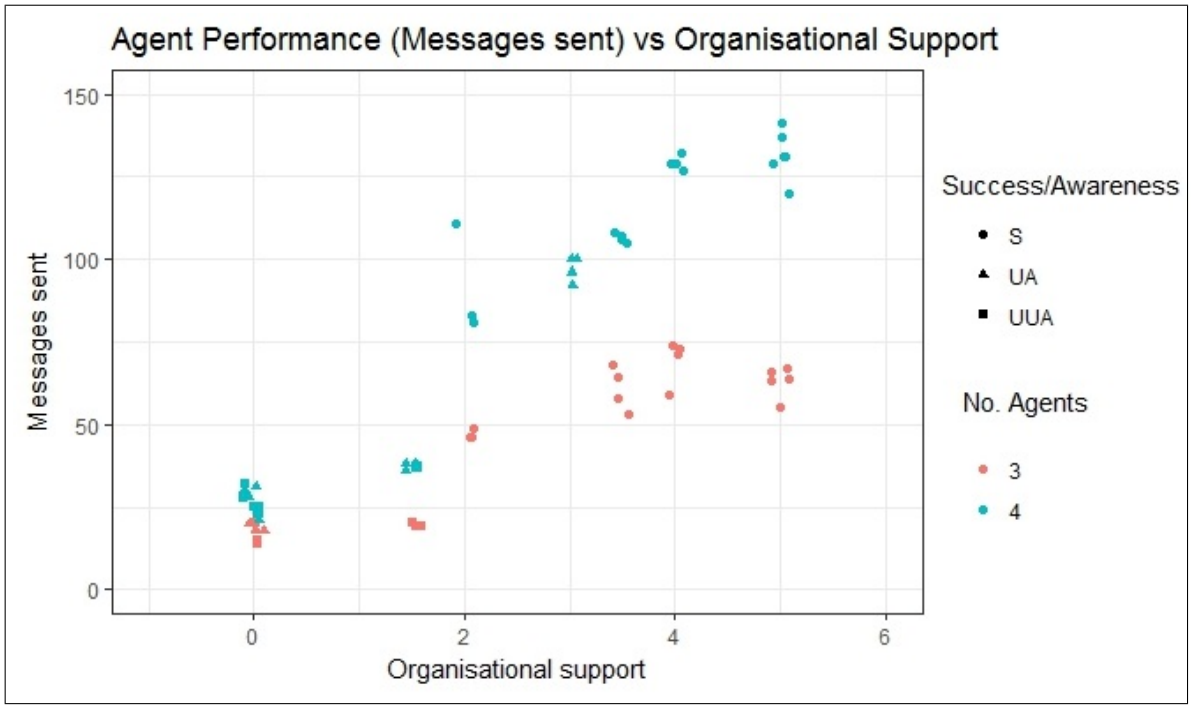


Figure 7.21: Agent messages count for categories of organisational support

and targeted as it reduces the overall time. This may not be the case for all domains or situations and would need further testing if larger numbers of agents were involved.

The complex room configuration involving 12 rooms and 5 stretcher rescues amongst 5 medic agents provides us with an opportunity to investigate more closely the social policies within an organisation to scrutinize whether there is any pattern in the results suggesting whether the trade off between the extra overhead of communication driven by organisational policies is beneficial to improve performance. The following graphs will focus on this data comparing averaged results from runs with no organisational support with averaged results where agents are afforded the flexibility to handover stretcher carrying tasks to officers. This data highlights any costs in terms of time or resources of allowing this run time flexibility.

An organisational level of zero indicates that there were no organisations created to help the medic agents or oblige the agents to share information. An organisational level of 6 indicates full organisational support with sharing of beliefs and rescue completion but no stretcher handover.

The Organisation structure categories from 7-9 indicate the level of flexibility afforded agents by social policies around the delegation of a stretcher carrying task. Organisational level 7 allowed delegation of the stretcher carrying task from a medic to an officer only if an officer is nearby. In level 8, the stretcher carrying delegation to an officer is automatic, but only if an officer is available. It will only occur in 3

rescues, because the officers can each only help in one rescue and then say they are unavailable. In this case, medic agents also shared their rescue intentions with others in the organisation, so if multiple pairs had adopted the intention to perform the same rescue, the agents would change their intentions to adopt a different rescue. Level 9 organisation structure represents that the stretcher handover only occurs in response to a busy message sent by the original stretcher carrying partner.

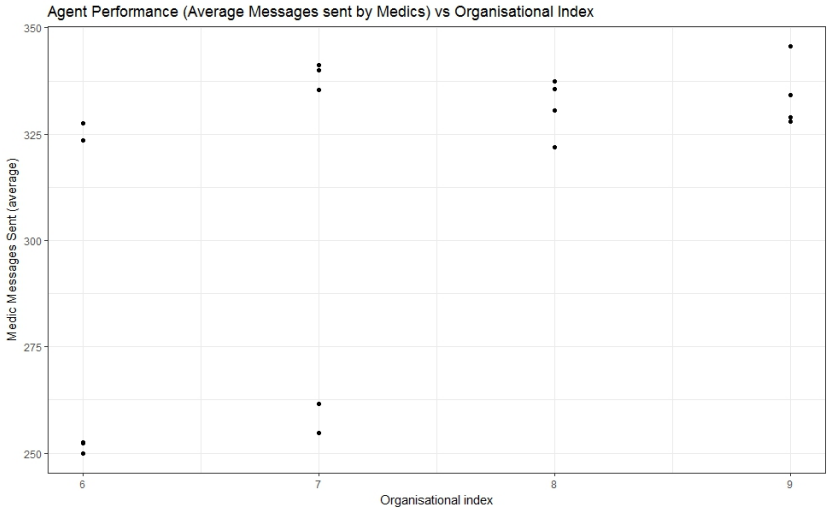
Figure 7.22 provides 4 graphs showing the number of messages sent, number of rooms entered, time taken and time spent idle by agents during the more complex simulation runs with index values 6-9. These involve simulations with 12 rooms, 5 medic agents, 5 seriously injured requiring stretcher rescues and various policies for stretcher handover. These results show that performance of agents can be affected by introducing policies.

The policies used in organisation indexes 8 and 9 seem to have positive impacts on the idle time of medic agents and the efficiency in terms of the total number of rooms entered by agents. There is some variability in different runs regarding time taken to complete the simulation, however it appears that on average, the time taken improves when policies for handover are introduced.

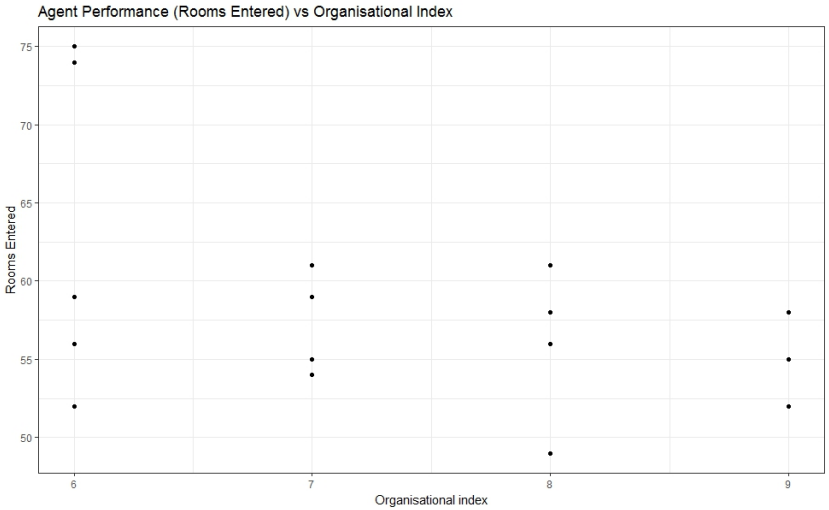
The data indicates the number of rooms entered decreases with more organisational support and flexibility in terms of delegation of the stretcher carrying task. The optimum performance is with level 9, where agents have the flexibility to respond to a potential delay and will only delegate in response to a busy message. When there is no organisation existing for the medic agents, the number of rooms entered is much higher as is also the time taken to complete the rescue. The best time performance occurs when there is full organisational support but no stretcher handover at all (level 6).

As with earlier data in the 4 room set up, the number of messages sent by all agents increases with more organisational support and flexibility. This is due to the requirement that agents negotiate and communicate in order to perform handover of their tasks.

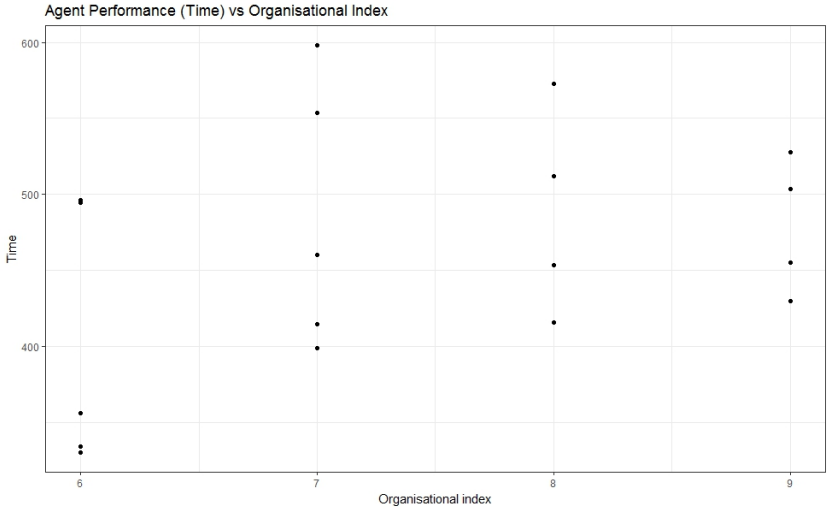
The behaviour of the MAS model described in this chapter provides demonstration of the value in having an organisation structure and using that to specify policies to help agents with coordination, using the OJAzzIC meta-model. There is also demonstrated success in allowing agents to improvise by adopting a goal to reach a task that is not in the agent's role responsibility. For example, the idle officer agent was able to step in and help a medic agent to complete a stretcher rescue when asked. If the officer was asked to carry a stretcher because a medic was unavailable, the time taken improved to produce the best overall performance.



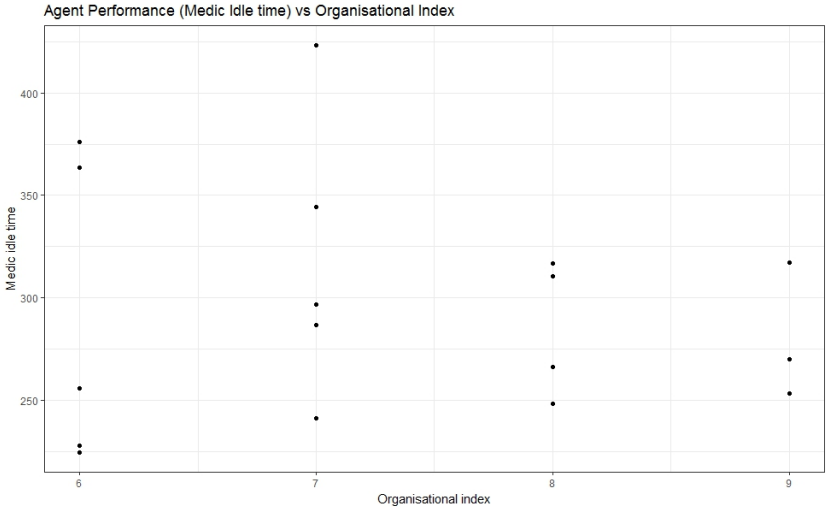
(a): Agent Messages count for each Organisation index



(b): Rooms Entered for each Organisation index



(c): Time taken to complete rescue for each Organisation index



(d): Medic Idle times for each Organisation index

Figure 7.22: Agent Performance in 12 room stretcher rescues for Organisation index 6-9

7.6 Discussion

The decision to build in capacity for flexibility is a trade off in terms of the design effort and should be informed by the requirements of a problem. It may depend on the level of uncertainty and how much brittleness is acceptable at run time in terms of a lack of flexibility. If flexibility at run time is required and the exact circumstances at run time cannot be directly predicted, then designing for flexibility can help agents adjust and cope better to avoid potential failure at run time. OJAzzIC matches capabilities to tasks to enable agents to adopt tasks and work together to achieve a shared goal. This required coordination.

Implementing a MAS based on the OJAzzIC meta-model provided the ability to think about policies during design that could help agents at run time. As agents in the OJAzzIC organisation apply the coordination policies within their organisation(s), the organisations provide a context for agent interaction and help agents to know who else to share information with. The benefits of this interaction and knowledge cultivation have been visible in the simulation system. Even though agents within the organisation were obliged to send more messages than if they were not in an organisation, the overall performance increased when agents were in an organisation.

