

A Model for Organizational Interaction:

Based on Agents, Founded in Logic

Virginia Dignum

A Model for Organizational Interaction

Based on Agents, Founded in Logic

Een Model voor Organisationele Interactie

Gebaseerd op agenten, gefundeerd in logica

(met een samenvatting in het Nederlands)

Proefschrift

ter verkrijging van de graad van doctor aan de Universiteit Utrecht op
gezag van de Rector Magnificus, Prof. Dr. W. H. Gispen, ingevolge het
besluit van het College voor Promoties in het openbaar te verdedigen op
maandag 12 januari 2004 des middags te 4.15 uur

door

Maria Virgínia Ferreira de Almeida Júdice Gamito Dignum

geboren op 2 mei 1964 te Lissabon

Supervisor: Prof. Dr. John-Jules C. Meyer, Utrecht University
Co-supervisors: Dr. Frank Dignum, Utrecht University
Dr. Hans Weigand, University of Tilburg

ISBN 90-393-3568-0

Copyright © 2004, Virginia Dignum

Cover illustration:

© Pablo Picasso, 'Polichinelle with Guitar Before the Stage Curtain' (1919), c/o Beeldrecht Amsterdam 2003.



SIKS Dissertation Series No. 2004-1

The research reported in this thesis has been carried out under the auspices of SIKS, the Dutch Research School for Information and Knowledge Systems.

The research reported in this thesis was financially supported by Achmea Holding N.V.

*'This writing business. Pencils and what-not.
Over-rated, if you ask me. Silly stuff. Nothing in it.'*

A.A. Milne, *Winnie-the-Pooh*

Preface

The work is done, the writing completed. It is now the time to look back on the last four years. Reading the prefaces of other Ph.D. dissertations, it seems to be a regular thing to use this space to describe how difficult, frustrating and time-consuming doing Ph.D. research has been. Alas, I will have to break with this trend. Of course, it was not always easy, and at times things really got bad, like the time when I had to change from my initial research group to the Intelligent Systems group at the Utrecht University. However, even these difficulties were just temporary, as this change was really for the best, as I got to join one of the best research groups around. There were also other times, when I thought things were not really going the right way, and a good end would never come. But, if I look back, the main feeling is that it all actually went very smooth, very quick, and pretty much the way I expected it to go at the beginning. And this all, without ever missing on the holidays, and taking the time to give an helping hand at the school of Martyn and Laura, and doing all kinds of other fun and social activities. And... no work in the evenings and weekends! (most of the time ☺).

Of course, this could not have happened all by myself. Thus, do not be fooled by the name on the cover of this book. A Ph.D. dissertation arises from the work of many people, whom with their thoughts, comments, discussions, related work and other ways have contributed to the work I am fortunate to be able to present here and claim as my own thesis. From all this people which consciously or unconsciously have contributed to this work, there are a few which I want to remember and thank here.

Firstly, of course my supervisor, John-Jules Meyer, who took me in his group, and who has been always available for discussions, comments and further leads. I want to thank him for his enthusiasm and for the many discussions on the logical and other aspects of this research.

My co-supervisor Hans Weigand, who has been around since the very beginning of this research, and has guided and shaped many of the work presented here. Thanks also for all the philosophical discussions and the opening of new horizons.

The members of the reading committee: Catholijn Jonker, Carles Sierra, Liz Sonenberg, Jan Treur and Rineke Verbrugge. I am privileged and thankful for their careful reading and thoughtful comments.

Achmea, for the opportunity it has given me to pursue this research. At Achmea, I must give special thanks to Bart Dik, Tom Leever, Marc Laan, Willem Jan Bremmer,

Mary Groenewoud, Gerard Zwerink, Benno Tijgelaar, Pauline Plasmeijer, and my fellow Ph.D. students Madelon Evers, Martin Groen, Hans van Leijen, Hanneke Koopmans and Erwin van Geenen. Very special thanks to Pieter van Eeden, manager of the Achmea Knowledge Networking group, with whom I shared several projects, trips, meetings, many discussions and regularly a good laugh.

At the Institute for Information and Computing Sciences of the University of Utrecht, I also have many people to thank: Wilke Schram, who promptly arranged a solution for my less than regular cooperation with the Institute, and who since then has always shown his interest in the evolution of my work; Wieke de Vries for the cheerful decoration and musical animation of our shared room during my first time in Utrecht; special thanks to Javier Vazquez-Sálceda for the cover design. Thanks as well to all my roommates Birna van Riemsdijk, Huib Aldewereld, Jurriaan van Digelen, and currently Huub Prüst; Mehdi Dastani, for the many discussions and for the articles that came forth from those; Jan Broersen, for his help with the ‘deadlines’; Richard Starmans from SIKS; and of course all the other colleagues at the Intelligent Systems Group.

Thanks to Ludger van Elst, Andreas Abecker, Davide Grossi, Ismail Ibrahim, Dulce Pulmareja, Tanya Boundarouk, Lai Xu who have directly contributed to different parts of this research; thanks also the many, many others, friends, colleagues, fellow researchers, whose ideas helped shape this research, and have commented, pointed new directions, and basically showed interested in its advance and results; thanks also to the many anonymous reviewers of the several papers where we have presented parts of this research, who gave us good, interesting, useful, important, even though sometimes infuriating, comments, and as such helped form this dissertation.

Special thanks to my parents, who have provided the basis for my development and supported me in my choices.

Of course, Martyn and Laura, the best children a parent can wish for, for always being there, for constantly reminding me of what is really important in life, and also for their willingness and enthusiasm for their role of paraninfen, in spite of the ‘gekke kleren’. Thank you many, many times.

Last, but certainly not least, Frank, who had the most difficult task of all, being both co-supervisor and husband. If I started this preface by saying that I hardly ever worked on my dissertation during evenings and weekends, the same cannot be said of Frank, who took several evenings and holidays to read and comment on my work ☺. There are no words to express all the thanks.

Maarsse, November 2003

Table of Contents

PREFACE	v
TABLE OF CONTENTS	vii
CHAPTER 1: INTRODUCTION	1
1.1 MOTIVATION	2
1.1.1 <i>Open societies</i>	2
1.1.2 <i>Individuals and organizations</i>	4
1.1.3 <i>Knowledge Management Environments</i>	5
1.2 RESEARCH APPROACH.....	6
1.2.1 <i>Research Questions</i>	6
1.2.2 <i>Research Methodology</i>	7
1.2.3 <i>Research Objectives</i>	8
1.3 SCOPE.....	8
1.3.1 <i>Knowledge Management</i>	9
1.3.2 <i>Agent Theory</i>	9
1.3.3 <i>Organization Studies</i>	10
1.4 ACHMEA.....	11
1.5 CASE STUDIES	12
1.5.1 <i>Knowledge Market</i>	12
1.5.2 <i>CareCircle</i>	12
1.5.3 <i>PAM</i>	13
1.6 THESIS STRUCTURE.....	14
CHAPTER 2: BACKGROUND AND RELATED WORK	15
2.1 KNOWLEDGE AND KNOWLEDGE MANAGEMENT.....	15
2.1.1 <i>Characteristics of knowledge</i>	16
2.1.2 <i>Knowledge sharing</i>	17
2.1.3 <i>IT support for KM</i>	18
2.1.3.1 <i>Business Intelligence Systems</i>	18

2.1.3.2 Knowledge-Based Systems.....	19
2.1.3.3 Software Engineering.....	19
2.1.3.4 Information systems.....	19
2.1.3.5 Requirements for KM support systems.....	21
2.2 THE KNOWLEDGE LEVEL IN IS AND KM.....	21
2.2.1 <i>Distributed and Heterogeneous Environment</i>	22
2.2.2 <i>Dealing with complexity in Knowledge Management</i>	23
2.3 THE AGENT PARADIGM.....	24
2.3.1 <i>What are agents?</i>	24
2.3.2 <i>Agent architectures</i>	25
2.3.3 <i>When should agents be used?</i>	27
2.3.4 <i>Agents for Knowledge and Information Sharing</i>	28
2.4 MULTI-AGENT SYSTEMS.....	29
2.4.1 <i>Agent Societies</i>	30
2.4.2 <i>Coordination in MAS</i>	33
2.4.2.1 Closed approaches to coordination.....	34
2.4.2.2 Open approaches to coordination.....	35
2.4.3 <i>Communication</i>	36
2.4.3.1 Communication Protocol.....	37
2.4.3.1.1 Speech Act Theory.....	37
2.4.3.1.2 Agent Communication Languages.....	38
2.4.3.2 Representing and sharing knowledge.....	39
2.4.3.2.1 Content interchange languages.....	39
2.4.3.2.2 Ontologies.....	40
2.4.3.2.3 Context.....	42
2.5 COORDINATION IN ORGANIZATIONAL STUDIES.....	42
2.5.1 <i>Organizational Forms</i>	42
2.5.2 <i>Social Structures</i>	43
2.6 DISCUSSION.....	44
2.7 CONCLUSIONS.....	46
CHAPTER 3: THE OPERA MODEL FOR AGENT SOCIETIES.....	47
3.1 MOTIVATION.....	48
3.2 RELATED WORK.....	50
3.3 OPERA ARCHITECTURE.....	52
3.3.1 <i>Road Map</i>	54
3.3.2 <i>Working example</i>	56
3.4 ORGANIZATIONAL MODEL.....	56
3.4.1 <i>Social Structure</i>	57
3.4.1.1 Roles.....	59
3.4.1.2 Groups.....	62
3.4.1.3 Dependencies between roles.....	63
3.4.2 <i>Interaction structure</i>	65
3.4.2.1 Scene script.....	66
3.4.2.2 Scene transitions.....	69
3.4.2.3 Role evolution relations.....	70
3.4.3 <i>Normative structure</i>	71
3.4.4 <i>Communicative structure</i>	74
3.5 SOCIAL MODEL.....	77