Because tasks are not limited to agents specified in organisation roles at design time, agents could use initiative to adopt a task to complete that was outside of their allocated role (if they had the capability). Giving an officer agent an extra capability to carry a stretcher at design time, enabled that officer to take part in helping a medic agent with a stretcher carrying rescue if needed, whilst still enacting the role of an officer. Policies could be implemented explicitly to specify the conditions when the officer agent could adopt a goal to reach a task that is outside of the current role. The use of policies to explicitly state conditions for agents at design time, but then allow the agents autonomy to respond at run time based on the situation seems to be a benefit.

In other meta-models, this would be possible if the designer allocated the officer to possibly enact either the rescue role (medic) or the officer role; or if the officer role was broadened to include the responsibility of stretcher carrying. The officer agent would potentially need a mechanism to prioritise which task or role to fulfil at run time and the officer agent would need to be given all the capabilities to match all potential roles enacted. The designer would need to make the choice at design time as to specify how the officer agent will act at run time. This prioritisation would be embedded implicitly in the officer agent.

The stretcher rescue task in the demonstration system explored coordination be-

tween agents using roles. Coordination between agents using roles adopted was achieved by specifying a high level plan at design time then allowing agents to adopt roles or tasks at run time by mutual adjustment. This approach was successful. Alternative approaches could be:

1. using roles fixed at design time by allocating agents to roles and specifying role responsibilities using detailed plan scripts,
2. specifying particular roles at design time then agents' adopting (or being allocated) roles at run time,
3. allowing role allocations to be entirely planned and negotiated at run time.

The first option lacks flexibility at run time. The second option gives some flexibility but requires that agents are available to match a role exactly as specified, so in the case of an unanticipated failure of the design, does not allow agents to find a solution. The third option option relates to the vast planning literature and requires complex agent deliberation at run time or an external system to perform the allocations of agents to tasks.

Agents in OJAzzIC possess organisation awareness to know who else is in the organisation and this awareness enables focused interactions to coordinate beliefs and intentions. OJAzzIC enables agents to coordinate dynamically at run time using explicit direct communication between agents. Agents make explicit commitments using SharedPlans, created at run time. The flexibility this provides allows agents at run time to negotiate a SharedPlan and agree on low level coordination rather than scripting detailed coordination at a task level during design. Specifying tasks below a goal gives some flexibility.

The use of SharedPlans commitments in order to coordinate a plan is demonstrated in the behaviour of agents in the MAS developed. Agents at run time were able to autonomously decide which agents were agreeing to complete a particular stretcher rescue, this was not pre-scripted. Agents created SharedPlans commitments at run time to agree on rescues and agents negotiated using mutual adjustment if necessary to adopt complementary roles of stretcher carrying agent or pick up agent. In other meta-models, this would require the designer or external coordinator to specify the coordination and task allocations

As discussed early in the thesis, coordination can be managed by people in different ways. This can be mirrored in components of a meta-model. With a less complex task, the solution can be scripted with more detail specified in plans at design time and not relying on a need for flexibility at run time. However with a more complex task, flexibility is important. Roles, goals and plans defined at design time can be used to

specify various levels of control for coordination in a MAS as follows:

- Mutual adjustment : Agents adjust at run time
- Direct Supervision : Design time plans provide coordination
- Standardisation of work processes : Roles explicitly define process
- Standardisation of skills : Roles explicitly define how agents behave
- Standardisation of outputs : Goal states or landmarks clearly defined
- Standardisation of norms : norms or policies specify behavioural expectations

We believe that separating concepts and making them explicit as outlined in the OJAzzIC meta-model is helpful to the designer and gives more autonomy to agents at run time.

When mutual adjustment can occur at run time so agents are not tied to a fixed script or allocation specified at design time, this gives agents the ability to adapt to the run time context to create a workable solution. This may involve allowing agents to show initiative by adopting goals to achieve tasks outside of a role specified at design time. It may also require agents to use a high level plan to guide them and then agree on roles dynamically, similar to the agents in the simulation system performing the stretcher carrying task.

The behaviour of agents in the MAS system also demonstrated that policies or other tie-breaking mechanisms for avoiding deadlock are required when agents are dynamically allocating roles or dynamically negotiating to agreeing on a plan. We found that allowing one agent to adopt a role and then have the other agent mutually adjust to that decision by swapping roles if required was an effective tie-breaking policy. This was more effective than agents persevering and attempting to 'talk about' or negotiate about who was going to do which role.

Agents sharing a mental model of a situation is important so that agents have an up-to-date model of the actual situation - not only in terms of task completion (e.g. who is rescued, who is yet to be rescued) but also in terms of intentions (who is intending to rescue). Without agents sharing beliefs about task completion and task intentions agents behave with inconsistent beliefs and may interfere with other agents. Also, in some cases, when agents did not share beliefs, agents were observed behaving in a non-productive way (e.g. waiting to perform a stretcher rescue that no longer needs to be done and not progressing to perform simple rescues that are required).

The data presented in the previous section indicates that performance is more efficient in the achievement of the objectives when agents belong to an organisation and have therefore the support of organisational policies to ensure coordination of

knowledge and plans. We measured performance in terms of time and the number of rooms entered. In runs where the agents were provided the full features of the OJAzzIC meta-model implementing policies for sharing beliefs and completion state on rescues between agents in the organisation, the best performance was observed. When we first added in the possibility that a medic agent might handover by delegation the stretcher carrying task (organisation level 5), initially, this was found to not improve performance and in some cases, it resulted in an increased completion time. Observing the behaviour of agents following a decision to handover the rescue carrying task, due to the design of the code implementing the stretcher rescue, in some cases, the two new collaborating rescue agents backtrack their positions to begin the rescue that had begun again. This added time to the simulation run in these cases. When a handover to an officer was used in response to a situation where a medic suddenly became unavailable and couldn't fulfil a previous SharedPlan, so informed its partner it was busy, then performance improvements were observed.

In order to examine the effect of enabling handover and delegation of the stretcher carrying, attention was given to the more complex scenarios involving 12 rooms. The results from these scenarios indicated that there is a time and communication cost involved in enabling agents the flexible run time behaviour of handover when comparing to organisational agents who do not handover. However the overall performance in terms of time is improved with the introduction of policies that allow agents to respond adaptively to the situation at run time. When agents are informed of the rescue intentions of others, they can respond to adjust their own intentions. This improves time and number of rooms entered, however does increase the number of messages sent. The ability to adjust social policies at design time to enable agents at run time to adapt and improve their performance is a trade off and the relative cost of each (entering a room or sending a message) in any particular situation needs to be considered.

Adapting the O-MaSE methodology to work with the OJAzzIC meta-model (see section 6.2) provided for the creation of policy definitions at design time to establish appropriate social contracts between agents at run time and to guide agents with run time flexibility. In particular, we introduced policies for information sharing, policies defining when agents could improvise to handover tasks to others or adopt goals based on capabilities rather than roles and policies for creating adhocracies. The design process also enabled us to specify the need for SharedPlans to be created at run time to make agents commitments to a stretcher rescue explicit. This enabled medic agents in the executable model to decide dynamically at run time which stretcher rescue they were going to adopt and who they would work with to complete that rescue. This was not pre-scripted.

The OJAzzIC meta-model provides the flexibility to be useful in a decentralised organisation that requires some level of dynamic run time coordination. Information sharing policies can be specified to obligate agents to share information with the appropriate set of other relevant agents based on a context defined at design time that is instantiated at run time. Similar behaviour could be achieved with other meta-models, by embedding coordination into plans at design time or by specifying interactions more explicitly at design time.

The benefits of OJAzzIC include that the coordination can be dynamically agreed between agents at run time. The flexibility at run time is based on higher level meta-design decisions implemented by policies based on the run time situational context. This flexibility relies heavily on good policy design. It would be valuable to explore further the best form of organisation and benefit of using organisations in larger configurations with more agents and more complex tasks. This would more explicitly test and highlight limitations in the meta-model that are not made clear in our demonstration system.

The focus in the demonstration system scenario was to test the ability for agents to dynamically negotiate SharedPlans at run time in order to elaborate on high level plans defined at design time. The MAS implemented does not include a dynamic role model and it does not have agents that can potentially adopt multiple roles. It also does not define goals that can be achieved by multiple pathways. These features of the meta-model should be further explored with more complex scenarios and simulations. These limitations will be discussed further in the final chapter.

Chapter 8

Conclusions

8.1 Overview

In this final chapter, the research question introduced early in the thesis is revisited in section 8.2. We highlight the thesis findings in section 8.3 and point to distinguishing features in the OJAzzIC meta-model in section 8.4. The contributions of in the thesis are mentioned in 8.5 and future work discussed in section 8.6.

8.2 Revisiting the research question

The research question asked in Chapter 1 was:

> "What elements are needed in the design of organisations of software agents, so that agents coordinate the sharing of knowledge and plans, and can demonstrate run time improvisation based on the design?"

This question was further refined in order to direct our research. We focused on meta-models giving consideration to the following issues:

1. What information needs to be available to agents about the domain, the organisation of software agents, the intentions and actions of other agents?
2. What elements need to be specified in a MAS at design time and what can emerge at run time?
3. What needs to be specified so that agents can improvise to take on a new role that needs filling when they have the capability to do so?
4. What needs to be specified so that multiple agents can coordinate their behaviour achieve a joint task or share a role that cannot otherwise be allocated?

To address the research question, we identified components that deal with these requirements and developed a meta-model for an organisation of agents. The OJAzzIC meta-model enables improvised coordination between agents by using the following main concepts:

- a goal-task design that enables agents to adopt goals based on capabilities as well as roles;
- when agents need to coordinate at run time they use run time commitments based on SharedPlans;
- policies written at design time guide agent behaviour at run time;
- agents deliberate with some awareness of organisational goals; and
- the organisation is used as a context to help agents identify relevant others and agents can form adhocracies at run time.

The policies specify obligations regarding knowledge cultivation and coordination. Within an organisation, at run time, agents can mutually adjust intentions and coordinate with others to agree on a solution using a SharedPlan. The adaptation can involve a re-organisation of adopted goals or improvisation beyond the roles defined at design time. The flexible representation of the functional model in terms of goals and tasks allows agents some flexibility. Additionally, allowing agents to adopt tasks outside of role specifications provides flexibility for improvisation.

Run time adaptation can also involve the formation of new organisations, referred to as adhocracies. Within adhocracies and design time organisations, agents are bound by policies written at design time to specify what information should be shared and with whom. The context for who to share with is based on membership of the organisation. When agents decide to work together on a joint task, using the construct of a SharedPlan and explicit commitments between agents meant that agents could dynamically adopt the plan details in terms of which agent would adopt which role.

The organisation meta-model includes dimensions of structure, function, interactions and behavioural policies. It was necessary to address mechanisms for how and when to create and populate adhocracies to support the desired behaviours. The approach taken in OJAzzIC places significant responsibility on the designer to design with flexibility in mind.

The OJAzzIC meta-model provides components to address:

- knowledge sharing between agents;
- agent awareness of the organisation's goals and roles;
- coordination policies;

- improvisation and
- individual agent capabilities.

These components then lead to the development of a system that is capable of flexible, improvised, coordinated behaviour.

We used the meta-model to design and implement a demonstration MAS system. This system demonstrated that the meta-model is sufficiently expressive to produce the desired behaviours in a MAS. The MAS responds to a simulated disaster recovery incident by conducting a coordinated search and rescue in a virtual world. Agents in our system successfully demonstrated the expected behavioural outcomes: improvisation and flexible coordination. A number of different scenario settings were used and multiple runs were observed. Agents performed better when the design included organisations with organisational policies for knowledge cultivation and adaptation. The system was run with a number of different scenario configurations to examine performance for a number of different levels of organisational support. Each configuration used different policies for knowledge and intention sharing. We also tested policies that allowed agents to adapt. For example, we used policies to allow an agent to find a new partner if a planned partner for collaboration on a joint task was unavailable.

8.3 Findings

Returning to the original research question introduced in section 1.3 and noted in section 8.2, in this section, we highlight that the OJAzzIC meta-model and associated design process address our research question and address the limitations of existing meta-models with regard to our requirements.

What information needs to be available to agents about the domain, the organisation of software agents, the intentions and actions of other agents? Agents have access at run time to information about the organisations they belong to. This information is specified in the OJAzzIC meta-model and includes membership, organisational structure, policies and interactions with others. Agent awareness is facilitated by the design of multiple OJAzzIC organisations. Providing this information in the meta-model facilitates flexible, coordinated agent behaviour and if required, improvisation at run time. Policies for knowledge cultivation also ensure that agents have consistent mutual knowledge within the organisation.

At a domain level, agents need to know how to take actions in order to achieve tasks or goal objectives. Tasks are associated with the decomposition of goals representing

objectives that are to be met by the organisational system. The organisation is a first class entity and agents can refer to this to learn which agents are members of the organisation. Within an organisation, the policies can specify agent obligations in terms of creating adhocracies, specifying interactions with others and obligations to share knowledge, create SharedPlans, share outcomes and share intentions.

Our results (see section 7.5) demonstrated that agents did benefit from being part of an OJAzzIC organisation. The OJAzzIC organisation provided policies to guide interactions between agents for sharing beliefs and creating SharedPlans. The policies for knowledge cultivation within the organisation, as well as policies enabling handover or delegation of tasks to another agent when necessary enabled agents to behave with some flexibility at run time. When these policies were disabled, the agents did not always experience success and performance was comparatively weaker. Using SharedPlans and explicit commitments between agents created dynamically gave agents the ability to adjust and fit in with each other.

What elements need to be specified in a MAS at design time and what can emerge at run time? At design time, the OJAzzIC meta-model allowed us to specify our problem and to model the agents with sufficient specificity so that at run time agents could select goals based on their roles or capabilities. Applying the OJAzzIC meta-model and design process, we specified policies at design time to guide the behaviour of agents at run time. Policies stated at design time explicitly stated obligations for agent to share beliefs and intentions with others at run time.

OJAzzIC requires high level top down plans in terms of goals and a decomposition of goals into required tasks at design time. Tasks or roles could align with agents' capabilities. This allows agents at run time to decide on operational details such as who is doing each particular task. At design time, organisation(s) are specified. The specification of the organisation includes roles that can be allocated to or adopted by agents. A list of member agents are assigned to the organisation and each agent can possess individual capabilities and/or be allocated roles within the organisation. Agents enacting roles can work toward achieving the organisation's goals. In addition to roles, capabilities can be specified at design time and aligned with tasks so that agents at run time can use this information to allow flexible behaviour to emerge in a bottom up way. At run time, capable agents can improvise to adopt individual goals to achieve tasks outside of roles specified.

When a joint task requires two agents to coordinate, the agents can at run time agree on a SharedPlan which specifies the roles each will adopt and creates commit-

ments between the agents regarding their SharedPlan. This provides a framework for coordination. At run time, adhocracies can be created when agents agree to work together to achieve a shared goal but were not in an organisation specified at design time.

What needs to be specified so that agents can improvise to take on a new role that needs filling when they have the capability to do so? We found that the OJAzzIC meta-model provided a flexible organisation structure that enabled a design to be successfully adapted at run time. Using explicit policies as part of the organisation framework enabled us to explicitly describe a meso-layer of rules to guide agents at run time. This assisted agents in run time deliberation regarding goal selection. Agents were guided by policies describing when it was permissible to adopt a goal to complete a task outside of a predetermined role responsibility. Enabling agents to adopt tasks allowed agents to collaborate and agree if necessary to create SharedPlans to coordinate shared responsibilities to fulfil an objective.

When deliberating to consider which goals to adopt, agents can use awareness of their own capabilities to match required tasks or goal objectives. Agents can then adopt goals to achieve the required objectives that would otherwise be achieved by an agent in a role specified at design time. This facilitates agent improvisation when there is not an agent available to match a role specified at design time. Social policies can be used for example, to specify tasks that agents can adopt using improvisation based on capabilities or to prioritise the way that agents consider the adoption of tasks. These policies guide agent behaviour so that agents at run time can find solutions that might not have been specified with design time roles. We demonstrated in our simulation that policies can be used to define situations where agents might be able to improvise in response to a change in situation. We used policies that allowed agents in our scenario to delegate or handover a task to another agent with the necessary capabilities. This flexibility was found to be of benefit when another agent who had originally agreed to collaborate on a joint task was busy. This is discussed in section 7.5.

What needs to be specified so that multiple agents can coordinate their behaviour to achieve a joint task or share a role that cannot otherwise be allocated? There may be situations where a goal is specified at design time and it is anticipated that this will be achieved by one agent enacting a particular role, however at run time this is not possible. If an agent with capabilities exactly matching that role, is not available at run time, then individual agents can match their individual capabilities against tasks that comprise the original goal. Agents can then adopt individual goals

based on their intentions to achieve these tasks. When agents need to coordinate behaviour at run time, they create SharedPlans and if necessary new adhocracies. Agents coordinate knowledge at run time using policies that are specified at design time. The creation of SharedPlans and adhocracies at run time is facilitated by the use of policies defined at design time and implemented plans that ensure agents have an ability to deliberate to create SharedPlans.

The behaviour of agents in the demonstration system built using the OJAzzIC meta-model and design demonstrates that agents performance is improved with policies specified to guide knowledge cultivation and flexibility. In situations that require a flexibility and improvisation by agents to adjust and fit in with each other at run time, it is helpful to specify functional components at a low level so that allocation/adoption at run time can be adjusted dynamically based on available agents and their capabilities. Using an organisational structure provided context to enable targeted communication between agents. We achieved flexibility by specifying solutions using high level plans described by task outputs in a decomposition of tasks based on goals. We then allowed agents some flexibility at run time about how those tasks were completed and by whom. Adopting the OJAzzIC meta-model clearly facilitated flexibility in run time behaviour.

Organisational responsibilities specified in policies increase the communication demands on agents. This could have a more significant impact in larger organisations or in situations where communication is costly. This needs to be considered when choosing policies for information sharing. Currently, the OJAzzIC meta-model relies upon homogeneous agents being part of one organisational system so that agents are aware of the existence of other agents. A limitation to this approach is that it is only applicable to situations where all agents have the same shared meta-model and agent design. However, the intentional approach requiring agents to explicitly declare their commitments to other agents externally does lead toward a possibility where heterogeneous agents could use their individual mental attitudes to target goals and then be guided by policies to make commitments public to other agents.

8.4 Distinguishing OJAzzIC from other meta-models

A comparative analysis was presented in Chapter 4.2 of the predicted ability of a number of organisation meta-models to address the chosen requirements. Other meta-models do identify goals and define multiple solution paths at design time to be selected from at run time (OMACS); use the organisation as a first class entity so that agents can identify who else is in the organisation and identify roles and responsibilities (OMACS,

JaCaMo+, OperA+); specify capabilities for agents (OMACS); and use explicit contracts (OperA+) or commitments between agents at run time (JaCaMo+).

Other meta-models do not allow agents flexibility to adopt a task or objective that is outside of an agent's role. This means that if agents are to have any flexibility at run time in terms of initiative or improvisation, then agents must be allocated to more potential roles at design time, or the scope of responsibility for roles would need to be broadened. This would introduce the need to specify mechanisms for agents to prioritise competing roles or responsibilities at run time. In OJAzzIC the main role an agent is designed to fulfil is identified, but the agent can be given permission using explicit policies to adopt a goal to achieve something outside of its role. The explicit nature of the policies means that improvisation is a clear design decision rather than an implicit coding decision. The OJAzzIC meta-model does not explicitly manage the possibility of an agent concurrently enacting multiple competing roles, however it does not exclude an agent having multiple potential roles that could be enacted. Prioritisation of goal selection is handled with policies specified at design time.

OJAzzIC uses the organisation to identify context for agent awareness and interactions so that agents know who else should be informed about beliefs and intentions. Other meta-models examined do not allow the creation of organisations at run time. This means that the organisation is useful as a construct for allocating responsibilities and specifying roles, but the organisation is not used to help give agents awareness regarding knowledge cultivation. Other meta-models do not use the organisation construct to identify relevance for interactions. Other meta-models do not use policies to explicitly obligate aware agents to interact to share beliefs and intentions. JaCaMo+ uses role protocols in a similar way but the agents do not directly interact with each other.

Other organisation meta-models examined do not use commitments at run time to allow agents to develop dynamic plans. This means that agents must adopt a plan that is specified at design time and when coordination is required the design time specifications must implicitly allocate agents to roles within the plan. This does not give agents flexibility at run time unless multiple context based plans are written at design time. JaCaMo+ uses commitment protocols and allows agents some ability to withdraw from a commitment, however the commitment protocol specifies how agents will coordinate, rather than allowing agents at run time to agree on roles or operational responsibilities.

8.5 Contributions

The major contribution of this thesis is to identify elements needed in the design of a MAS agent based organisation so that agents address the requirements of improvised coordination. This is presented as a meta-model: OJAzzIC. The design process is presented and demonstrated. We have suggested a set of policies that should be adopted to guide agents to coordinate within an organisation.

In the meta-model, OJAzzIC, the organisation structure includes low level elements such as capabilities and tasks. Agents can use these low level constructs to match to their individual attributes to help provide flexibility with the adoption of goals. This facilitates run time flexibility so that agents are not required to exactly match a role description at a higher level in order to adopt a goal.

The OJAzzIC organisation structure is specified at design time and includes high level plans that are defined at design time with goals and tasks that are specified at a lower level. At design time, choices are made about the decomposition of system goals, the potential organisations and organisational roles that might be enacted to adopt goals. Lower level information about tasks and associated matching capabilities is also specified. The abstraction of capabilities associated with tasks in addition the specification of design time roles gives agents flexibility to adapt at run time.

During the design of an OJAzzIC organisation policies are specified to guide agents at run time. Policies can help agents with coordination of beliefs and intentions. Policies can also guide agents regarding tasks that could be achieved by an agent improvising outside of their role. If agents find at run time that it is not possible to adopt a role with an exact match to capabilities of the available agents, then because of the lower level information provided in OJAzzIC at design time, agents can display some emergent behaviour and improvise. An agent who is not in an allocated role can adopt a goal if they have the capabilities required. Also if an agent has appropriate authority, they can request help from or delegate to another agent with the appropriate capabilities.

At run time, agents can create a new organisational adhocracy. The adhocracy has all the features of a design time organisation and can be instantiated based on default design time parameters. The adhocracy can be created to join agents who were for example, not already in an pre-designed organisation, but who, at run time agree to work together to achieve a shared objective. At design time, triggers for the creation of adhocracies need to be specified. Creating adhocracies at run time enables agents to be supported by the organisational constructs - knowledge of who is in the organisation, knowledge of policies that will facilitate coordination of knowledge and coordination of

intentions using SharedPlans.

The OJAzzIC meta-model proposed in this thesis indicates components that should be considered in the creation of OCMAS in situations that require improvised coordination.

8.6 Future work

The OJAzzIC model has some significant limitations that are mentioned in this section.

In OJAzzIC, it is assumed that agents will trust others and believe information shared and immediately update their own beliefs. The meta-model does not include any provision for managing or prioritising inconsistent beliefs between agents. For simplicity, it is assumed that each agent has individual attitudes that will be consistent with others in the organisation based of the interactions between agents. It may be better that agents can model uncertainty in mental beliefs (Pinyol and Sabater-Mir 2013).

There is significant onus on the OJAzzIC designer to design a system that will in fact demonstrate the flexibility desired at run time. The designer must also consider potential failure of the design and anticipate policies that should be created for improvisation. Due to the nondeterministic nature of the domain environment, agents should detect if something fails and create new goals to resolve it. This will only work if the MAS design provides a number of different alternative goal pathways for agents to select from. If an agent does not have a particular potential goal to adopt matching a run time context, the system will fail. Semantics for success and failure are considered as part of JaCaMo (Boissier, R. H. Bordini, J. Hübner, Ricci, and Santi 2013). This work is important but not currently addressed in OJAzzIC.

It is important that agents in the domains considered find a workable solution at run time. The focus at design time is to consider specifying multiple alternative paths so that at run time at least one valid path might be possible. Flexibility is provided by providing alternative paths in a goal decomposition tree and allowing agents to select any valid path, however as more pathways are provided, it is possible that more than one plan (goal pathway) may be valid for a particular context. The meta-model does not include a mechanism for agents to use when there are potentially multiple conflicting or competing goals valid. This is a realistic concern when agents are members of multiple organisations. Further work is need to explore if this can be dealt with by good design principles and policies or if agents need an explicit additional mechanism to deal with managing priorities for goal selection.

There is potential value in incorporating an explicit planning process in the deliberation used in BDI systems (Sardina, De Silva, and Padgham 2006). This is not considered in OJAzzIC. As the human experience in domains of interest suggests that humans do not necessarily adopt an explicit plan formation stage, we chose not to incorporate a planning process. However, planning could be necessary to help agents choose a solution at run time if multiple design time options are valid.

If there are multiple pathways valid, and this cannot be anticipated and dealt with by using prioritisation policies, then there must be time available for deliberation. The current OJAzzIC meta-model would rely on policies to guide whether agents should deliberate on planning or just select the first available valid option. This has not been addressed specifically in the OJAzzIC meta-model. Incorporating look ahead planning and failure handling in BDI systems is important and has been explored by others (Sardina and Padgham 2011).

The proposed OJAzzIC meta-model has only been implemented in one small demonstration system. As discussed in section 7.6, the OJAzzIC meta-model offered features to enable flexibility and improvisation by agents in the demonstration system. Further evaluation using different domain scenarios and implementation models to establish if it is a valid meta-model to apply more broadly. Further work is needed to explore whether it is possible to identify policies that should be included as generic domain independent policies or whether policies are particular to the specific problem domain. The handover policies implemented in the demonstration system (Section 7.3.1) assisted agents to achieve more flexibility when agents had the autonomy to choose their plan of action based on the run time context. This may suggest that domain specific policies to drive plan selection are of value. The current demonstration model has some limitations in terms of complete testing of the meta-model components. The meta-model could be further explored using a more sophisticated simulation implementation platform such as BRAHMS (Sierhuis, C. Jonker, B. v. Riemsdijk, and Hindriks 2009).