3.5.1	<i>Social contracts</i>	80
3.5.1.1	Setting up social contracts.....	81
3.5.1.2	Ending social contracts	82
3.5.2	<i>Role enacting agents</i>	83
3.5.2.1	Relationship between role expectation and agent behavior.....	83
3.5.2.2	Consistency and compatibility of agents and roles	86
3.6	INTERACTION MODEL.....	89
3.6.1	<i>Interaction contracts</i>	90
3.6.2	<i>Setting up interaction contracts</i>	92
3.7	CONCLUSIONS	94
3.7.1	<i>OperaA Summary</i>	94
3.7.2	<i>Contribution to multi-agent systems</i>	95
CHAPTER 4: LOGIC FOR CONTRACT REPRESENTATION.....		99
4.1	INTRODUCTION.....	100
4.2	RELATED WORK	101
4.3	LOGIC FOR CONTRACT REPRESENTATION.....	102
4.3.1	<i>Syntax of LCR</i>	103
4.3.2	<i>Semantics of LCR</i>	104
4.3.2.1	Controllable and uncontrollable propositions	104
4.3.2.2	Path and State Semantics	105
4.3.3	<i>Representing deontic modalities in LCR</i>	107
4.3.3.1	Obligations with deadlines.....	107
4.3.3.2	Conditional Obligations	108
4.4	CONTRARY-TO-DUTY IMPERATIVES	110
4.4.1	<i>The forward version of the Chisholm paradox</i>	110
4.4.2	<i>The parallel version of the Chisholm paradox</i>	111
4.4.3	<i>The backward version of the Chisholm paradox</i>	112
4.5	CONCLUSIONS	113
CHAPTER 5: FORMAL MODEL FOR OPERA.....		115
5.1	BUILDING BLOCKS OF OPERA	116
5.1.1	<i>Application Domain</i>	116
5.1.1.1	Identifiers in Opera	117
5.1.1.2	Achievement	117
5.1.1.3	Beliefs.....	119
5.2	LOGICAL INTERPRETATION OF THE ORGANIZATIONAL MODEL	119
5.2.1	<i>Social Structure</i>	120
5.2.1.1	Roles	120
5.2.1.2	Groups	121
5.2.1.3	Dependency relations between roles	122
5.2.1.3.1	Power relations.....	123
5.2.1.3.2	Authorization relations	123
5.2.1.3.3	Role dependency relations.....	124
5.2.1.4	Social structure definition	125
5.2.2	<i>Interaction structure</i>	125
5.2.2.1	Landmarks	126
5.2.2.2	Scene scripts	127
5.2.2.3	Scene Transitions.....	129
5.2.2.4	Interaction structure definition	131

5.2.3	<i>Communicative structure</i>	132
5.2.4	<i>A language for communication and contracts: Illocutionary LCR</i> ...	136
5.2.5	<i>Normative expressions</i>	136
5.2.6	<i>Formal representation of Organizational Model</i>	137
5.3	USING CONTRACTS TO MODEL INTERACTION IN OPERA	138
5.3.1	<i>Contracts and normative systems</i>	138
5.3.2	<i>Contract lifecycle</i>	139
5.3.3	<i>Contract Specification</i>	140
5.3.3.1	Generic contracts	140
5.3.3.2	Example	141
5.3.4	<i>Contract Negotiation and Execution</i>	142
5.4	LOGICAL INTERPRETATION OF THE SOCIAL MODEL	144
5.5	LOGICAL INTERPRETATION OF THE INTERACTION MODEL.....	144
5.5.1	<i>Interaction contracts</i>	145
5.6	ON THE VERIFICATION OF OPERA MODELS.....	146
5.7	SYNTAX OF OPERA.....	148
5.8	CONCLUSIONS	151
CHAPTER 6: DESIGNING AGENT SOCIETIES.....		153
6.1	INTRODUCTION	153
6.2	AGENT-ORIENTED SE METHODOLOGIES	154
6.2.1	<i>Related Work</i>	156
6.2.2	<i>Determination of Agent Appropriateness</i>	159
6.3	AGENT SOCIETY FRAMEWORKS.....	160
6.3.1	<i>Market framework</i>	162
6.3.2	<i>Network framework</i>	164
6.3.3	<i>Hierarchy framework</i>	165
6.4	THE OPERA METHODOLOGY	167
6.4.1	<i>Organizational Model design</i>	169
6.4.1.1	Coordination Level	171
6.4.1.1.1	Description of facilitation roles and scripts	172
6.4.1.1.2	Analysis of the application example.....	173
6.4.1.1.3	Summary of Coordination Level	174
6.4.1.2	Environment Level.....	175
6.4.1.2.1	Requirements and use cases	175
6.4.1.2.2	Communication and ontologies	176
6.4.1.2.3	Stakeholders and roles.....	177
6.4.1.2.4	Social Norms.....	179
6.4.1.2.5	Summary Environment Level	180
6.4.1.3	Behavior Level.....	181
6.4.1.3.1	Roles	182
6.4.1.3.2	Interaction Scripts	183
6.4.1.3.3	Interaction Structure	184
6.4.1.3.4	Summary of Behavior Level	185
6.4.2	<i>Social Model design</i>	186
6.4.2.1	Role Negotiation Scenes	187
6.4.2.2	Summary of Social Model design	189
6.4.3	<i>Interaction Model Design</i>	189
6.4.3.1	Interaction Scene Enactment.....	190
6.4.3.2	Summary of Interaction Model design.....	191

6.5 SOCIETY DESIGN UPDATE.....	192
6.6 CONCLUSIONS	193
CHAPTER 7: APPLICATIONS OF OPERA	195
7.1 INTRODUCTION.....	195
7.2 KNOWLEDGE MARKET	196
7.2.1 <i>Background and Motivation</i>	196
7.2.1.1 Development of the knowledge repository	198
7.2.1.2 Evaluation of the knowledge repository.....	199
7.2.2 <i>Knowledge Market: Agent-based knowledge sharing</i>	201
7.2.2.1 Coordination Level	202
7.2.2.2 Environment Level.....	203
7.2.2.3 Behavior Level.....	205
7.2.2.3.1 Social Structure	205
7.2.2.3.2 Interaction Structure.....	209
7.2.2.4 Social Model.....	211
7.2.2.5 Interaction Model.....	212
7.2.3 <i>Implementation of Knowledge Market</i>	213
7.3 AGENT SOCIETIES IN NON-ORGANIZATIONAL SETTINGS	213
7.4 PERSONAL ASSISTANTS FOR KM	216
7.5 DISCUSSION.....	218
CHAPTER 8: CONCLUSIONS.....	221
8.1 DISCUSSION OF RESULTS	221
8.1.1 <i>Autonomy and Organizational Modeling</i>	222
8.1.2 <i>Applicability of agent paradigm for open organizational models</i>	223
8.1.2.1 Formal models	224
8.1.2.2 Contract specification	225
8.1.3 <i>Development of KM support systems</i>	225
8.1.4 <i>Concluding Remarks</i>	226
8.2 FURTHER WORK.....	226
8.2.1 <i>Implementation</i>	226
8.2.2 <i>Theory</i>	227
8.2.2.1 Integration of agent models.....	227
8.2.2.2 Coordination issues.....	227
8.2.2.3 Delegation and responsibility.....	227
8.2.2.4 Ownership.....	228
8.2.2.5 Normative systems.....	228
8.2.3 <i>Practice</i>	229
8.2.3.1 Applying OperA to other areas	230
8.2.3.2 Linking to the environment.....	230
APPENDIX A: DEVELOPMENT OF COMMUNITIES OF PRACTICE .	231
A.1 INTRODUCTION.....	231
A.2 COMMUNITIES OF PRACTICE AT ACHMEA.....	232
A.3 THE SES MODEL FOR COMMUNITY FACILITATION.....	234
A.3.1 <i>Seduce</i>	235
A.3.2 <i>Engage</i>	236
A.3.3 <i>Support</i>	237
A.4 CONCLUSIONS	239

REFERENCES.....	241
SUMMARY	259
SAMENVATTING.....	261
SUMÁRIO	263
CURRICULUM VITAE.....	265
SIKS DISSERTATION SERIES	267

Chapter 1

Introduction

'A journey of a thousand miles must begin with a single step.'
- Lao-Tzu (604 BC – 531 BC), *The Way of Lao-Tzu*

A look at the titles of recent articles about Knowledge Management (KM) in professional and scientific publications shows a recurring link between the concept of KM to such doom words as 'failure', 'pitfall', 'myth', 'hype', 'folly' or 'mistake'. This illustrates an increasing awareness of researchers and practitioners that the current development and application of KM solutions, in many cases, does not really achieve the intended results. The stress on the technological component of KM is often pointed out as the culprit. On the other hand, people often fail to adopt and accommodate to processes and solutions that are thought of 'offline', as the sure ways to guarantee an optimal KM strategy. That is, KM seems to demand cultural, organizational and technological changes that are difficult if not impossible to implement and apply in the daily practice of organizations. Nevertheless, it remains a fact that modern organizations cannot do without one or another form of structured knowledge creation, sharing, and maintenance. Thus, we seem to have reached a deadlock point here.

In this thesis, we will try to go beyond both the hype and the deadlock situation. We propose a model that makes the most of technology to support interaction and collaboration in ways that enrich the organization and take into account individual requirements and motivations. We believe that such a model will provide a stepping stone to progress the KM field.

This chapter is organized as follows. Section 1.1 provides the motivation for the research. In section 1.2 the research approach is presented, including its objectives, research questions and research approach. Section 1.3 describes the scope and contributions of our work to different research areas. In section 1.4 we provide a brief introduction of Achmea (the organizational setting and sponsor of this research). Section 1.5 presents the case studies where our work will be applied. Finally, section 1.6 provides an outline of the following chapters.

1.1 Motivation

Three main observations form the starting point for this research. Firstly, if one main contribution of Internet can be singled out, that will have to be openness. In a time when the whole world is seemingly at your doorstep, everybody is watching everybody else, and manifold partnership possibilities arise, organizations must be able to manage and adapt their processes dynamically and accordingly to the changes and demands of the environment. That is, businesses, and specifically, knowledge processes are increasingly dynamic and unpredictable which makes it difficult to give a complete a priori specification of all the activities and their knowledge needs. Novel modeling primitives and methods are needed that take this into account.

Secondly, as the external environment of organizations is changing, so too is the inner environment. People are demanding more autonomy, and more sense of ownership, shared power and participation. They insist on meaningful employment, which typically means that they feel that their work (including inevitable routine tasks) is part of an overall mission that they find personally meaningful [Harman, Porter, 1997]. Consequently, organizations and their supporting systems must be able to allow for a degree of negotiation and adaptability in order to accommodate individual participation.

Thirdly, in the area of Knowledge Management, solutions must be devised that incorporate the management of knowledge assets with environments that facilitate and encourage interaction between people. KM systems must allow for dynamic classification and distribution of knowledge, and be able to adapt to changing contexts and personal styles.

In summary, changes in society have consequences to organizations and to the way they manage their knowledge and interactions. In order to be able to continually support organizations, the design, implementation and use of information systems must reflect these changes. In this thesis we propose a model to represent organizational interaction that fulfils these requirements. In the next subsections, the three observations above will be further detailed.

1.1.1 Open societies

At the most fundamental, philosophical level, the concept of open society – as proposed by Karl Popper in [Popper, 1945] – is based on the recognition that people act on imperfect knowledge and that no one is in possession of the ultimate truth. In the more narrow sense of open society systems, we are concerned with organizations of autonomous individuals, each with limited resources and knowledge, that inhabit a common space and collaborate within it.

In an increasing number of domains, organizations are working together in transactions, tasks or missions. Therefore, they are forced to co-ordinate and describe their work processes, resulting in a increasing need for transparency. Open environments are highly dynamic and unpredictable, which makes it difficult to give a complete a priori specification of all the activities that need to be performed, what are their knowledge needs, and how they should be ordered. Virtual Enterprises and

Connected Communities are thus common visions of future open commercial and social structures.

Information and communication technologies that are used to support organizations operating in open environments, are faced with new constraints and requirements in order to model and support open societies. Traditional modeling techniques are based on absolute and a priori knowledge of the domain and its requirements, which is not the case in situations, such as open societies, involving heterogeneous components and dynamic environments. In order to be able to support organizations, systems' architectures and modeling methods should reflect the openness of the environment and the heterogeneity of its components.

Recently, the initiative "Universal Information Ecosystem"¹, promoted by the Information Society Technologies program of the European Commission, introduced *information ecosystems* as an emerging view of future distributed software systems, where the populations are (semi-)autonomous heterogeneous software entities [Davidsson, 2001]. Underpinning these visions is the idea of the open agent society: a flexible network of heterogeneous software processes, each individually aware of the opportunities available to them, capable of autonomous decision-making to take advantage of them, and co-operating to meet transient needs and conditions [Pitt et al., 1999].

In our opinion, models for open society support systems, such as the framework we will describe in this thesis, must meet the following requirements:

- **internal autonomy requirement:** interaction and structure of the society must be represented independently from the internal design of participating entities.
- **collaboration autonomy requirement:** activity and interaction in the society must be specified without completely fixing in advance the interaction structures.

The first requirement relates to the fact that since, in theory, an open society allows the participation of multiple, diverse and heterogeneous entities, the number, characteristics and architecture of which are unknown to the society designer, the design of the society cannot be dependent on their design. With respect to the second requirement, fundamentally, a tension exists between the goals of the society designer and the autonomy of the participating entities. On the one hand, the more detail the society designer can use to specify the interactions, the more requirements are possible to check and guarantee at design time. This allows, for example, to ensure the legality of the interactions, or that certain rules are always followed [Weigand et al., 2003]. On the other hand, there are good reasons to allow the agents some degree of freedom, basically to enable their freedom to choose their own way of achieving collaboration, and as such increase flexibility and adaptability. In short, the advantages of collaboration autonomy are:

¹ <http://www.cordis.lu/ist/fetuie.htm>

- **Extensibility:** not everything needs to be, and often cannot be, known beforehand. The society can evolve smoothly when new interactions or ways to perform these interactions are being developed.
- **Flexibility:** specifying a certain interaction structure beforehand often means that design choices are made that are not really necessary. Forestalling these choices gives the agents the possibility to make these choices themselves, geared to their particular situation.
- **Reusability:** not only is it possible to develop new interaction structures within the society, but it is also easier to reuse interaction structures across societies.

It must be clear that the collaboration autonomy (as any autonomy requirement) is always relative. Certain rules or structures must be present, so that the agents can build on top of that. For example, in a market society it must be possible to arrive at a transaction. It is often not important how the transaction is achieved, but it must be clear at some point that a business transaction has been closed.

1.1.2 Individuals and organizations

As work relationships between people and enterprises are shifting from the ‘job-for-life’ paradigm to project-based ‘virtual’ enterprises in which people and organizations become independent contractors, the ability to organize and maintain its business processes, the support of communication and collaboration, and the management of knowledge are issues that are increasingly more important to ensure the survival and sustainable advantage of organizations [MIT manifesto, 1999]. Table 1-1 describes some of the organizational consequences and challenges resulting from current social changes and how those should be reflected by the information systems that support the organization.

Table 1-1: Organizational consequences of social changes

Social Changes	Organizational Consequences	System Requirements
Overall availability of information and easy access to information	- Access/combine information from heterogeneous sources - 'Distil' task-relevant information	- dynamic classification of knowledge
Open environments	- Transparent processes - Combine heterogeneous processes from different organizations	- Interoperability and flexible connections
Mobility	- Motivate people and keep expertise within organization - Efficient training of new people - Create novel forms of collaboration	- Storage and proactive/reactive availability of knowledge
Flat organizations	- Increased and more complex communication channels - Distributed responsibility	- Support for peer-to-peer communication

People and organizations have different perspectives concerning their interaction and goals. Nevertheless, both individuals and organizations share a common objective: *achieve sustainable adaptability and advantage to the environment.*

Organizations are mainly concerned with the management of the flow of knowledge within the organization: from the organization viewpoint, KM should provide methods and techniques to monitor and guide the creation, acquisition,

storage, sharing and application of knowledge. The knowledge flow must relate and support to the strategic goals of the organization.

People are concerned with their personal development, satisfaction and creativity. From a person's viewpoint, KM should provide individuals with the needed domain knowledge, knowledge about organizational goals and processes, and an intelligent and adaptable platform for optimal development and application of their creativity.

Only when organizations and individuals work together to organize the flow of knowledge such that it supports individual development and relates to the strategic objectives of the organization, will KM lead to innovation and, thus, to competitive and sustainable advantage of the organization. Business processes and models will need to reflect this interaction between individual and organizational views and aims. The framework we have developed as result of this Ph.D. research is a proposal to achieve this requirement.

1.1.3 Knowledge Management Environments

Nowadays, several organization are shifting their strategies from operational excellence and supply-chain to client-based management [Traacy, Wiersema, 1997]. A company that is operationally excellent tries to organize its production process as efficiently as possible. This has been the case since the 19th century when Taylor proposed the division of the production process into simple tasks, each performed by a different person. The latest developments on Enterprise Resource Planning and Just In Time production are also part of this strive for operational excellence. Operationally excellent companies usually focus on the product that the company delivers. The focus of management in a supply-chain organization is its products and processes. Each person or group (department, project team) has a well defined function and will need to deal with a specific 'set' of knowledge. Documents and information systems are designed with a specific 'function' in mind: sales, product information, etc.

A client-oriented organization, on the other hand, operates from the perspective of the needs and wishes of people (potential clients) and tries to find innovative ways to fulfil those needs. This will effect its traditional products, services and processes enormously. Knowing your customers and producing those products that have the highest added value for the customers is the most important. One of the consequences of this shift is that the company has to have close contacts with its customer base. Contacts do not only go through the sales department, but customers also have direct access to data concerning the production planning in order to find out the status of their order. In this 'new' organization, the knowledge needs of people and groups are substantially different from the supply chain organization.

That is, there is a need for dynamic, intelligent knowledge management environments where knowledge is maintained, shared and used in a way that is transparent to all users and easily accessible to the organization as a whole. The emphasis of the knowledge management environment is the support of (collaboration between) human users in their knowledge-intensive tasks by providing, maintaining and distributing relevant knowledge. In our opinion, the application of models for

open societies that take into account individual characteristics to knowledge management, will enable the development of systems that support the requirements above.

1.2 Research Approach

In this thesis, we investigate models for organizations that can reflect changing objectives and requirements in an open environment. The agent paradigm has been proposed as modeling tool for open systems [Jennings, Wooldridge, 1998]. At an abstract level, the concept of agents can refer to any autonomous entity participating in an interaction (including people). The bottom line is however, that we are interested in building software systems that support organizational interaction. In this sense, the concept of agent, used throughout this thesis, refers to software agents, and systems are assumed to be computational systems that model and support the organizational system.

In an open environment where agents (either natural or artificial) with different goals and architectures co-exist, a balance must be found between the autonomy of agents, their coordination needs and the environment expectations. Our main assumption is that, in most social settings, agents need to collaborate in order to realize their individual goals. Furthermore, in many cases, societies have global goals or objectives that lay outside the scope of each individual agent participating but which must be achieved by (coordinated) activity of the participating agents. In this case, if agent autonomy is to be taken seriously, Multi-Agent Systems (MAS) design must take into account that agents will not simply take up those society goals, but a process of negotiation and adjustment of both parties is needed.