It would be interesting to see if the OJAzzIC meta-model concepts could apply in open systems where agents might be in a situation where they could dynamically join an organisation and adopt goals toward reaching the organisation goal then register their capabilities to coordinate behaviour between agents at run time to allow a solution to emerge. The soccer team scenario and BRAHMS system described by Sierhuis (Sierhuis, C. Jonker, B. v. Riemsdijk, and Hindriks 2009) could be a scenario and implementation platform to adopt for this purpose. It would be interesting to explore the possibility that the OJAzzIC meta-model could become part of a more general

model that could be applied to open agent systems and service oriented architectures. One possibility is that the OJAzzIC organisation meta-model could be used as the basis for organisations incorporated into other generic models. In an open environment, agents could potentially learn about the capabilities of other agents dynamically rather than this being provided at design time (Y. Zhang, Sreedharan, and Kambhampati 2015). This would lead toward agents being able to dynamically create adhocracies and coordinate behaviour in an emergent way toward reaching a solution.

References

Abdalla, R. and A. Mishra (2018). "Comparing the Artifacts of Agent Methodologies". In: *TEM Journal* 7 (2), pp. 433–438.

Adam, C. A., G. Danet, J. Thangarajah, and J. Dugdale (2016). "BDI modelling and simulation of human behaviours in bushfires". In: *International Conference on Information Systems for Crisis Response and Management in Mediterranean Countries*. Springer, pp. 47–61.

Adam, C. A. and B. Gaudou (2016). "BDI agents in social simulations: a survey". In: *The Knowledge Engineering Review* 31.3, pp. 207–238.

AFAC (2004). *The Australasian Inter-service Incident Management System*. Australian Fire Authority Council. URL: `http://www.afac.com.au`.

Agmon, N. and P. Stone (2012). "Leading Ad Hoc Agents in Joint Action Settings with Multiple Teammates". In: *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems AAMAS - Volume 1*. AAMAS '12. Valencia, Spain, pp. 341–348.

Aldewereld, H., O. Boissier, V. Dignum, P. Noriega, and J. Padget, eds. (2016). *Social Coordination Frameworks for Social Technical Systems*. Vol. 30. Law, Governance and Technology Series. Springer, Cham.

Aldewereld, H., V. Dignum, C. M. Jonker, and M. B. van Riemsdijk (2012). "Agreeing on role adoption in open organisations". In: *KI-Künstliche Intelligenz* 26.1, pp. 37–45.

Amir, O., B. Grosz, K. Gajos, S. Swenson, and L. Sanders (2015). "From Care Plans to Care Coordination: Opportunities for Computer Support of Teamwork in Complex Healcare". In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pp. 1419–1428.

Argente, E., V. Botti, and V. Julian (2011). "GORMAS: An Organizational-Oriented Methodological Guideline for Open MAS". In: *Agent-Oriented Software Engineering X*. Ed. by M-P Gleizes and J.J. Gomez-Sanz. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 32–47.

Ashamalla, A., G. Beydoun, and G. Low (2017). "Model driven approach for real-time requirement analysis of multi-agent systems". In: *Computer Languages, Systems & Structures* 50, pp. 127–139.

Au, A. (2011). "Analysis of command and control networks on Black Saturday". In: *Australian Journal of Emergency Management* 26.3.

Baldoni, M., C. Baroglio, F. Capuzzimati, and R. Micalizio (2018a). "Commitment-based Agent Interaction in JaCaMo+". In: *Fundamenta Informaticae* 159.1-2.

— (2018b). "Type checking for protocol role enactments via commitments". In: *Autonomous Agents and Multi-Agent Systems, JAAMAS* 32, pp. 349–386.

Balke, T. and N. Gilbert (Oct. 2014). "How Do Agents Make Decisions? A Survey". In: *Journal of Artificial Societies and Social Simulation* 17.4.

Beydoun, G., G. Low, B. Henderson-Sellers, H. Mouratidis, J-J. Gomez-Sanz, J. Pavón, and C. Gonzalez-Perez (Nov. 2009). "FAML: A Generic Metamodel for MAS Development". In: *IEEE Transactions on Software Engineering* 35.6, pp. 841–863.

Bigley, G. A. and K . H. Roberts (2001). "The Incident Command System: High Reliability Organizing for Complex and Volatile Task Environments". In: *Academy of Managment Journal* 44.6, pp. 1281–1299.

Boissier, O., R. H. Bordini, J. Hübner, and A. Ricci (2014). "Unravelling Multi-agent-Oriented Programming". In: *Agent-Oriented Software Engineering*. Ed. by O Shehory and A Sturm. Heidelberg: Springer-Verlag.

Boissier, O., R. H. Bordini, J. Hübner, A. Ricci, and A. Santi (2013). "Multi-agent Oriented Programming with JaCaMo". In: *Science of Computer Programming* 78.6, pp. 747–761.

Boissier, O., R Bordini, J. F. Hübner, and A. Ricci (2019). "Dimensions in programming mult-agent systems". In: *The Knowledge Engineering Review* 34, pp. 1–28.

Boissier, O., V. Dignum, and E. García (2016). "Frameworks Comparison". In: *Social Coordination Frameworks for Social Technical Systems. Law, Governance and Technology Series*. Ed. by H Aldewereld, O Boissier, V Dignum, P Noriega, and J Padget. Cham: Springer International Publishing, pp. 213–228.

Boissier, O., J. Hübner, and A. Ricci (2016). "The JaCaMo Framework". In: *Social Coordination Frameworks for Social Technical Systems. Law, Governance and Technology Series*. Ed. by H Aldewereld, O Boissier, V Dignum, P Noriega, and J Padget. Cham: Springer International Publishing, pp. 125–151.

Bonnet-Torrès, O. and C. Tessier (2009). "A Formal Petri Net Based Model for Team Monitoring". In: *Handbook of Research on Multi-Agent Systems: Semantics and Dynamics of Organizational Models*. Ed. by Virginia Dignum. Hershey, New York: IGI Global Information Science Reference, pp. 568–589.

Bradshaw, J. M., A. Uszok, M. Breedy, L. Bunch, T. C. Eskridge, P. J. Feltovich, M. Johnson, J. Lott, and M. Vignati (Jan. 2013). "The KAoS Policy Services Framework". In: *Eighth Cyber Security and Information Intelligence Research Workshop CSIIRW 2013*. Oak Ridge, TN: Oak Ridge National Labs.

Bratman, M., D. Israel, and M.E. Pollack (1988). "Plans and resource-bounded practical reasoning". In: *Computational Intelligence* 4.4, pp. 349–355.

Bresciani, P., P. Giorgini, F. Giunchiglia, J. Mylopoulos, and A. Perini (2004). "Tropos: an agent-oriented software development methodology". In: *Journal of Autonomous Agents and Multi-Agent Systems JAAMAS* 8, pp. 203–236.

Bunch, L., M. Carvalho, J. M. Bradshaw, T. Eskridge, P. J. Feltovich, J. Lott, and A. Uszok (Dec. 2012). "Policy-Based Governance Within Luna: Why We Developed Yet Another Agent Framework". In: *Software Agent Teamwork for the Semantic Web 2012 Workshop in conjunction with WI-IAT 2012*. Macau, China.

Burke, C. S., K. C. Stagl, E. Salas, L. Pierce, and D. Kendall (2006). "Understanding Team Adaptation: A Conceptual Analysis and Model". In: *Journal of Applied Psychology* 91.6, pp. 1189–1207.

Carabelea, C. and O. Boissier (2006). "Coordinating Agents in Organizations Using Social Commitments". In: *Electronic Notes in Theoretical Computer Science* 150, pp. 73–91.

Chesani, F., P. Mello, M. Montali, and P. Torroni (2013). "Representing and monitoring social commitments using the event calculus". In: *Autonomous Agents and Multi-Agent Systems, JAAMAS* 27, pp. 85–130.

Chistoffersen, K. and D. D. Woods (2002). "How to Make Automated Systems Team Players". In: *Advances in Human Performance and Cognitive Engineering Research*. Ed. by E Salas. Vol. 2. JAI press/Elsevier, pp. 1–12.

Cohen-Hatton, S. R., P. C. Butler, and R. C. Honey (Mar. 2015). "An Investigation of Operational decision-making in Situ: Incident Command in the U K Fire and Rescue Service". In: *Human Factors: The Journal of the Human Factors and Ergonomics Society*.

Comfort, L., K. Ko, and A. Zagorecki (2004). "Coordination in Rapidly Evolving Disaster Response Systems The Role of Information". In: *American Behavioral Scientist* 48.3, pp. 295–313.

Cooke, N. J., J. C. Gorman, C. W. Myers, and J. L. Duran (2013). "Interactive Team Cognition". In: *Cognitive Science* 37, pp. 255–285.

Corkill, D., E. Durfee, V. Lesser, H. Zafar, and C. Zhang (May 2011). "Organizationally Adept Agents". In: *Proceedings of 12th International Workshop on Coordina-*

tion, Organization, Institutions and Norms in Agent Systems COIN, AAMAS 2011. Taipei, Taiwan, pp. 15–30.

Cossentino, M., V. Hilaire, N. Gaud, S. Galland, and A. Koukam (2014). "The ASPECS process". In: *Handbook on Agent-Oriented Design Processes*. Springer, pp. 65–114.

Cossentino, M., V. Hilaire, A. Molesini, and V. Seidita, eds. (2014). *Handbook on Agent-Oriented Design Processes*. Springer-Verlag.

Coutinho, L.R., A.A.F. Brand, J.S. Sichman, and O. Boissier (2008). "Model-Driven Integration of Organizational Models". In: *Agent-Oriented Software Engineering IX. LNCS 5386*. Ed. by M. Luck and J.J. Gomenz-Sanz.

Cox, J. and E. Durfee (2009). "Efficient and distributable methods for solving the multiagent plan coordination problem". In: *Multiagent and Grid Systems - An International Journal* 5, pp. 373–408.

Dalpiaz, F., A. K. Chopra, J. Mylopoulos, and P. Giorgini (2011). "From Intentions to Social Commitments: Adaptation in Multiagent Systems". In: *Knowing, Reasoning, and Acting: Essays in Honour of Hector J Levesque*. Ed. by Gerhard Lakemeyer and Sheila A McIlraith. College Publications.

Dalpiaz, F., J. Dix, and M.B. van Riemskijk, eds. (2014). *Engineering Mult-Agent Systems*. Vol. 8758. Springer.

Dam, H.K. and M. Winikoff (2013). "Towards a next-generation AOSE methodology". In: *Science of Computer Programming* 78.6. Special section: The Programming Languages track at the 26th ACM Symposium on Applied Computing SAC 2011 & Special section on Agent-oriented Design Methods and Programming Techniques for Distributed Computing in Dynamic and Complex Environments, pp. 684–694.

Dam, H.K., T. Zhang, and A. Ghose (2013). "Improving the Reactivity of BDI Agent Programs". In: *PRIMA 2013: Principles and Practice of Multi-Agent Systems*. Ed. by G. Boella, E. Elkind, B. Tony R. Savarimuthu, F. Dignum, and M. K. Purvis. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 85–100.

Dastani, M. (2014). "A Survey of Multi-agent Programming Languages and Frameworks". In: *Agent-Oriented Software Engineering*. Ed. by O. Shehory and A. Sturm. Springer Berlin Heidelberg, pp. 213–233.

Dastani, M., D. Grossi, JJ.C. Meyer, and N. Tinnemeier (2009). "Normative Multi-agent Programs and Their Logics". In: *Knowledge Representation for Agents and Multi-Agent Systems. KRAMAS 2008. Lecture Notes in Computer Science*. Ed. by Meyer J-J.C. and Broersen J. Vol. 5605. Berlin, Heidelberg: Springer.

DeLoach, S. A. (2009). "OMACS: a Framework for Adaptive, Complex Systems". In: *Multi-Agent Systems: Semantics and Dynamics of Organizational Models*. Ed. by V. Dignum. IGI Global: Hershey, PA ISBN: 1-60566-256-9.

— (2014). “O-MaSE: An Extensible Methodology for Multi-agent Systems”. In: *Agent-Oriented Software Engineering*. Ed. by O. Shehory and A. Sturm. Heidelberg: Springer-Verlag.

DeLoach, S. A. and J. C. Garcia-Ojeda (2010). “O-MaSE: A Customizable Approach to Designing and Building Complex, Adaptive Multiagent Systems”. In: *International Journal Agent Oriented Software Engineering* 4.3.

DeLoach, S. A., W. H. Oyenan, and E. T. Matson (2008). “A capabilities-based model for adaptive organizations”. In: *Autonomous Agents and Multi-Agent Systems JAAMAS* 16, pp. 13–56.