1.2.1 Research Questions

The considerations above lead to the following high-level research questions:

1. How can organizations and other types of societies be modeled in a way that integrates global aims and requirements with the autonomy of its participants?
2. Is the agent paradigm adequate to model organizational systems in open environments, especially in knowledge management situations?
3. Can such a society model be used to support knowledge acquisition and sharing across heterogeneous sources in a virtual organization in order to support knowledge intensive tasks and processes?

In order to use the agent paradigm to model open systems, our idea is to separate society modeling from the agents' architectures, and design mechanisms to make collaboration agreements explicit. Such agreements can be specified by means of contracts that must have syntactic, semantic and pragmatic meaning. A language to describe contracts must be rich enough to describe knowledge needs and situations, and also be executable. Furthermore, because the model abstracts from the architecture of the agents, it is able to treat both human and software agents in the same way, at the level of society description. This leads to the following sub-questions:

4. How can efficient coordination between agents (both natural as artificial) be achieved without compromising the agents' autonomy?
5. How to define a contract language that will be used by agents to coordinate their activities within a society?

1.2.2 Research Methodology

The starting point for this research was the observation that distributed knowledge intensive environments require different models and techniques than the ones currently provided by traditional Information Systems (IS) and KM frameworks [Noll, 2003], [Staab, Schnuur, 1999], [Bieber et al., 2002]. The possibilities of multi-agent approaches, on the one hand, and the need for formal methods to represent commitments and interaction, on the other hand, form the basis for this Ph.D. research. Moreover, the application of social concepts in multi-agent systems is currently an area of great research activity. The study and analysis of this body of research was of great value for the launching of our work, and one of the starting points for this Ph.D. project.

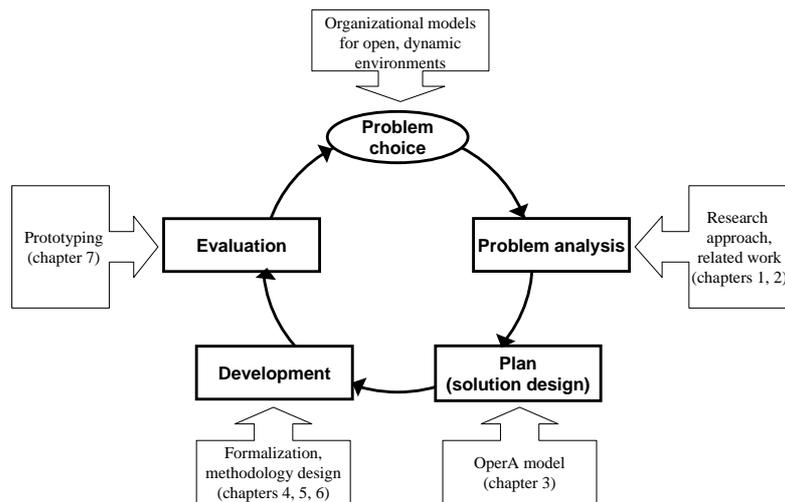


Figure 1-1: Development methodology cycle

The analysis - described in chapter 2 - of literature on existing models and tools for KM systems, on the one hand, and of architectures and proposals for multi-agent systems, on the other hand, has revealed that no models are available that meet the problem we are looking at. This means that an empirical study based on existing applications of agent-based models to organizational environments, is not feasible. Therefore, we have chosen for a design-oriented research approach rather than an empirical study of (artificial) societies and their characteristics and applications. A framework is developed that integrates different views and concepts. A formal model based on temporal deontic logic is provided as semantic foundation for the

framework, and the applicability of the concept is demonstrated in different case studies. The research methodology can be described by the cycle in Figure 1-1.

Departing from a specific problem, in our case the need for organizational models that describe and adapt to open and changing environments, the problem is analyzed, including the research of possible existing solutions, as described in chapter 2. The solution chosen is the development of the OperA model, for the reasons exposed in chapter 3. The development of OperA includes its formalization and the design of a methodology for its application as will be described in chapter 4, 5 and 6. Finally, the solution and its applicability to the original problem will be evaluated through prototyping in a number of case studies, as will be discussed in chapter 7.

1.2.3 Research Objectives

In order to work out the questions above, a number of objectives were formulated for this research:

1. Development of a framework for agent societies (the OperA Model) to legitimate the concept of autonomy between society requirements and agent goals.
2. Development of a formal model to support and give a fundament to the agent society framework.
3. Development of a methodology to guide the process of development of agent societies.
4. Development of prototypes to evaluate the framework and its methodology, and to demonstrate the applicability of the approach to knowledge management domains.

With the development of a framework for agent societies, as is the aim of the first research objective, we provide an answer for the first research question, by means of a concrete way to model organizations that conjugates global structure with the autonomy of the participants. Because of the choice for an agent basis for OperA, and considering that the foremost case study concerns an KM application, the OperA framework is as well an answer to the second research question, which is about the possible advantages of the use of agents for open, KM, environments. The formalization of the model, as aimed by the second research objective, provides a formal semantics for coordination of autonomous entities, based in contracts, and as such answers both sub-research questions described in section 1.2.1. Finally, the development of a practical design methodology and the use of the model in concrete realistic domains, as aimed at in the third and fourth objectives, answers the third research question, which concerns the support of knowledge sharing.

1.3 Scope

This thesis describes multidisciplinary research, that combines aspects from Knowledge Management, Agent Theory, and Organization Studies. In the following subsections, we give an overview of the different fields. Related work and state-of-the-art developments in these fields are presented in chapter 2.

1.3.1 Knowledge Management

The last decade has shown a true hype around KM, resulting in a fast growing number of publications, congresses and consultancy companies on the subject. Reasons for this hype are the claims that controlling knowledge assets leads to performance optimization. Nonaka refers to knowledge as ‘the one sure source of competitive advantage’ [Nonaka, 1991], and Davenport and Prusak state that ‘the aim of knowledge management is to create company value and improve performance’ [Davenport, Prusak, 1998].

Unfortunately, there is no universal definition of Knowledge Management (KM), just as there is no agreement as to what constitutes knowledge in the first place. In fact, KM can be seen as a container concept to which many disciplines contribute [van Engers, 2001]. If an attempt at a definition can be made, then, in the broadest context, ‘*KM is the process through which organizations generate value from their intellectual and knowledge-based assets*’. Most often, generating value from such assets involves sharing them among employees, departments and even with other companies in an effort to devise, use and share best practices. In this sense, knowledge management is not just about managing knowledge sources per se or about managing knowledge workers, but the whole organizational context (strategy, goals, etc.) where knowledge is created, shared and used must be considered.

Note that the definition above says nothing about technology. That is, while KM is often facilitated by IT, technology by itself is not KM. In fact, the role played in KM by technology depends on the approach taken:

- the **stock approach** sees knowledge as a product that can be objectively transferred. In this case, IT plays a predominant role, and KM is realized making use of knowledge systems, databases and other forms of information storage.
- the **flow approach** sees knowledge as a process and argues that its dissemination is dependent on the actors that add subjective value. In this case, IT can be used to facilitate interaction amongst people. The predominant role is taken here by human resources management (HRM).

In our view, knowledge management must integrate both approaches. That is, knowledge management is more than managing knowledge assets or knowledge workers. This is the view taken in this thesis, where we will describe a model that combines both approaches and shows the value of ICT for that combination.

1.3.2 Agent Theory

Agent-based systems are one of the most vibrant and important areas of research and development to have emerged in information technology in the last decades. As the computing landscape moves from a focus on an individual, stand-alone computer system to a situation in which the real power of computers is realized through distributed, open and dynamic systems, agent-based systems can be an answer to new challenges. Currently, agents are being used in an increasingly wide variety of applications, ranging from comparatively small systems as email filters, to large, open, complex, mission critical systems such as air traffic control. Furthermore, the

use of agents as a metaphor for autonomous, intelligent entities offers a way to deal with complex systems that have multiple and distinct components [Luck et al., 2003].

One of the most cited definitions of agent is the following: ‘*An agent is an encapsulated computer system that is situated in some environment and that is capable of flexible, autonomous action in that environment in order to meet its design objectives*’ [Wooldridge, Jennings, 1995]. The key aspect of this definition is **autonomy**, which refers to the principle that agents can operate on their own, without the need for human guidance. An autonomous agent has the control over its own actions and internal state, that is, can decide whether or not to perform a requested action. The definition also situates the agent in a particular environment, which can be sensed and affected by the agent. This indicates **responsive** behavior. Furthermore, the definition implies that agents are problem solving entities, with well-defined boundaries and interfaces, designed to fulfil a specific purpose, that is, agents have particular goals to achieve, and exhibit **flexible** and **pro-active** behavior. Agents are also often capable of **social** behavior, that is, they can **communicate** and **co-operate** with each other and with users. Lastly, for agents to be truly intelligent, it is desirable that they are able to **learn** as they react and interact with their external environment.

Multi-agent systems and concepts play a key role in this thesis both as metaphor as well as a technology for the design and implementation of organizational models. Our main concern is about models for specification and support of interaction. In this sense, we look at MAS theory as a means to model interaction, rather than at the agents in themselves, or at agent architectures. Only some basic features of agents are assumed, such as that agents have goals and means to achieve those goals. For the support of KM, MAS must take into account social and global requirements and therefore adds some constraints to multi-agent systems. We hope to contribute to the development of the MAS area, by proposing a social, conceptual MAS framework, that is both formal and applicable in real-life situations.

1.3.3 Organization Studies

Organizations can be distinguished from other social collectivities, such as groups, families, or mobs, by the fact that organizations are social structures created by individuals to support the collaborative pursuit of specified goals [Scott, 1987], [Parsons, 1956]. As a research field, organization studies comprise an interdisciplinary focus on [Pfeffer, 1997]:

- the effect of social organizations on the behavior and attitudes of individuals within them
- the effects of individual characteristics and actions on the organization, with a special focus on the individual influence (e.g. through leadership)
- the performance, success and survival of organizations
- the effect of the environment on the organization, and vice-versa
- the epistemology and methodology of organizations

The wide variety of subjects and disciplines covered by organizational sciences has given rise to several models and theories on organizations. Traditionally,

organizational studies are the field of business schools which led to an emphasis on economic models. Recently, there has been increasing interest on social views on organizations. One important distinction between both areas, is the extent to which the dimensions of social structure are used to understand behavior. That is, how much emphasis is placed on the individuals or situations and the influence between them and the organization and vice versa.

In this thesis, we are mainly concerned with the first two focus areas above. We hope to contribute to this field of research by developing a framework to model organizations that is both flexible and efficient in handling the relation between individuals and organizations.

1.4 Achmea

This research has been initiated and is performed at Achmea Holding N.V. Achmea originates from the merge between a large number of companies in the Netherlands, mostly active in the insurance and financial services field. The Achmea group offers businesses, institutions and consumers a broad range of insurance, banking and mortgage products, and accompanying services. Achmea also administers pension schemes, provides assistance at home and abroad, and offers health and safety services, absenteeism prevention and reintegration services, and services which encourage a healthy lifestyle. Above all, Achmea is noted for services which 'unburden' its customers. Achmea boasts an important market position in Life (including Pensions), Non-Life and Occupational Health Insurance and is market leader in Health Insurance.

Achmea realizes that a flexible, innovative and personal response to the requirements of customers is in great demand in its field of operation. Since 2001 the central theme of the mission is summarized in the slogan '**Achmea unburdens**'². The realization of this claim has large consequences for the structure and processes of the organization and the transformation movement it originated. In order to improve operating performance, the Achmea units are making increasing use of shared IT systems and exchanging expertise and best practice. On the one hand, processes are being harmonized across the organization, and on the other hand, Achmea strives to achieve larger synergy between people across the different business units. However, the current organizational structure, based on business unit independence, is not always conducive for the realization of synergy. Furthermore, Achmea aims at a position of sustainable adaptability and advantage in its environment, which requires an innovative and flexible approach to customers and their needs and plans, and therefore a better management of knowledge and expertise in the organization [Drucker, 1995]. In order to guide and facilitate KM activities, the Achmea Knowledge Networking (AKN) group, lead by Pieter van Eeden, was formed. AKN is active in applied research activities in collaboration with several Dutch universities, and acts as a de facto organizational research lab.

² In Dutch: Achmea ontzorgt

With this thesis we hope to contribute to the (Knowledge Management) innovation and change within Achmea by developing and applying a model for the design of ICT support systems that takes in account the distributed and dynamic nature of the Achmea organization and its requirements.

1.5 Case Studies

The relevance and applicability of this research is demonstrated through three case studies all of which took place at Achmea, in different business units and environments. The following aspects were investigated:

1. Agent societies to support knowledge sharing: **Knowledge Market**
2. Agent societies supporting interaction in non-organizational settings: **CareCircle**
3. Agents as personal assistant for knowledge workers: **PAM**

All three case studies will be extensively described and discussed in chapter 7. Below, we provide a brief overview of the aims and background of the experiments.

1.5.1 Knowledge Market

The Knowledge Market project is the prime application of the OperA model described in this thesis. This project is an extension to a KM project at Achmea, the KennisNet for the non-life insurance group. KennisNet has the aims to structure, initiate and organize the sharing of knowledge between non-life insurance experts across Achmea by setting up a framework that assures the continuous availability of consistent and up-to-date knowledge.

In the first phase of the project a knowledge repository was developed based on Lotus Notes technology. Evaluation of this system demonstrated that users needed a more personal means of interaction to make them comfortable and willing to exchange knowledge. That is, knowledge owners prefer to share their expertise within a controllable, trusted group under conditions negotiated by themselves for the specific situation and partners. Moreover, people will be more motivated to share their knowledge with others if they feel that they will gain something from the exchange.

The Knowledge Market system extends KennisNet to enable personalized knowledge sharing. An agent society will ensure the preservation of individual needs and perspectives, employing agents to monitor and assist on the exchange. Agents are also used to search the network for suitable partners, to publish and search results in the repository on behalf of their owners, and to monitor news and discussion groups. The Knowledge Market project uses the full potentiality of the OperA model and will demonstrate its applicability for the modeling of KM applications.

1.5.2 CareCircle

The OperA model for agent societies, that we will describe in this thesis, is primarily meant to model interaction within (existing) organizations. However, we are certain that the model is rich enough to model other types of interactions, outside a structured

organizational scope. The CareCircle project aims to demonstrate the applicability of OperA in such non-organizational settings.

CareCircle supports the organization of care services to people in a community. Such services, that often fall outside regular, institutional, services provided by health care organizations, are too expensive for most people, or have long waiting lists. The project is based on the LETS³ (Local Exchange Trading Systems) concept where goods and services are exchanged without recurring to the use of money [Lietaer, 2001], [Boyle, 1999]. The project aims to achieve the following objectives:

1. **Provide extra care services**, currently not covered by professional organizations or for which there are long waiting lists. Examples of such services are: household help during recovery, transport to and from doctor/hospital, babysitting (e.g. to make it possible for parent to visit the doctor), getting medicines /shopping, preparation of meals, etc.
2. **Stimulate community involvement**. Care-givers and care-takers are members of the same community (neighborhood, enterprise, organization). Through CareCircle people will get in contact with each other in ways that extend daily life contacts.
3. **Contribution to ‘Achmea unburdens’**. Through the project Achmea can achieve better answer to client problems through leading edge services, differentiation and sustainable advantage.

An agent society is to be developed that manages the exchange between personal agents representing the members and their services or needs. The system provides registration, matching and conflict resolution capabilities. A point system (like air miles) is used as local currency to regulate exchange.

1.5.3 PAM

In the PAM project, a prototype for an intelligent ‘Customer Care Assistant’ is developed to support decision forming, cross-sell and communication activities of call center agents at the Non-life Call Center of Achmea. PAM aims to improve the way that information is gathered, managed, shared, and presented to people in knowledge-intensive business activities. This project does not use OperA. The reason to include a short description of PAM in this dissertation is that it gives a good idea of the state of the art in the application of agents in KM. Furthermore, the characteristics of the domain are such that the eventual use of OperA to model this situation would result in an improved solution. In this project, PAM, a single agent that manages different sources of information for its owner is created. In particular, PAM:

- allows call center agents to access relevant information wherever it is situated,

³ More information on LETS systems can be found in: <http://www.gmlets.u-net.com/home.html>

- proactively identifies and timely delivers, relevant information which may not have been explicitly asked for (e.g. because the call center agent is unaware of its existence and/ or relevance for the situation on hand),
- informs the call center agent of changes which have been made elsewhere in the business processes which have consequences upon the current decision context, and,
- identifies and informs parties interested in the outcome and results of the decision making activity.

1.6 Thesis structure

This dissertation consists basically of three parts. We will first introduce the OperA model for agent societies and extensively describe its components and features. In the second part, we provide a formal semantics for OperA based on temporal deontic logic. Finally, in the third part, we describe a methodology to develop OperA models and demonstrate the applicability of the framework to the case studies introduced in section 1.5 above. The chapter structure of the thesis is as follows:

- **Chapter 2** provides the reader with background on existing models and theories in the areas of agents, knowledge management, and social organizations.
- **Chapter 3** introduces the OperA model for agent societies.
- **Chapter 4** presents our logic for contract representation, LCR, which is based on deontic temporal logic.
- **Chapter 5** provides the formalization of the OperA model, using the logical concepts introduced in chapter 4.
- **Chapter 6** presents the methodology for the design of OperA agent societies.
- **Chapter 7** describes the case studies and the applicability of the OperA model to the development of real-world applications supporting each case.
- **Chapter 8** presents our conclusions and discusses areas for further research.

Chapter 2

Background and Related Work

'All men have been created to carry forward an ever-advancing civilization.'
- Bahá-u-lláh (1817 – 1892)

No research work ever stands on its own. Research is for a large extent the recognition of the validity of previous ideas and its applicability to different settings. This is certainly true for this work. This chapter gives an overview of basic theories and related research areas, to help gain a better understanding of the concepts and ideas described in the next chapters. It is not our aim to give a complete overview of these subjects, but to present the foundations upon which our research is built. Due to the multidisciplinary character of this research, we provide some background on the different disciplines, which may be an overkill for practitioners in that field, but which is aimed to help researchers from the other fields to achieve a common understanding of the work in the remainder of this thesis.

The chapter starts by describing current theories on Knowledge and Knowledge Management in section 2.1. Section 2.2 looks at knowledge level issues in KM and information systems. In section 2.3 the main aspects of the agent paradigm are discussed. Work on multi-agent systems is presented in section 2.4. Section 2.5 presents coordination approaches in organizational studies. A discussion on the cross-fertilization between the fields of KM, MAS and organizational studies, and its relevance for this dissertation is given in section 2.6. Conclusions are presented in section 2.7.

2.1 Knowledge and Knowledge Management

The aim of knowledge management is to create company value and improve performance [Davenport, Prusak, 1998]. In this sense, knowledge management is not just about managing knowledge sources per se or about managing knowledge workers, but the whole organizational context (strategy, goals, etc.) where knowledge is created, shared and used must be considered. It is only when organizations begin to

link different information and knowledge sources through technological and social connections, and to provide access through these links in meaningful ways, that they gain knowledge that has real business value and can lead to innovation. Knowledge management initiatives should be *embodied in the business environment*, in the sense that they should be designed to implement business strategies and deliver real commercial benefits.

Knowledge management makes sense and delivers real value only when it includes *practical, measurable steps* that deliver concrete results. Knowledge management initiatives may aim to support the formal and informal networks by which knowledge can be identified, retrieved and shared, or they may try to identify, map, codify and capture knowledge so it can be accessed and applied as required. In any case, they should have clear business objectives, be structured in an implementable and measurable way and lead to concrete outcomes [Knownet, 2000].