DeLoach, S.A. and M. Miller (2010). “A Goal Model for Adaptive Complex Systems”. In: *International Journal of Computational Intelligence: Theory and Practice* 5.2.

Dignum, F. (1999). “Autonomous Agents with Norms”. In: *Artificial Intelligence and Law* 7.1, pp. 69–79.

Dignum, V. (2004). *A Model for Organizational Interaction: based on Agents, founded in Logic*. PhD Thesis, Universiteit Utrecht.

— ed. (2009). *Handbook of Research on Multi-Agent Systems: Semantics and Dynamics of Organizational Models*. Hershey, New York: IGI Global Information Science Reference.

Dignum, V. and F. Dignum (2012). “A logic of agent organizations”. In: *Logic Journal, IGPL* 20(1), pp. 283–316.

Dignum, V., J-J. Meyer, and H. Weigand (2002). “Towards an Organizational Model for Agent Societies Using Contracts”. In: *Proceedings of First International Joint Conference on Autonomous Agents and Multi-agent Systems, AAMAS, Bologna, Italy*.

Dignum, V. and C. Tick (2007). “Agent-based Analysis of Organizations: Formalization and Simulation”. In: *Proceedings of IAT 2007: The 2007 IEEEE/WIC/ACM International Conference on Intelligent Agent Technology*. IEEE, pp. 244–247.

Dimopoulos, Y. and P. Moraitis (2006). “Multi-Agent Coordination and Cooperation through Classical Planning”. In: *Proceedings of the IEEE/WIC/ACM International conference of Intelligent Agent Technology IAT’06*.

Dustdar, S. and F. Li (2011). *Service Engineering*. New York: Springer-Verlag/Wein.

Endsley, M., B. Bolte, and D. Jones (2003). *Designing for Situation Awareness: an approach to user-centered design*. CRC Press.

Endsley, M.R. (1995). “Toward a Theory of Situation Awareness in Dynamic Systems”. In: *The Journal of the Human Factors and Ergonomics Society* 37.1, pp. 32–64.

Engesser, T., T. Bolander, R. Mattmüller, and B. Nebel (2015). "Cooperative Epistemic Multi-Agent Planning With Implicit Coordination". In: *Proceedings of the ICAPS-2015 Workshop on Distributed and Multi-Agent Planning DMAP 2015*.

Entin, E. E. (June 2001). "The Effects of Leader Role and Task Load On Team Performance and Process". In: *Proceedings of the 6th International Command and Control Research and Technology Symposium*. Annapolis, Maryland.

Esparcia, S., E. Argente, J. Vicente, and V. Botti (2014). "GORMAS: a methodological guideline for organizational-oriented open mas". In: *Handbook on agent-oriented design processes*, pp. 173–218.

Evertsz, R., J. Thangarajah, and M. Papasimeon (2017). "The Conceptual Modelling of Dynamic Teams for Autonomous Systems". In: *Proceedings of International Conference on Conceptual Modeling 2017*. Springer, Cham, pp. 311–324.

Fan, X., M. McNeese, B. Sun, T. Hanratty, L. Allender, and J. Yen (2010). "Human-Agent Collaboration for Time Stressed Multi-Context decision-making". In: *IEEE Trans on Systems, Man, and Cybernetics, Part A* 40(2), pp. 306–320.

Fan, X., S. Sun, B. Sun, G. Airy, M. McNeese, and J. Yen (2005). "Collaborative RPD-Enabled Agents Assisting the Three-Block Challenge in Command and Control in Complex and Urban Terrain". In: *Proceedings of 2005 BRIMS Conference behaviour Representation in Modeling and Simulation*. Universal City CA, pp. 113–123.

Ferber, J. and O. Gutknecht (July 1998). "A meta-model for the analysis and design of organizations in multi-agent systems". In: *Proceedings of International Conference on Multi Agent Systems*, pp. 128–135.

Ferber, J., O. Gutknecht, and F. Michel (July 2004). "From Agents to Organizations: An Organizational View of Multi-Agent Systems". In: *Agent-Oriented Software Engineering AOSE IV, 2003*. Vol. 2935. Melbourne, pp. 214–230.

Fornara, N. and T. Balke-Visser (2018). "Modelling Organizations and Institutions in MAS". In: *Journal of Applied Logics-IFCOLOG Journal of logics and their applications* 5.2, pp. 565–590.

Fornara, N. and M. Colombetti (2007). "Agent communication and artificial institutions". In: *Autonomous Agents and Mult-Agent Systems JAAMAS*.

— (2009). "Specifying Artificial Institutions in the Event Calculus". In: *Handbook of research on multi-agent systems: Semantics and dynamics of organizational models*. IGI Global, pp. 335–366.

Franco, M.R. and J.S. Sichman (2014). "Comparing and Evaluating Organizational Models: A Multi-Agent Programming Contest Case Study". In: *Pre-proceedings The 17th International Workshop on Coordination, Organisations, Institutions and Norms, AAMAS*.

García, E., E. Argente, and A. Giret (May 2010). "EMFGormas: a CASE tool for developing service-oriented open MAS". In: *Proceedings of 9th International Conference on Autonomous Agents and Multiagent Systems AAMAS 2010*.

García, E., S. Miles, M. Luck, and A Giret (2015). "Evaluating how agent methodologies support the specification of the normative environment through the development process". In: *Journal of Autonomous Agents and Multi-Agent Systems JAAMAS* 29, pp. 1041–1060.

García, E., S. Valero, and A. Giret (2016). "ROMAS-Magentix2". In: *Social Coordination Frameworks for Social Technical Systems. Law, Governance and Technology Series*. Ed. by H Alderwereld, O Boissier, Dignum V, P Noriega, and J Padget. Vol. 30. Springer.

Gâteau, B., D. Khadraoui, E. Dubois, and O. Boissier (2005). "MOISE*Inst*: An Organizational Model for Specifying Rights and Duties of Autonomous Agents". In: *Proceedings of Third European Workshop on Multi-Agent Systems EUMAS 2005*. Elsevier Science B V, pp. 484–485.

Gaudou, B., A. Herzig, and D. Longin (2006). "Grounding and the expression of belief". In: *KR*, pp. 221–229.

Gervits, G., T. Fong, D. Thurston, Q. Pham, R. Thielstrom, and M. Scheutz (2020). "Toward Genuine Robot Teammates: Improving Human-Robot Team Performance Using Robot Shared Mental Models". In: *Proceedings of the 19th International conference on Autonomous Agents and Multi-Agent Systems AAMAS 2020*. Ed. by B. An, N. York-Smith, A. El Fallah Seghrouchni, and G. Sukthankar.

Gorman, J.C., N.J. Cooke, and J.L. Winner (2006). "Measuring team situation awareness in decentralized command and control environments". In: *Ergonomics* 49.12-13, pp. 1312–1325.

Grossi, D. and F. Dignum (2009). "Structural Aspects of Organizations". In: *Handbook of Research on Multi-Agent Systems: Semantics and Dynamics of Organizational Models*. IGI Global, pp. 190–219.

Grosz, B. and L. Hunsberger (2006). "The Dynamics of Intention in Collaborative Activity". In: *Cognitive Systems Research* 7(2-3), pp. 259–272.

Grosz, B. and S. Kraus (1999). "The evolution of SharedPlans". In: *Foundations and Theories of Rational Agency*, pp. 227–262.

Grosz, B.J. and C.L. Sidner (1990). "Plans for discourse". In: *Intentions in Communication*. MIT Press, pp. 417–444.

Hannoun, M., O. Boissier, J. Sichman, and C. Sayettat (2000). "MOISE: An organizational model for multi-agent systems". In: *Advances in Artificial Intelligence*, pp. 156–165.

Harbers, M., J. M. Bradshaw, M. Johnson, P. Feltovich, K. l. van den Bosch, and J-J. Meyer (2011). "Explanation and Coordination in Human-Agent Teams: A Study in the BW4T Testbed". In: *Proceedings of the 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology - Volume 03*. WI-IAT '11. Washington, DC, USA: IEEE Computer Society, pp. 17–20. ISBN: 978-0-7695-4513-4.

Harbers, M., B. Van Reimsdijk, and C. Jonker (2012). "Measuring Sharedness of Mental Models and its relation to Team Performance". In: *Proceedings 14th International Workshop on Coordination, Organisations, Institutions and Norms COIN*, pp. 106–120.

Hendriks, K. V. and J. Dix (2014). "GOAL: A Multi-agent Programming Language Applied to an Exploration Game". In: *Agent-Oriented Software Engineering*. Ed. by O. Shehory and A. Sturm. Heidelberg: Springer-Verlag.

Hevner, A (2007). "A three cycle view of design science research". In: *Scandinavian Journal of Information Systems* 19.2, pp. 87–92.

Hevner, A., T. M. Salvatore, P. Jinsoo, and R. Sudha (2004). "Design science in information systems research". In: *MIS quarterly* 28.1, pp. 75–105.

Hindriks, K. V. (2009). "Programming rational agents in goal". In: *Multi-Agent Programming:* pp. 119–157.

— (Mar. 2014). *Goal Programming Guide*. URL: `http://mmi%20tudelft%20nl/trac/goal/attachment/wiki/WikiStart/Guide%20pdf`.

Hollan, J., E. Hutchins, and D. Kirsh (June 2000). "Distributed Cognition: Toward a New Foundation for Human-Computer Interaction Research". In: *ACM Transactions on Computer-Human Interaction* 7.2, pp. 174–196.

Horling, B. and V. Lesser (2005). "A survey of multi-agent organizational paradigms". In: *The Knowledge Engineering Review* 19.4, pp. 281–316.

Hübner, J., O. Boissier, and R. H. Bordini (2009). "Normative Programming for Organisation Management Infrastructures". In: *LNAI* 6069. Ed. by J Padget, pp. 114–129.

— (2010). "From Organisation Specification to Normative Programming in Multi-Agent Organisations". In: *International Workshop on Computational Logic in Multi-Agent Systems*. Vol. 6245. Springer, pp. 117–134.

Hübner, J., R. Kitio, and A. Ricci (2010). "Instrumenting multi-agent organisations with organisational artifacts and agents 'Giving the organisational power back to the agents'". In: *Autonomous Agents and Multi-Agent Systems, JAAMAS* 20, pp. 369–400.

Hübner, J., J. Sichman, and O. Boissier (2002). "A Model for the Structural, Functional, and Deontic Specification of Organizations in Multiagent Systems". English. In: *Advances in Artificial Intelligence*. Ed. by G. Bittencourt and G. L. Ramalho. Vol. 2507. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 118–128.

Hübner, J.F., J.S. Sichman, and O. Boissier (2007). "Developing organised mult-agent systems using the MOISE+ model: Programming issues at the system and agent levels". In: *Agent Oriented Software Engineering* 1, pp. 370–395.

Hunsberger, L. (1999). "Making SharedPlans More Concise and Easier to Reason About". In: *Agent Theories, Architectures and Languages (ATAL-98), Lecture Notes in Artificial Intelligence* 1555.

Hutchins, E. (1991). "Organizing work by adaption". In: *Organization Science* 2.1, pp. 14–39.

— (2000). "Distributed Cognition. IESBS distributed cognition". In: *International Encyclopedia of the Social and Behavioral Sciences (IESBS)*.

Hutchins, S., A. Bordetsky, Kendall T., and E. Bourakov (2007). "Empirical Assessment of a Model of Team Collaboration. 12th ICCRTS, Adapting C2 to the 21st Century". In: *Proceedings of 12th ICCRTS International Command and Control Research and Technology Symposium: Adapting C2 to the 21st Century*.

Jennings, N. R. (1993). "Commitments and Conventions: The Foundation of Coordination in Multi-Agent Systems". In: *The Knowledge Engineering Review* 8.3, pp. 223–250.

Jensen, A. S., H. Alderwereld, and V. Dignum (2013). "Dimensions of Organizational Coordination". In: *Proceedings of the 25th Benelux Conference on Artificial Intelligence*. Ed. by K. Hindriks, M. de Weerdt, B. van Riemsdijk, and M. Warnier, pp. 80–87.

Jensen, A., V. Dignum, and J. Villadesn (2014). "The AORTA Architecture: Integrating Orgnizational Reasoning in *Jason*". In: *Pre-proceedings of EMAS 2014: 2nd Workship on Engineering Mult-Agent Systems*, pp. 112–128.

Jiang, J., V. Dignum, and Y-H. Tan (2011). "An Agent Based Inter-organizational Collaboration Framework: OperA+". In: *Proceedings of Web Intelligence/IAT Workshops*, pp. 21–24.

Johnson, M., C. Jonker, B. van Reimsdijk, P. J Feltovich, and J. M. Bradshaw (2009). "Joint Activity Testbed: Blocks World for Teams(BW4T)". In: *LNAI, ESAW 2009*. Ed. by H. Aldewereld, V. Dignum, and G. Picard. Vol. 5881. Springer-Verlag, pp. 254–256.