People are the main generators and consumers of knowledge in an organization, thus, the human factor of knowledge management cannot be ignored. This means that supporting (human) communication must be one of the main aspects of any KM initiative. Furthermore, Knowledge Environments should support people in their knowledge intensive and communication tasks, instead of adding an extra burden to their jobs.

2.1.1 Characteristics of knowledge

The question ‘What is knowledge?’ has been the subject of many philosophical discussions and has as many answers. In logic, to say that an agent knows a sentence either means that he consciously assents to it, or that he immediately sees it to be true when the question is presented. Epistemic logic concerns the notions of knowledge and belief, and is the basis for much work in the area of Artificial Intelligence [Meyer, van de Hoek, 1995]. A classic example of a formal modal logic of knowledge is described in [Hintikka, 1962]. However, formally describing actual, every-day, knowledge is a nearly impossible task: actual knowledge does not seem to obey any logic. Pragmatic notions of knowledge, are mainly used in social and organizational research, and concern the actual use and effect of knowledge. Peter Drucker has said that ‘*Knowledge is information that changes something or somebody either by becoming grounds for actions, or by making an individual [agent] (or an institution) capable of different or more effective action*’ [Drucker, 1989], and West Churchman states that ‘*To conceive of knowledge as a collection of information seem to rob the concept of all its life... Knowledge resides in the user and not in the collection. It is how the user reacts to a collection of information that matters.*’ [Churchman, 1971]. It is not our intention to provide yet again another definition of knowledge. However, it is important to look at some characteristics of knowledge that must be considered in any KM initiative.

- **Persistency.** Knowledge does not go away when given away. That is, in the knowledge flow process, knowledge does not move but spreads. Harlan Cleveland compares knowledge to a sponge [Cleveland, 1997]. “*Information, the raw material for producing knowledge and wisdom, cannot be bottled up for long: it leaks. (...) The competitiveness of an organization depends on their being*

a sponge for inventions, innovations and applications elsewhere.(...) If a company or a country keeps its ideas secret ... it will attract that much less knowledge from others". The aim of KM should then be the management of the saturation of that sponge, that is, what knowledge should be leaking, what knowledge should be absorbed.

- **Non-determinism.** Knowledge processes always involve an actor, who uses (creates, maintains, updates) it in order to perform actions necessary to reach a goal. Knowledge can, and should, be evaluated by the decisions or actions to which it leads [Davenport, Prusak, 1998]. The process of putting it to action refines and extends knowledge. Moreover, knowledge is owner and context sensitive. In this sense, (explicit) knowledge is non-deterministic as no two different agents possessing the 'same' knowledge will act in exactly the same way. Their individual background (experience, skills, etc.) will determine the action taken. If knowledge is the recipe for a cake, then the cook's experience will determine the quality of the cake. (same knowledge, different results) [Gurteen, 1998].
- **Individuality.** Knowledge is personal and cannot be completely duplicated or reproduced (factors such as personality and subjectivity have to be considered). However, the potential for knowledge can be, and should be, shared. We define potential knowledge as the combination of explicit knowledge (which some authors see as information) with the context of application (including insights, lessons learned, applicability and other factors considered important by the generator). The receiver will determine whether and how he will apply that knowledge, making it its own, that is creating his own new knowledge based on the shared potential knowledge.

2.1.2 Knowledge sharing

One of the main objectives of knowledge management is to provide an environment for optimal sharing of knowledge between its users (which can be both people or machines). In the context of this research, agents refer therefore to both human and software agents. Knowledge sharing is basically done in two ways: by articulation and by socialization [Nonaka, 1991]:

- **Socialization:** Sharing of tacit knowledge between agents. In this way knowledge moves from tacit to tacit. Knowledge does not become explicit and cannot easily be used by the organization as a whole.
- **Articulation:** An individual succeeds in formulating the fundamentals of his/her own tacit knowledge in a way that can be communicated to others. This process of making tacit knowledge explicit allows it to be shared within the organization.

Socialization has occurred since the beginning of human history, and is in many ways the preferred way of learning. This is the way the apprentice learns from his/her master. However, in current distributed organizations it is not always possible to approach the 'master' in a direct way. Moreover, because learning occurs directly between individuals, the organization has less control over the learning processes, and dissemination of results occurs infrequently and hazardously. In fact, one could even

venture that efforts towards knowledge management start exactly in an attempt to compensate for the limitations of socialization!

Enterprises, like Achmea, are concerned with the optimal use of knowledge that some of its employees possess. For example, consider the case of an account manager who is expert in the determination of the best insurance for vintage cars, or mortgage packages. If the organization is interested in keeping, using and making available this knowledge across the whole company, several options are available. Sharing it through socialization processes, although usually yielding good results, is often not an option because it is a lengthy process and one can not expect the expert to be able to master thousands of apprentices. Articulation solutions are in such cases the most appropriate.

Consequently, often knowledge management efforts focus on the articulation, or formalization, of knowledge, that is in the conversion of tacit, personal knowledge into explicit, organizational knowledge. Knowledge representation issues are paramount, which leads to a strong dependence of KM on IT. This view presents many advantages, but is not optimal or applicable to all situations. Furthermore, as anyone who has been involved in the development of expert systems and knowledge based systems can tell, the cost of formalizing knowledge is very high and the resulting solution is not always very useful. The rate of usefulness (speed with which knowledge becomes obsolete or useless) and the probability of its reuse determine the benefit of formalizing a 'piece' of knowledge. Parts of the corporate knowledge that need to be processed by computer must be formalized, but other parts that are mainly to be understood by human users can be left informal [Abecker et al, 1998].

2.1.3 IT support for KM

In the last years considerable effort has been made by different computer science disciplines to develop methodologies and applications to support KM both in the area of intelligent information gathering and storage as in the area of task specific support systems. In the following we describe current directions in IT that are increasingly being used to support KM.

2.1.3.1 Business Intelligence Systems

Business Intelligence Systems, such as data warehouses, were designed to support the work of statisticians and analysts. These systems focus mainly on large amounts of structured data, such as in databases. The true value of business intelligence is to help people act on information: to make better decisions, to improve processes, and to seize opportunities [SAS, 1999]. For example, a data warehouse containing information about clients and their insurance policies can be used to discover relations and characteristics previously unknown that then can be used for further business development (e.g. that holders of vintage car insurance have a large probability of also having a recreation boat insurance, or that in a certain region very few households choose for a life insurance mortgage combination).

A restriction of data warehouses is that all data must be stored in the exact same format. Increasingly business activities occur in an environment where people gather

information from various sources, ranging from structured and formal data sets to semi-structured and non-formal documents which call for a distributed (web-based) infrastructure able to support a variety of decision-makers, with different goals and different backgrounds. In a situation where different departments or business units use their own information systems this is not always trivial to achieve and concessions and agreements must be made.

2.1.3.2 Knowledge-Based Systems

From the Knowledge-Based Systems point of view, there are two widespread approaches to build knowledge management systems [Benjamins et al, 1998]:

- **Vertical approaches** deliver task-specific, performance support systems (e.g. expert systems). By incorporating (and formalizing) much application specific knowledge, they provide high value solutions in particular business situations. Such systems are, by nature, restricted to a narrow application area.
- **Horizontal approaches** deliver general frameworks for providing useful corporate information in a wide area of applications. However, in practice this approach essentially amounts to document management or information retrieval systems.

2.1.3.3 Software Engineering

In the area of Software Engineering a concept similar to knowledge management systems, the **Experience Factory** [Basili et al, 1994], was developed to store and reuse documents, designs, code and other artifacts in the Learning Software Organization. As in the case of business intelligence systems, these systems are based on the observation that semi-structured and non-formal documents play a prominent role in an organization knowledge management efforts, and are geared towards a formal and structured representation of knowledge. In recent implementations of Experience Factories, case based reasoning is used to deal with non-formal, unstructured types of knowledge while only very stable, useful and worthy knowledge is codified into formal representations [Althoff et al, 1998].

2.1.3.4 Information systems

Information Systems (IS) can be defined as a set of inter-related components that collect, retrieve, process, store and distribute information to support decision-making, coordination, and control. Information systems help people (managers and workers) analyze problems, visualize complex subjects, and create new products. The role of information systems has, in the last years, shifted from the support of one specific function and set of users, to that of supporting collaboration and business processes in a decentralized, distributed environment [Verharen, 1997].

Information systems are used in KM as tools for storage and sharing of knowledge. This is due to the fact that information is the explicit representation of someone's (or some organization's) knowledge. As such, IS methods and tools are commonly used to support and an large number of IT packages and solutions are available that can contribute to solve the KM problems of organizations. The use of IS in KM is mainly

concerned with the efficient representation and use of explicit knowledge. Furthermore, the amount of information available is increasing at fast pace, a considerable and increasing amount of time is needed to find relevant information, from which to create relevant knowledge. This increases the need for systems that can support workers in specific complex tasks. These include expert systems, decision support systems, workflow management systems and transaction transformation systems.

Examples of information systems designed to support knowledge management efforts within an organization are Document Management Systems (DMS), GroupWare and Intranets and Extranets [Schmid, Stanoevsk-Slabeva, 1998].

Document management systems (DMS) provide database-like storage, management and accessibility of documents. DMS provide access to already available documents without further adding value to them. DMS have applied the concepts of management of structured information to such unstructured information as documents. Especially, the lack of management of the context of the documents in DMS prevents in many situations an effective usage of their content.

GroupWare supports coordination of co-operative work by capturing a repository of (unstructured) pieces of information created by a team during their common work. One well-known example is Lotus/Notes. GroupWare is designed and basically used for informal communication during co-operation. Even though GroupWare has enhanced teamwork, it still is not a sufficient solution for knowledge management since, as DMS, GroupWare basically does not capture the context and there is no added-value summary of the created knowledge. GroupWare tends to make informal knowledge explicit, but generally fails to create or manage coherent team or organizational knowledge.

Organizational Memory Information Systems (OMIS), or Corporate Memories are motivated by the desire to preserve and share the knowledge and experiences that reside in an organization. They represent an effort to coherently integrate know-how dispersed within an organization aimed at enhancing its access and reuse and leading to a shared model of the world. This know-how relates to problem solving expertise in functional disciplines, experiences of human resources, and project experiences in terms of project management issues, design technical issues and lessons learned. OMIS integrate context, documents and structured information. Existing OMIS are, however, usually developed for a special application area. There is no integrated support for the processes necessary for the creation of memory and its dissemination. Practical implementations of Organizational Memories mostly fail, because they are not a natural extension of the knowledge creating process but require additional efforts, which do not provide immediate value to the primary business process, and are often not provided for in the organizational structure [Stein, Zwass, 1995].

The applicability of **Intranet and Extranet** technology to the management of information and knowledge within organizations is increasingly more often seen as the solution for KM systems. Intranets and Extranets apply the basic principles of DMS and OMIS systems, can be enhanced with GroupWare functionality and have

brought the multi-media aspect to knowledge management. They have, however, much of the same drawbacks as the above mentioned systems.

2.1.3.5 Requirements for KM support systems

The above systems have to a great extent improved information availability but have not reached the goal of providing an efficient support for knowledge management. The major weaknesses can be summarized as follows [Dignum, Heimannsfeld, 1999]:

- The concepts and solutions concentrate on explicit knowledge, leaving the fluid, tacit knowledge of humans and human carriers outside of the system. Thus an important, integral part of organizational knowledge is not integrated into the system.
- Knowledge is considered without the context within which it was created. This limits its reusability to employees who have background knowledge about the context.
- The systems are not designed to be an integral part of knowledge creation. In order to extract added value from the stored information, additional tasks have to be performed, which do not provide immediate value and therefore are often omitted, even though they may be of importance in the mid - or long - term.
- The meaning of terms, part of structured or unstructured information, is not explicitly stored in the system. As the meaning of words might change over time, the stored knowledge might be misunderstood.
- Most systems focus on knowledge management within a specific area of application. As a result they do not provide a generic solution and do not provide support for knowledge combination across organizational boundaries as departments or functional areas. Thus existing solutions apply the conventional paper-based knowledge management concepts without adapting them to the potential of the new medium.

2.2 The Knowledge Level in IS and KM

In the above section we discussed the applicability of using IT and in particular Information Systems as a medium for Knowledge Management. We presented several initiatives and approaches, their aims and principal drawbacks. Organizational knowledge is usually embedded in information systems, but in such a way that knowledge is not easily shared through the system. The user is usually the carrier of contextually bound knowledge. Organizational knowledge is not handled formally by the system. The user needs to have the knowledge already in order to be able to use the information system. Management of this implicit knowledge, needed to be able to use information systems, increases complexity in the organization. Furthermore, newer information systems such as Intranets and the Internet also do not simplify organizational behavior: they provide an increasingly complex web of information and knowledge, in a changing, open and dispersed environment.

Information systems are an attempt to concretize concepts, tacit understandings and social process, to provide an objective description of the organization, to

algorithmically compress the elements of the organization into a form in which the maximal informational content is communicated through the shortest possible description. While information systems are developed in order to simplify and fix organizational behavior, their interaction with the organization results in complex behavior, which is emergent and unpredictable.

2.2.1 Distributed and Heterogeneous Environment

Although traditional information systems can provide support to knowledge workers in their daily work, such support is often 'offline', that is, not integrated in the primary processes. Environments are needed that integrate the business process aspects of knowledge work with active support for using and adding to heterogeneous knowledge sources [Staab, Schnurr, 1999]. Moreover, dynamic relationships are also needed between knowledge-intensive business processes and their knowledge sources.

At the symbol level, distributed computing frameworks have been developed to support distributed computing in heterogeneous environments and provide an interface description language and services that allow distributed objects to be defined, located and invoked. The most popular of such distributed object paradigms are OMG's (Object Management Group) Common Object Request Broker Architecture (CORBA), **Microsoft's** Distributed Component Object Model (DCOM) and **JavaSoft's** Java/Remote Method Invocation (Java/RMI) [Burghart, 1998]. Such frameworks encapsulate the heterogeneity of legacy systems and applications within standard, interoperable wrappers. These frameworks are defined and are well suitable to the 'data' level of communication. They presuppose a relatively stable environment and some common grounds of understanding.

In the same way as the distributed object paradigm integrates systems at the data level, at the knowledge level [Newell, 1993] it is necessary to develop a higher level of integration based on the semantics of the problem at hand. At this level, integration can be achieved through **Knowledge Management Environments** which provide uniform access to a diversity of knowledge and information sources of different degree of formality. In order to be able to support the execution of knowledge-intensive tasks, using knowledge from heterogeneous sources, according to diverse user preferences, a common knowledge description must be available, as well as a means to 'translate' domain concepts and relationships between heterogeneous participants. This can be achieved by separating the use of knowledge from the specific characteristics of the knowledge source. These environments should include:

- Loosely connected heterogeneous, multimedia sources.
- Dynamically defined goals.
- Virtual, dynamic links between knowledge needs and knowledge sources.
- Adaptable, intelligent personal assistants, providing support to users.

In our view, organizational memories, as presented in section 2.1.3.4, represent a powerful concept to create and implement Knowledge Management Environments. Ideally, an organizational memory can be seen as a shared, cooperative information system: a space of meanings, terminologies, practices, understandings, cultural norms,

and shared values in an essentially human oriented network within which artificial agents and technologies play an important support role [Gammack, 1998]. This view implies an extension of the concept of information systems, where people and technology are seen as a total cognitive system. In this way, an organizational memory can be seen as a cognitive system, that is '*a complex information processing system that perceives, solves problems, learns, and communicates. Cognitive systems can evolve naturally or be intentionally designed, or both, as in the case of human-computer cognitive systems*' [Webster, 1995]. Such an organizational memory system should actively support users working on knowledge intensive tasks by providing them with all the necessary and useful information for fulfilling that task.

However, in order to present a practical solution for Knowledge Management, the drawbacks of organizational memories must be taken care of. These drawbacks are twofold: methodological and organizational. That is, on the one hand, there is need for a methodology and tools to support and guide the processes necessary for the creation of the memory and its dissemination. On the other hand, to make organizational memories effective, organizational changes are required in order to create and support the view that knowledge creation and sharing are not just a by-product, but an essential part of the organizational effort and strategy.

Furthermore, such systems should be *proactive*, that is, be able to take initiatives in a goal-oriented way as well as *reactive*, that is, respond to user requests or environment changes. The main goal of a knowledge management environment is in our opinion, to provide relevant knowledge to assist the human user in executing knowledge intensive tasks. To be effective, such environments must provide users with relevant knowledge at the right time. By relevant knowledge we mean knowledge which enables users to perform their tasks better with this knowledge than without it.

However, to be accepted by the user, the environment must be able to adapt to the different needs and preferences of users, and integrate naturally with existing work methods, tools and processes. The knowledge management environment relies on an explicit modeling of business processes, such as conventional business process models and workflow management systems.

2.2.2 Dealing with complexity in Knowledge Management

The nature of many processes in today's world is distributed, as is the knowledge involved in those processes. In the real-world, we deal with the increased complexity of the business environment which leads us to delegate both responsibility and authority for certain negotiations and decisions to our representatives or agents, such as real-estate agents, stock brokers, personal shoppers, secretaries, etc. Different systems (either human or automated) are often responsible for different parts of a process: the combination of the different parts defines the effect of the whole. On the other hand, users expect dedicated assistance from the applications they use: the applications should intelligently anticipate, adapt, and actively seek ways to support users [Sycara et al. 1998]. Software agent technology is a joint development from several fields in response to these requirements.

Heterogeneous knowledge environments are open and might change rapidly over time. Because knowledge is embedded in a multitude of different sources, knowledge management systems should be able to handle formal and informal knowledge representations, as well as heterogeneous multimedia knowledge sources. The knowledge assets available in a knowledge management environment are more than ‘traditional’ information systems alone. Such assets include structured and unstructured information, multimedia knowledge representations and links to people (e.g. through knowledge maps or yellow pages – personal directories). Besides using existing knowledge sources, the environment should be able to create (and store) new knowledge based on its observation of the user’s task performance [Leake et al, 1999].

2.3 The Agent Paradigm

The major issues confronting users of increasingly complex knowledge and information systems, as described above, include access and availability of information and knowledge resources, confidence in the veracity and applicability of information provided, and assessment of the trustworthiness of the provider [Klusck, 1999]. Intelligent agents are a new paradigm for developing software applications and are currently the focus of intense interest on the part of several fields of computer science and artificial intelligence [Jennings, Wooldridge, 1998]. Agents have made it possible to support the representation, coordination, and co-operation between heterogeneous processes and their users. A growing number of researchers and organizations are using agents in an increasingly wide variety of applications. Current ‘real world’ agent applications, cover several domains in industry, commerce, health care and entertainment, and range from comparatively small systems such as e-mail filters to large, open, complex, mission critical systems such as air traffic control. It is not our intention to give here a complete overview of the agent field, but we will just describe concepts, characteristics and architectures that are relevant for the remainder of this dissertation.

2.3.1 What are agents?

As already introduced in chapter 1, software agents are commonly defined as [Wooldridge, Jennings, 1995]:

An agent is an encapsulated computer system that is situated in some environment and that is capable of flexible, autonomous action in that environment in order to meet its design objectives.