Kinny, D, E A Sonenberg, M Ljungberg, G Tidhar, A S Rao, and E Werner (1994). "Planned team activity". In: *Artificial Social Systems. MAAMAW 1992. Lecture Notes in Computer Science (Lecture Notes in Artificial Intelligence)*. Ed. by C. Castelfranchi and E. Werner. Vol. 830. Springer.

Klein, G., P. J. Feltovich, J. M. Bradshaw, and D. D. Woods (2004). "Common ground and coordination in joint activity". In: *Organizational Simulation*. Ed. by W. R. Rouse and K. B. Boff. New York: Wiley, pp. 139–184.

Kremer, R. and R. Flores (2006). "Flexible Conversations Using Social Commitments and a Performatives Hierarchy". In: *Lecture Notes in Artificial Intelligence* 3859. Ed. by F. Dignum, R. van Eijk, and R. Flores, pp. 93–108.

Lesser, V. et al. (2004). "Evolution of the GPGP/TAEMS Domain-Independent Coordination Framework". In: *Autonomous Agents and Multi-Agent Systems, JAAMAS* 9, pp. 87–143.

Leveson, N. (2004). "A new accident model for engineering safer systems". In: *Safety Science* 42, pp. 237–270.

Li, S., W. Sun, and T. Miller (May 2015). "Communication in Human-Agent Teams for Tasks with Joint Action". In: *Pre-Proceedings 20th International workshop on Coordination, Organizations, Instituions and Norms in Agent Systems COIN AAMAS 2015*.

Martinez, E., I. Kwiatkowski, and P. Pasquier (2011). "Towards a model of Social Coherence in Multi-agent Organizations". In: *COIN 2010 International workshops LNAI* 6541, pp. 114–131.

Mendonca, D., T. Jefferson, and J. Harrald (2007). "Emergent Interoperability: Collaborative Adhocracies and Mix and Match Technologies in Emergency Management". In: *Communications of the ACM* 50.3, pp. 45–49.

Mendonca D .and Wallace, W. A. (2007). "A Cognitive Model of Improvisation in Emergency Management". In: *IEEE Systems, Man and Cybernetics: Part A* 37.4, pp. 547–561.

Meneguzzi, F., M.C. Magnaguagno, and M.P. Singh (2018a). "GoCo: Planning Expressive Commitment Protocols". In: *Autonomous Agents and Multi Agent Systems JAAMAS*.

— (2018b). "GoCo: planning expressive commitment protocols". In: *Autonomous Agents and Multi-Agent Systems JAAMAS*. URL: https://doi.org/10.1007/s10458-018-9385-0.

Mentler, T. and M. Herczeg (2015). "Interactive cognitive artifacts for enhancing situation awareness of incident commanders in mass casualty incidents". In: *Journal of Interaction Science* 3.7.

Miller, T., B. Lu, L. Sterling, G. Beydoun, and K. Taveter (2014). "Requirements engineering using the agent paradigm: a case study of an aircraft turnaround simulator". In: *IEEE transactions on Software Engineering* 40.10, pp. 1007–1024.

Mintzberg, H. (1979). *The Structuring of Organisations*. New Jersey, USA: Prentice-Hall.

— (1980). "Structure in 5's: A Synthesis of the Research on Organization Design". In: *Management Science* 26.3, pp. 322–341.

— (1983). *Structure in Fives: Designing Effective Organizations*. New Jersey, USA: Prentice-Hall.

— (1989). "The Structuring of Organizations". In: *Readings in Strategic Management*. Ed. by D. Asch and C. Bowman. London: Palgrave.

Naikar, N. and B. Elix (2016). "Integrated system design: Promoting the capacity of sociotechnical systems for adaptation through extensions of cognitive work analysis". In: *Frontiers in Psychology* 7, p. 962.

Nieves, J.C., J. Padget, W. Vasconcelos, A. Staikopoulos, O. Cliffe, F. Dignum, J. Vàzquez-Salceda, S. Clarke, and C. Reed (2011). "Coordination, Organisation and Model-driven Approaches for Dynamic, Flexible, Robust Software and Services Engineering". In: *Service Engineering European Research Results*. Ed. by S. Dustdar and F. Li.

Ntuen, C. A., O. Balogun, E. Boyle, and A. Turner (2006). "Supporting command and control training functions in the emergency management domain using cognitive systems engineering". In: *Ergonomics* 49.12-13, pp. 1415–1436.

Odell, J., M. Nodine, and R. Levy (2005). "A Metamodel for Agents, Roles and Groups". In: *Proceedings of international workshop on Agent-Oriented Software Engineering (AOSE) V*. Vol. 3382. Berlin: Springer, pp. 78–92.

Odell, J., H. Van Dyke Parunak, and M. Fleischer (2003). "The Role of Roles in Designing Effective Agent Organizations". In: *Software Engineering for Large-Scale Multi-Agent Systems*. Vol. 2603. Springer, Berlin, pp. 27–38.

Ofra, A., B. Grosz, and K. Z. Gajos (July 2016). "Mutual influence potential networks: Enabling information sharing in loosely-coupled extended-duration teamwork". In: *Proceedings of the Twenty-Fifth international joint conference on Artificial Intelligence IJCAI16*. New York, NY, pp. 796–803.

OMG, Object Management Group (2016). *MOF 2.5.1 Specification*. URL: `https://www.omg.org/spec/MOF`.

— (n.d.). *Unified Modelling Language*. URL: `uml.org`.

Omodei, M., J. McLennan, G. Cumming, C. Reynolds, G. Elliott, A. Birch, and A. Wearing (2005). *Why Do Firefighters Sometimes Make Unsafe Decisions? Some Preliminary Findings*.

Padgham, L., J. Thangarajah, and M. Winikoff (2014). "Prometheus Research Directions". In: *Agent-Oriented Software Engineering Reflections of architectures, methodologies, languages and frameworks*. Ed. by O. Shehory and A. Sturm.

Padgham, L. and M. Winikoff (2004). *Developing intelligent agent systems: A practical guide*. Chicheser: John Wiley & Sons.

Panzarasa, P. and N. R. Jennings (2001). "Negotiation and joint commitments in multi-agent systems". In: *Sozionik aktuell* 3, pp. 65–81.

Pavon, J., J.J. Gomez-Sanz, and R. Fuentes (2005). "The INGENIAS methodology and tools". In: *Agent-Oriented Methodologies*. Ed. by B. Henderson-Sellers and P. Giorgini. Idea Group Publishing, pp. 236–276.

Peffers, K., T. Tuunanen, M. Rothenberger, and S. Chatterjee (2007). "A design science research methodology for information systems resarch". In: *Journal of Management Information Systems* 24.3.

Picard, G., J. H́ubner, O. Boissier, and M-P. Gleizes (2009). "Reorganisation and self-organisation in multi-agent systems". In: *Proceedings of 1st international workshop on Organizational Modeling ORGMOD*.

Pinyol, I. and J. Sabater-Mir (2013). "Computational trust and reputation models for open multi-agent systems: a review". In: *Artificial Intelligence Review* 40.1.

Pitt, J., A. Bourazeri, A. Nowak, M. Roszczynska-Kurasinska, A. Rychwalska, I. Rodriguez Santiago, M. Lopez Sanchez, M. Florea, and M Sanduleac (June 2013). "Transforming Big Data into Collective Awareness". In: *Computer* 46.6, pp. 40–45.

Prue, B., M. Voshell, D. Woods, J. Peffer, J. Tittle, and W. Elm (Apr. 2008). "Synchronised Coordination Loops: A Model for Assessing Distributed Teamwork". In: *Proceedings of NATO Research and Technology Organisation (RTO) Human Factors and Medicine Panel (HFM) symposium Adaptability in Coalition Teamwork*, pp. 17 1–17 12.

Quillinan, T. B., H. Aldewerald, F. Dignum, V. Dignum, L. Penserini, and N. Wijngaards (May 2009). "Developing Agent-based Organizational Models for Crisis Management". In: *Proceedings of 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2009)*. Ed. by K. Decker, J. Sichman, C. Sierra, and C. Castelfranchi. Budapest, Hungary.

Rahwan, I., T. Juan, and L. Sterling (2006). "Integrating social modelling and agent interaction through goal-oriented analysis". In: *Computer Systems Science and Engineering* 3, pp. 87–98.

Rao, A. S. and M. P. Georgeff (1991). "Modeling rational agents within a BDI-architecture". In: *Proceedings of Knowledge Representation and Reasoning*. Ed. by R. Fikes and E. Sandewall. San Mateo, CA: Morgan Kaufmann, pp. 473–484.

— (1995). "BDI Agents: From Theory to Practice". In: *Proceedings of the First International Conference of Multi-Agent Systems*. Vol. 95, pp. 312–319.

Ricci, A., M. Viroli, and A. Omicini (2007). "Give Agents their Artifacts: The A&A Approach for Engineering Working Environments in MAS". In: *Proceedings of the sixth International Joint conference on Autonomous Agents and Multi-agents Systems AAMAS*.

Rouse, W. B., J. A. Cannon-Bowsers, and Salas E. (1992). "The Role of Mental Models in Team Performance in Complex Systems". In: *IEEE Transactions on Systems, Man and Cybernetics* 22.6.

Sardina, S., L. De Silva, and L. Padgham (2006). "Hierarchical planning in BDI agent programming languages: A formal approach". In: ed. by P. Stone and G. Weiss.

Sardina, S. and L. Padgham (2011). "A BDI agent programming language with failure handling, declarative goals and planning". In: *Autonomous Agents and Multi-agent Systems, JAAMAS* 23, pp. 18–70.

Scerri, P., D. Pynadath, N. Schurr, A. Farinelli, S. Gandhe, and M. Tambe (2004). "Team Oriented Programming and Proxy Agents: The Next Generation". In: *Programming Multi-Agent Systems: First International Workshop ProMAS, 2003*. Ed. by M. Dastani, J. Dix, and E. Falah-Seghrouchni. Vol. 3067. Springer-Verlag, pp. 131–148.

Schaafstal, A., J. Johnston, and R. Oser (2001). "Training teams for emergency management". In: *Computers in Human Behavior* 17, pp. 615–626.

Schatten, M., J. Ševa, and I. Tomičić (2016). "A roadmap for scalable agent organizations in the Internet of Everything". In: *Journal of Systems and Software* 115, pp. 31–41.

Scheutz, M., S. A. DeLoach, and J. A. Adams (2017). "A Framework for Developing and Using Shared Mental Models in Human-Agent Teams". In: *Journal of Cognitive Engineering and Decision Making* 3, pp. 203–224.

Schneider, M. F., M. E. Miller, and J. M. McGuirl (2019). "Towards a Meta-Model to Specify and Design Human-Agent Teams". In: *proceedings of 20th International Symposium on Aviation Psychology*, pp. 97–98.

Schraagen, J. M. and J. van de Ven (2011). "Human factors aspects of ICT for crisis management". In: *Cognition, Technology and Work*.

Schurr, N., Pratik P., F. Pighin, and M. Tambe (2006). "Using Multiagent Teams to Improve the Training of Incident Commanders". In: *Proceedings of the Industry Track*

of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS.

Siebra, C. and A. Tate (2005). "Integrating Collaboration and Activity-Oriented Planning for Coalition Operations Support". In: *Lecture Notes on Artificial Intelligence Selected papers from Robocup 2005 ALSO in Proceedings of the 9th International Symposium on Robocup 2005* 4020, pp. 561–568.

Sierhuis, M. (2001). *Modeling and simulating work practice : BRAHMS: a multiagent modeling and simulation language for work system analysis and design*. Amsterdam: PhD Thesis, Universiteit van Amsterdam.

Sierhuis, M., W.J. Clancey, and R.J.J van Hoof (2006). "Brahms: A multiagent modelling environment for simulating work processes and practices". In: *International Journal of Simulation and Process Modelling* 3.3, pp. 134–152.

Sierhuis, M., C. Jonker, B. van Riemsdijk, and K Hindriks (2009). "Toward organization aware agent-based simulation". In: *International Journal of Intelligent Control and Systems* 16, pp. 1–17.

Silva, A. R. da (2015). "Model-driven engineering: a survey supported by the unified conceptual model". In: *Computer Languages, Systems and Structures* 43, pp. 139–155.

Sims, M., D. Corkill, and V. Lesser (2008). "Automated organization design for multiagent systems". In: *Autonomous Agents and Mult-Agent Systems, JAAMAS* 16, pp. 151–185.

Singh, M. P. (1999). "Social Policies: An ontology for commitments in multiagent systems: Toward a unification of normative concepts". In: *Artificial Intelligence and Law* 7, pp. 97–113.

— (2012). "Commitments in Multiagent Systems: Some History, Some Confusions, Some Controversies, Some Prospects". In: *The Goals of Cognition Essays in Honor of Cristiano Castelfranchi*. London: College Publications, pp. 1–29.

Singh, R., E. A. Sonenberg, and T. Miller (May 2016). "Communication and Shared Mental Models for Teams Performing Interdependent Tasks". In: *Pre-proceedings 21st International workshop on Coordination, Organizations, Institutions and Norms in Agent Systems COIN+CARE AAMAS*, pp. 3–18.

Smith, W. and J. Dowell (2000). "A case study of co-ordinative decision-making in disaster management". In: *Ergonomics* 43.8, pp. 1153–1166.

— (n.d.). *Inter-agency coordination in disaster management: A cognitive engineering framework*.

So, R. and E. A. Sonenberg (2007). "Situation Awareness as a Form of Meta-level Control". In: *Proceedings of the First International Workshop on Metareasoning in*

Agent-Based Systems at the Sixth International Joint Conference on Autonomous Agents and Multiagent Systems AAMAS. Honolulu, Hawaii.

Spivey, J.M. (2001). "The Z Notation: A reference manual". In: *Series in Computer Science*. Prentice Hall International.

Sterling, L. and K. Taveter (2009). *The Art of Agent-Oriented Modeling*. London, Cambridge: MIT Press.

Stone, P., G. A. Kaminka, S. Kraus, J. S. Rosenscheind, and N. Agmona (2013). "Teaching and leading an ad hoc teammate: Collaboration without pre-coordination". In: *Artificial Intelligence* 203, pp. 35–65.

Sturm, A. and O. Shehory (2014). "The Landscape of Agent-Oriented Methodologies". In: *Agent-Oriented Software Engineering*. Ed. by O. Shehory and A. Sturm. Heidelberg: Springer-Verlag.

Tambe, M. (1997). "Towards Flexible Teamwork". In: *Journal of Artificial Intelligence Research, JAIR* 7, pp. 83–124.

Tapia, D. I., S. Rodríguez, J. Bajo, and M. Chorchado (2009). "FUSION@, A SOA-Based Multi-agent Architecture". In: *International Symposium on Distributed Computing and Artificial Intelligence 2008 DCAI 2008*. Ed. by J.M. Corchado, S. Rodríguez, J. Llinas, and J. M. Molina. Springer, pp. 99–107.

Telang, P. R., F. Meneguzzi, and M. P. Singh (May 2013). "Hierarchical planning about goals and commitments". In: *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2013*. Saint Paul, Minnesota, USA, pp. 877–884.

Thangarajah, J., L. Padgham, and J. Harland (2002). "Representation and Reasoning for Goals in BDI Agents". In: *Proceedings of the Twenty-fifth Australasian Conference on Computer Science - Volume 4*. ACSC '02. Melbourne, Victoria, Australia: Australian Computer Society, Inc., pp. 259–265.

Tidhar, G. (1999). *Organization-Oriented Systems: Theory and Practice*. PhD Thesis, Department of Computer Science, University of Melbourne.

Tidhar, G. and E. A. Sonenberg (Aug. 1999). "Observations on Team-Oriented Mental State Recognition". In: *Proceedings of the IJCAI Workshop on Team Modeling and Plan Recognition*. Stockholm.

Torroni, P., P. Yolum, M. P. Singh, M. Alberti, F. Chesani, M. Gavanelli, E. Lamma, and P. Mello (2009). "Modelling Interactions via Commitments and Expectations". In: *Handbook of Research on Multi-Agent Systems: Semantics and Dynamics of Organizational Models*. Ed. by V Dignum. IGI Global, Hershey, Pennsylvania, pp. 263–284.

Trotter, M.J., P.M. Salmon, and M.G. Lenné (2013). "Improvisation: theory, measures and known influencing factors". In: *Theoretical Issues in Ergonomics Science* 14.5, pp. 475–498.

Tuomela, R. (2006). "Joint Intention, We-Mode and I-Mode". In: *Midwest Studies in Philosophy* 30 (1), pp. 35–58.

Uez, D.M. and J.F. Hübner (2017). "Modeling for openness in MAS". In: *Proceedings of 5th International Workshop on Engineering Multi-Agent Systems, EMAS*.

Valentine, M. A. and A. C. Edmondson (2015). "Team Scaffolds: How Meso-Level Structures Support Role-based Coordination in Temporary Groups". In: *Organization Science* 26.2, pp. 405–422.

Van Diggelen, J., J. Bradshaw, M. Johnson, A. Uszok, and P. Feltovich (2009). "Implementing collective obligations in human-agent teams using KAoS policies". In: *Proceedings of Workshop on Coordination, Organization, Institutions and Norms (COIN), Eighth International Conference on Autonomous Agents and Multi-Agent Systems AAMAS*. Budapest, Hungary.

Van Riemsdijk, M. B., K. Hindriks, and C. Jonker (2009). "Programming organization-aware agents". In: *International Workshop on Engineering Societies in the Agents World*. Springer, pp. 98–112.

Vázquez-Salceda, J., V. Dignum, and F. Dignum (Nov. 2005). "Organizing Multiagent Systems". In: *Autonomous Agents and Multi-Agent Systems JAAMAS* 11.3, pp. 307–360.

Voshell, M., D. D. Woods, B. Prue, and L. Fern (June 2007). "Coordination Loops: A New Unit of Analysis for Distributed Work". In: *Proceedings or the eighth International NDM Conference*. Ed. by K. Mosier and U. Fischer. Pacific grove, CA.

Weigand, H., V. Dignum, J-J. Meyer, and F. Dignum (2003). "Specification by Refinement and Agreement: Designing Agent Interaction Using Landmarks and Contracts". In: *Third International Workshop on Engineering Societies in the Agents World ESAW2002, LNAI*. Ed. by P. Petta, R. Tolksdorf, and F. Zambonelli. Vol. 2577. Springer-Verlag, pp. 257–269.

Weyns, D. (2017). "Software Engineering of Self-Adaptive Systems: An Organised Tour and Future Challenges". In:

"Designing Multi-Agent Systems from Ontology Models" (2019a). In: *Engineering Multi-Agent Systems 6th International Workship EMAS 2018, Stockholm, Sweden, July 14-15, 2018, Revised Selected Papers, LNAI 11375*. Ed. by D. Weyns, V. Mascardi, and A. Ricci. Springer, pp. 76–95.

Weyns, D., V. Mascardi, and A. Ricci (2019b). *Engineering Multi-Agent Systems: 6th International Workshop, EMAS 2018, Stockholm, Sweden, July 14-15, 2018, Revised Selected Papers*. Vol. 11375. Springer.

Woods, D. and M. Branlat (2011). "Basic Patterns in How Adaptive Systems Fail". In: *Resilience Engineering in Practice*. Ed. by E. Hollnagel, J. Pariès, D. D. Woods, and J. Wreathall. Vol. 3.

Wooldridge, M. (2002). *An Introduction to MultiAgent Systems*. Chichester, England: John Wiley & Sons Ltd.

Yen, J., X. Fan, S. Sun, T. Hanratty, and J. Dumer (2006). "Agents with Shared Mental Models for Enhancing Team Decision-Makings". In: *Journal of Decision Support Systems: Special issue on Intelligence and Security Informatics* 41.3, pp. 634–653.

Zachary, W., W. Weiland, D. Scolaro, J. Scolaro, and T. Santarelli (2002). "Instructorless team training using synthetic teammates and instructors". In: *Proceedings of the Human Factors and Ergonomics Society 46th Annual Meeting*. Human Factors and Ergonomics Society, pp. 2035–2038.

Zambonelli, F., M. Jennings, and Wooldridge M. (2003). "Developing Multi Agent Systems The Gaia Methodology". In: *ACM Transactions on Software Engineering and Methodology* 12, No 3, pp. 317–370.

Zhang, Q., E. Durfee, S. Singh, A. Chen, and S. Witwicki (2016). "Commitment Semantics for Sequential Decision Making under Reward Uncertainty". In: *IJCAI*, pp. 3315–3323.

Zhang, Y., S. Sreedharan, and S. Kambhampati (2015). "Capability models and their applications in planning". In: *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems AAMAS*, pp. 1151–1159.

Appendix A

MAS system specification based on OJAzzIC

The following Z specification provides a partial specification of a MAS system based on OJAzzIC for the rescue response scenario described in Sections 2.3.2, 5.8 and 5.10.

$$[AGENTNAME, ORGNAME, ROLENAME,$$
$$CAPABILITYNAME, GOALNAME, TASKNAME,$$
$$POLICY, POLICYNAME, BELIEFS, PLANID$$
$$DOMAIN_ROLES, DOMAIN_BELIEFS]$$
$$AGENTTYPE ::= Medic \mid Officer \mid Bystander$$
$$ROLENAME ::= Medic \mid Officer$$
$$RELATIONSHIP ::= Peer \mid Authority \mid Subordinate$$
$$DEPENDENCY ::= hierarchical$$

$$Scenario \mathrel{\widehat=} Emergency_Response_System \land \forall a : AGENTNAME \mid Agent_Cycle(a)$$

Emergency_Response_System

$org_goal : \quad ORGNAME \pfun GOALNAME$

$sub_goal : \quad GOALNAME \pfun \mathbb{P}\ GOALNAME$

$org_roles : \quad ORGNAME \pfun \mathbb{P}\ ROLENAME$

$org_policies : \quad ORGNAME \pfun \mathbb{P}\ POLICYNAME$

$goal_comprises : \quad GOALNAME \pfun \mathbb{P}\ TASKNAME$

$before : \quad GOALNAME \pfun GOALNAME$

$member_agents : \quad ORGNAME \pfun \mathbb{P}\ AGENTNAME$

$agent_is_member_of : \quad AGENTNAME \pfun \mathbb{P}\ ORGNAME$

$agent_is_capable_of : \quad AGENTNAME \pfun \mathbb{P}\ CAPABILITYNAME$

$role_responsible : \quad GOALNAME \pfun \mathbb{P}\ ROLENAME$

$role_related : \quad ROLENAME \pfun ROLENAME, RELATIONSHIP$

Emergency_Response_System (Continued)

$goal_requires : \quad GOAL \pfun \mathbb{P}\ CAPABILITYNAME$

$role_requires : \quad ROLE \pfun \mathbb{P}\ CAPABILITYNAME$

$task_requires : \quad TASK \pfun CAPABILITYNAME$

$capabilityset : \quad \mathbb{P}\ CAPABILITYNAME$

$goal_role_responsible : \quad GOAL \pfun \mathbb{P}\ ROLE$

$role_achieves : \quad ROLENAME \pfun \mathbb{P}\ GOAL$

$role_is_capable : \quad ROLENAME \pfun \mathbb{P}\ CAPABILITYNAME$

$agent_can_achieve : \quad AGENT \pfun \mathbb{P}\ TASKNAME$

$current_enacting_role : \quad AGENTNAME \pfun ROLE$

$possible_enacting_role : \quad AGENTNAME \pfun \mathbb{P}\ ROLE$

$shared_plan : AGENTNAME \pfun AGENTNAME, GOAL, plan$

$plan : \quad PLANID \pfun \mathbb{P}(OBJECTIVE, ACTION)$

$agent_operation_next : \quad AGENTNAME \pfun ACTION$

$policies_adopted : \quad POLICYNAME \pfun \mathbb{P}\ POLICY$

$policy_objectives : \quad POLICY \pfun GOAL$

$belief_set : \quad ORGNAME \pfun \mathbb{P}\ BELIEFS$

$dependency : \quad ROLENAME \pfun (ROLENAME, DEPENDENCY)$

$considered_goals : \quad AGENTNAME \pfun GOALNAME$

$current_goal : \quad AGENTNAME \mapsto GOALNAME$

Agent

$\mathrm{dom}\ org_goal = \subseteq GOALNAME$

$\mathrm{ran}\ member_agents = \subseteq AGENTNAME$

$\mathrm{dom}\ ROLENAME = \mathrm{dom}\ AGENTTYPE$

$role_responsible(GOALNAME, ROLENAME) =$

$\quad role_is_capable(ROLENAME, capabilityset)$

$\quad \land\ goal_requires(GOALNAME, capabilityset)$

$agent_can_achieve(AGENTNAME, GOAL) =$

$\quad current_enacting_role(AGENTNAME, ROLENAME) \land$

$\quad role_achieves(ROLENAME, GOAL)$

$agent_can_achieve(AGENTNAME, TASKNAME) =$

$\forall\ capability : task_requires(TASK, capability) \mid$

$\quad agent_is_capable_of(AGENTNAME, capability)$

$agent_can_partially_achieve(AGENTNAME, TASKNAME) =$

$\exists\ capability : task_requires(TASKNAME, capability) \mid$

$\quad agent_is_capable_of(AGENTNAME, capability)$

Init

Δ *Emergency_Response_System*

$DEFAULT_AGENTS ::= medic1, medic2, medic3, officer1, officer2, officer3$

$\text{ran } AGENT_NAME = DEFAULT_AGENTS$

$ORGNAME = \{sysOrg, medicOrg, officerOrg\}$

$member_agents = \{ medicOrg \mapsto (medic1, medic2, medic3),$

$officerOrg \mapsto (officer1, officer2, officer3),$

$sysOrg \mapsto (medic1, medic2, medic3, officer1, officer2, officer3), \}$

$agent_is_member_of = \{medic1 \mapsto medicOrg, medic2 \mapsto medicOrg, medic3 \mapsto medicOrg,$

$officer1 \mapsto officerOrg, officer2 \mapsto officerOrg, officer3 \mapsto officerOrg \}$

$org_goal = \{sysOrg \mapsto resolveDisaster, medicOrg \mapsto rescueInjured,$

$officerOrg \mapsto ensureSafetyOnSite\}$

$sub_goal = \{resolveDisaster \mapsto \{ensureSafetyOnSite, rescueInjured\},$

$ensureSafetyOnSite \mapsto \{clearBystanders, findFights, clearAwayFights\},$

$rescueInjured \mapsto \{locateInjured, performRescue\},$

$performRescue \mapsto \{ performComplexRescue, performSimpleRescue\},$

$performComplexRescue \mapsto \{ stretcherRescue(agent1), stretcherRescue(agent2)\} \}$

$goal_comprises\{stretcherRescue \mapsto \{getStretcher, putPatientOnStretcher,$

$carryStretcher, moveToDropZone\}\}$

$goal_comprises\{performSimpleRescue \mapsto \{pickUp, moveToDropZone\}\}$

$org_roles = \{medicOrg \mapsto Medic, officerOrg \mapsto Officer\}$

$\text{dom } org_policies = \emptyset$

$role_related = \{Medic \mapsto (Officer, Authority)\}$

$role_requires = \{Medic \mapsto (rescue, pickup, move)\} \cup \{Officer \mapsto (pickup, clearbystanders)\}$

$goal_requires = \{(locateInjured \mapsto move),$

$(performSimpleRescue \mapsto move, pickup, hold, clearRoom),$

$(performComplexRescue \mapsto move, pickup, hold, carryStretcher,$

$coordinateStretcherCarrying, createSharedPlan), (findFight \mapsto move) \}$

init_policies

---

init_policies

$\forall\, anofficeragent : DEFAULT_AGENTS \mid anofficeragent \in member_agents(officerOrg)$

$Create_a_Policy(create_a_Commitment(amedicagent,\ anofficeragent,\ someobjective,\ someorg)$

$spolicyA,\ requesthelp(amedicagent, anofficeragent, someobjective, someorg)\ \wedge$

$amedicagent \in member_agents(medicOrg),$

$someobjective \in agent_is_capable_of(anofficeragent),\ someorg)$

$\forall\, anagent : DEFAULT_AGENTS \mid anagent \in member_agents(someOrg)$

$\wedge\ \exists\, belief : b \mid (informed(anotheragent), b)$

$Create_a_Policy(create_a_Commitment(anagent,\ anotheragent, inform(anotheragent, b),$

$someorg),\ spolicyB, \wedge\ anotheragent \in member_agents(someOrg),$

$updated(b),\ someorg)$

$\forall\, anorg : ORGNAME$

$Create_a_Policy(create_a_Commitment(agentx,\ otheragents,\ informIntentions(otheragents),$

$anorg),\ spolicyC, agentx \in member_agents(anorg)\ \wedge$

$\exists\ commitment(agentx,\ someotheragent,\ anorg, obj2) \in contract(agentx)),$

$otheragents \in member_agents(anorg),\ anorg)$

$\forall\, anagent : DEFAULT_AGENTS$

$Create_a_Policy(create_a_Commitment(anotheragent,\ anagent,\ someobjective,\ someorg),$

$spolicyD,\ requesthelp(anotheragent,\ anagent,\ someobjective,\ someorg)\ \wedge$

$therole \in current_enacting_role(anagent)\ \wedge$

$therole \in goal_role_responsible(someobjective),$

$anotheragent \in member_agents(someorg)\ \wedge\ anagent \in member_agents(someorg))$

$\forall\, anorg : ORGNAME$

$Create_a_Policy(create_a_Commitment(agentx,\ otheragents,\ informBusy,\ anorg),$

$spolicyE,\ agentx \in member_agents(anorg)\ \wedge$

$\exists\ commitment(agentx,\ someotheragent,\ anotherorg, obj2) \in contract(agentx)),$

$agentx \in member_agents(anotherorg))$

$Create_a_Policy(create_Org(neworg,\ commongoal,\ agent1, agent2),$

$spolicyF, \exists\, agent1, agent2 : current_goal(agent1) = commongoal\ \wedge$

$current_goal(agent2) = commongoal\ \wedge$

$\nexists\ anexistingorg : agent1 \in member_agents(anexistingorg))\ \wedge$

$agent2 \in member_agents(anexistingorg))$

Agent_cycle

$\Delta\, Agent$

$agentname? : \quad AGENTNAME$

---

do

get_perceived_events $\wedge$

process_individual_and_organisational_attitudes $\wedge$

Adopt_Org_Policy $\wedge$

update_intentions

od

process_individual_and_organisational_attitudes

$\Delta\, Agent$

$agentname? : \quad AGENTNAME$

$\exists\ orggoal : org_goal(myorg) \mid isValid(orggoal) \wedge$

$\quad myorg \in agent_is_member_of(agentname?) \wedge$

$\quad agent_can_achieve(agentname?, orggoal) \wedge$

$\quad considered_goals(agentname?)' =$

$\quad\quad considered_goals(agentname?) \cup \{agentname? \mapsto (orggoal)\}$

$\exists\ orggoal : org_goal(myorg) \mid isValid(orggoal) \wedge$

$\quad myorg \in agent_is_member_of(agentname?) \wedge$

$\quad role_responsible(orggoal, arole) \wedge arole \in current_enacting_role(agentname?) \wedge$

$\quad considered_goals(agentname?)' =$

$\quad\quad considered_goals(agentname?) \cup \{agentname? \mapsto (orggoal)\}$

$\exists\ goal : GOALNAME \mid isValid(goal) \wedge agent_can_achieve(agentname?, goal) \wedge$

$\quad considered_goals(agentname?)' =$

$\quad\quad considered_goals(agentname?) \cup \{agentname? \mapsto (goal)\}$

$(considered_goals \neq null \vee my_goals \neq null) \wedge adopt_Goal(agentname?)$

enact_goal

$\Delta\, Agent$

$agentname? : \quad AGENTNAME$

$agent_can_achieve(goal) \land goal_comprises(goal) \mapsto tasks \land performtasks(tasks) \lor$

$agent_can_partially_achieve(agentname?, goal) \land \exists\, a : AGENTNAME \mid$

$\quad a \in member_agents(myorg) \land agent_can_partially_achieve(a, goal) \land$

$\quad Create_Shared_Plan(myorg, goal, agentname?, a) \land Enact_Shared_Plan \lor$

$agent_can_partially_achieve(agentname?, goal) \land \exists\, a : AGENTNAME \mid$

$\quad a \notin member_agents(myorg) \land agent_can_partially_achieve(a, goal) \land$

$\quad Create_Org(neworg, goal, (agentname?, a)) \land$

$\quad Create_Shared_Plan(neworg, goal, agentname?, a) \land Enact_Shared_Plan$

Enact_Shared_Plan

$\Delta\, Agent$

$\forall\, tasks : shared_plan \mid adopt_Goal(task)$

Create_SharedPlan

$\Delta\, Agent$

$orgname? : \quad ORGNAME$

$goalname? : \quad GOALNAME$

$agent1name? : \quad AGENTNAME$

$agent2name? : \quad AGENTNAME$

$plan? : \quad PLANID$

$agent1name? \in AGENTNAME$

$agent2name? \in AGENTNAME$

$orgname? \in ORGNAME$

$orgname? \subseteq agent_is_member_of(agent1name)$

$orgname? \subseteq agent_is_member_of(agent2name)$

$(commitment(agent1name?, agent2name?, orgname?, plan?)) \in contract$

$\exists\, (commitment(agent2name?, agent1name?, orgname?, plan?))$

$shared_plan' = shared_plan \cup plan?$

Adopt_a_Role

$\Delta\, Agent$

$agentname? : \quad AGENTNAME$

$rolename? : \quad ROLENAME$

$orgname? : \quad ORGNAME$

$rolename? \in ROLENAME$

$rolename? \in orgroles(orgname)$

$current_enacting_role' = current_enacting_role \cup rolename?$

adopt_Goal

$\Delta\, Agent$

$agentname? : \quad AGENTNAME$

$agentname? \in AGENTNAME$

$goal? \in considered_goals(agent1name?) \lor$

$goal? \in my_goals$

$adopted_goal = goal$

$\exists\, org : org \in agent_is_member_of(agentname?) \land$

$spolicyC \in org_policies(myorg) \land$

$\forall\, agents : agents \in member_agents(myorg) inform(agents, adopted_goal)$

$\land\ enact_goal(adopted_goal)$

Create_a_Commitment

$\Delta\, Agent$

$agent1name? : \quad AGENTNAME$

$agent2name? : \quad AGENTNAME$

$obj? : \quad PLANID$

$orgname? : \quad ORGNAME$

$agent1name? \in AGENTNAME$

$agent2name? \in AGENTNAME$

$contract' = contract \cup commitment(agent1name?, agent2name?, orgname?, obj?)$

Request_a_Commitment

$\Delta\, Agent$

$agent1name? : \quad AGENTNAME$

$agent2name? : \quad AGENTNAME$

$obj? : \quad PLANID$

$orgname? : \quad ORGNAME$

$agent1name? \in AGENTNAME$

$agent2name? \in AGENTNAME$

$spolicy2 \in org_policies(orgname?)$

$considered_goals(agent1name?)' = considered_goals(agent1name?) \cup$
$\{agent1name? \mapsto (obj?, agent2name?)\}$

isValid

$\Delta\, Emergency_Response_System$

$goalname? : \quad GOALNAME$

$\forall\, c : preconditions(goalname?) \mid c$

Create_org

$\Delta\, Emergency_Response_System$

$orgname? : \quad ORGNAME$

$orgobj? : \quad GOALNAME$

$orgmembers? : \quad AGENTNAME$

$orgname? \notin \text{ran } org_goal$

$org_goal' = org_goal \cup \{orgname? \mapsto orgobj?\}$

$member_agents = orgmembers?$

Add_Agent_to_Org

Δ *Emergency_Response_System*

orgname? : *ORGNAME*

agentname? : *AGNAME*

---

agentname? $\in$ *AGENTNAME*

orgname? $\in$ *ORGNAME*

$member_agents' = member_agents \cup \{orgname? \mapsto agentname?\}$

$agent_is_member_of' = agent_is_member_of \cup \{agentname? \mapsto orgname?\}$

Add_Role_to_Org

Δ *Emergency_Response_System*

rolename? : *ROLENAME*

orgname? : *ORGNAME*

goalname? : *GOALNAME*

---

rolename? $\in$ *ROLENAME*

orgname? $\in$ *ORGNAME*

$org_roles' = org_roles \cup \{orgname? \mapsto rolename?\}$

$role_responsible' = role_responsible \cup \{goalname? \mapsto rolename?\}$

$role_achieves = goalname?$

$role_is_capable = goal_requires(goalname?)$

member_Agent_in_Org

Δ *Emergency_Response_System*

orgname? : *ORGNAME*

agentname? : *AGENTNAME*

---

agentname? $\in$ *AGENTNAME*

orgname? $\in$ *ORGNAME*

$agentname? \in member_agents(orgname?)$

$orgname? \in agent_is_member_of(agentname?)$

Create_a_Policy

$\Delta\, Emergency_Response_System$

$objective? : \quad GOALNAME$

$newpolicyname?$

$preconditions?$

$spconditions?$

$orgname? : \quad ORGNAME$

$\forall\, c : preconditions \mid c$

$\forall\, const : spconditions \mid const$

$member_Agent_in_Org(orgname?, agent1name?)$

$member_Agent_in_Org(orgname?, agent2name?)$

$newpolicy! = (preconditions, spconditions,$

$\quad Create_a_Commitment(agent1name?, agent2name?, orgname?, objective?))$

$policy_objectives(newpolicyname)' = policy_objectives(newpolicyname) \cup newpolicy$

$Add_a_Policy(orgname?,\ newpolicyname?)$

$POLICYNAME' = POLICYNAME \cup newpolicyname?$

$policies_adopted' = policies_adopted \cup \{newpolicyname? \mapsto policy\}$

Add_a_Policy

$\Delta\, Emergency_Response_System$

$orgname? : \quad ORGNAME$

$policyname? : \quad POLICYNAME$

$policy? : \quad POLICY$

$org_policies' = org_policies \cup \{orgname? \mapsto policyname\}$

Adopt_Org_Policy

$\Delta\, Agent$

$agentname? : AGENTNAME$

$agentname? \in AGENTNAME$

$\exists\, myorg : agent_is_member_of(agentname?)$

$\forall\, policy : org_policies(myorg) \mid$

$\quad my_goals' = my_goals \cup policy_objectives(policy)$