A few of the notions introduced in this definition are worth further explanation. By ‘encapsulated computer system’ is meant that there is a clear distinction between the agent and its environment. Moreover, the definition implies that there is a well-defined boundary and concrete interface between the agent and its environment. The key aspect of the definition is **autonomy**, which refers to the principle that agents can

operate on their own, without the need for human guidance. An autonomous agent has the control over its own actions and internal state, that is, an agent can decide whether to perform a requested action.⁴ The definition situates an agent in a particular environment, which the agent can sense and effect. This indicates **responsive** behavior. Furthermore, the definition implies that agents are problem solving entities, with well-defined boundaries and interfaces, designed to fulfil a specific purpose, that is, having particular goals to achieve, and exhibiting **flexible** and **pro-active** behavior.

Agents are often regarded as socio-cognitive entities capable of individual social behavior [Weber, 1978]. For an agent to be termed **cognitive** it must be endowed with mental attitudes representing the world and motivating action [Panzarasa et al., 2002], [Wooldridge, 2000]. Further, for a cognitive agent to be deemed **socio-cognitive** it must not only have an intentional stance towards the environment, but also assume other agents to be cognitive entities similarly endowed with mental attitudes for representational and motivational purposes [Dennett, 1987]. Social behavior is characterized by the ability to **communicate** and **co-operate** with others and with users. Lastly, for agents to be truly intelligent, they must be able to **learn** as they react and interact with their external environment [Nwana, Ndumu, 1998]. Considering these characteristics of agents, and their applications, agents can be classified in different categories, [Nwana, Ndumu, 1998], [Franklin, Gasser, 1996]. Agent taxonomies classify different agent types including software agents, life-like agent (like humans and artificial life types), and robots.

2.3.2 Agent architectures

Concerning the implementation of agents, several architectures have been proposed that can be roughly classified into the following types [Wooldridge, 1999], increasingly less abstract:

- **Logic-based agents:** reasoning and decision making are realized through logical deduction [Genesereth, Nilsson, 1987], [Lesperance et al., 1996], [Fischer, 1994].
- **Reactive agents:** in which decision making is implemented as some direct mapping from situation to action [Brooks, 1986], [Maes, 1990].
- **Belief-desire-intention (BDI) agents:** decision making depends on the manipulation of some representation of the beliefs, desires and intentions of the agent [Bratman et al., 1988], [Rao, Georgeff, 1992].
- **Layered agents:** decision making is realized via several software layers, each explicitly reasoning about the environment at different levels of abstraction [Müller et al, 1995], [Fergusson, 1995].

Of the above architectures, we want to pay special attention to the BDI architecture. On the one hand, this architecture has become a *de facto* standard for

⁴ This is a fundamental difference between agents and objects: objects have no control over its own methods, once a publicly accessible method is invoked, the corresponding actions are performed [Wooldridge, 1997].

agent models and is at the basis of namely the FIPA standard, and, on the other hand, it is generic enough to enable the modeling of both natural as artificial agents. Throughout this thesis we will argue that agent models for architectures cannot rely on the internal specifications of the individual agents. Being a generic architecture, BDI provides the best approach to this requirement.

The BDI model has its roots in the philosophical tradition of understanding **practical reasoning** in humans (e.g. [Bratman et al, 1988], [Cohen, Levesque, 1990]). Practical reasoning involves two important processes: deciding *what* goals to achieve (deliberation), and *how* to achieve those goals (means-ends analysis). The process starts by analyzing the options available, which depend on the agent's **beliefs** and **desires**, and deciding which ones to choose. These chosen options became the agent's **intentions**, which then determine its actions. Intentions play an crucial role in the practical reasoning process, as they lead to action. Important aspects of intentions are [Bratman, 1987], [Wooldridge, 2000]:

- *Lead the means-ends reasoning process*: once an intention is formed, the attempt to achieve it involves deciding *how*.
- *Constrain future deliberation*: a rational agent will not entertain options that are inconsistent with its intentions.
- *Persistency*: Agents will not give up their intentions without a good reason. Intentions persist until they are achieved or found impossible to achieve
- *Influence beliefs*: Plans for the future will be based in the belief that the intentions will be achieved.

In summary, agents have a set of beliefs, which are based on their perception of the environment. Beliefs and intentions are used to determine the current options (desires) available to the agent. A deliberation process determines the agent's intentions based on its beliefs, desires and intentions. Intentions are the current focus of the agent: the states it is committed to bring about, and for which the agent will specify a plan on how to reach them. Finally, an action selection function, determines which action to perform based on the current intentions. This process of practical reasoning in a BDI agent is described in Figure 2-1.

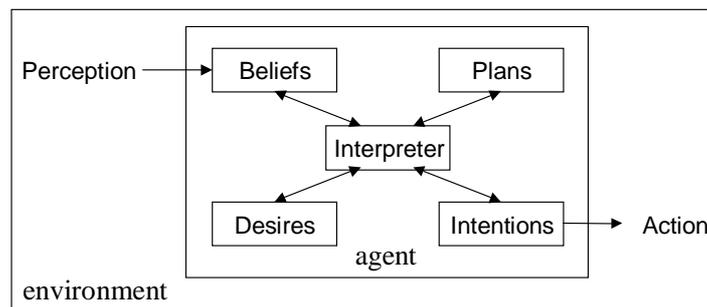


Figure 2-1: The BDI agent model

BDI models have been applied to a number of practical problems including air traffic control, spacecraft handling and telecommunications management and a great

deal of effort has been devoted to their formalization [Rao, Georgeff, 1992]. The best known implementation of the DBI model is the PRS system [Georgeff, Lansky, 1987]. Finally, BDI models have been extended by many researchers, for example to include communication between agents [Haddadi, 1996], [Dignum et al., 2000], or normative behavior [Broersen et al., 2001].

2.3.3 When should agents be used?

Having briefly introduced agents and their characteristics, it is important now to describe in which cases the agent paradigm can or should be used. That is, what do agents have to offer? According to [Jennings, Wooldridge, 1998] the usefulness of any technology should be judged in two directions:

- Its ability of solving new types of problems, and
- Its ability to improve the efficiency of current solutions.

The agent paradigm provides a natural way to view and characterize intelligent and/or reactive systems [Weiss, 1999]. Intelligence and interaction are deeply and inevitably coupled, and multi-agent systems reflect this insight. Multi-agent systems can provide insights and understanding about poorly understood interactions between natural, intelligent beings, as they organize themselves into groups, societies and economies in order to achieve improvement.

Systems that maintain an ongoing interaction with some environment, are inherently quite difficult to design and correctly implement. Process control systems and network management systems are examples of such reactive systems. Applications of the agent paradigm, can be broadly divided in three classes: open systems, complex systems and ubiquitous systems.

- **Open systems** are systems in which the structure of the system is capable of dynamically changing. Their components are not known in advance, can change overtime, and may be highly heterogeneous. An excellent example of an open system is the Internet. Any computer system that must operate in the Internet must be capable of dealing with many and very different organizations and agendas, without constant guidance from users. Such functionality is almost certain to require techniques based on negotiation and co-operation, which lie firmly in the domain of multi-agent systems.
- **Complex systems** relate to particularly complex, large or unpredictable domains. The most powerful tools to deal with complexity in systems are modularity and abstraction. Application of the agent paradigm entails that the overall problem can be partitioned into a number of sub-problems of less complexity, that are easier to handle. This decomposition allows agents to employ the most appropriate paradigm to solve a sub-problem. The notion of an autonomous agent is also a powerful abstraction, in just the same way as data types or objects.
- **Ubiquitous systems** have the goal of enhancing computer use by making many computers available throughout the physical environment, but making them effectively invisible to the user. Ubiquitous systems are roughly the opposite of virtual reality. Where virtual reality puts people inside a computer-generated world, ubiquitous computing forces the computer to live out there in the world

with people [Weiser, 1993]. In ubiquitous systems the need for an equal partnership between the system and its user is paramount. The system has to cooperate with the user to reach their goal. It has been predicted that in the future, delegating to, rather than manipulating computers [Negroponte, 1995] will drive computing. Software applications to deliver such functionality need to be autonomous, pro-active, responsive and adaptive. In other words, such applications need to behave as an intelligent agent. This gives rise to the idea of 'expert assistants', which are agents knowledgeable about both the application and the user.

Agent technology has been successfully applied to several of the above types of systems. However, the fact that a system can be designed as a (multi-)agent system does not mean that an agent-based solution is always the most appropriate one. Other pitfalls to the development of agent-based systems have been discussed in [Wooldridge, Jennings, 1999]. These include political (overselling agents), management (using agents no matter what), conceptual (the risk of the silver bullet), and development (yet another agent architecture) pitfalls. From a software engineering perspective, there are basically four limitations to the use of agents [Jennings, Wooldridge, 1998]:

- *Agent systems have no overall system controller.* An agent-based solution may thus not be appropriate in situations where global constraints have to be maintained.
- *Agents have local perspective.* Agent actions are determined by its own local state. Since in most applications, agents do not maintain complete global knowledge, this may mean that agents make global sub-optimal decisions. One of the aims of multi-agent systems research is to reconcile decision making based on local knowledge with the desire to achieve globally optimal performance [Bond and Gasser, 1988].
- *Trust and delegation limitations.* Both individuals and organizations have to be confident that agents will work on their behalf. The process of learning to trust an agent and to learn how to delegate tasks to an agent takes time.
- *Careful personalization limitations.* Profiles that an agent makes of its user must be comprehensive, accurate, require minimal user input, enforce privacy issues. Furthermore an agent must know its limitations and be trustworthy.

2.3.4 Agents for Knowledge and Information Sharing

Concerning the area of knowledge and information sharing, software agents are often employed as tools to manage loosely coupled information sources, to provide unifying presentation of distributed heterogeneous components and to personalize knowledge presentation and navigation. Agents can either enhance the capability, generality and usefulness of other computer systems (like information agents, which make information sources available to other agents), or be used as an assistant to the user, performing various tasks at the user's request. Possible agent-based services in a KM system are [Klusch, 1999]:

- search for, acquire, analyze, integrate and archive information from multiple heterogeneous sources,
- inform us (or our colleagues) when new information of special interest becomes available,
- negotiate for, purchase and receive information, goods or services,
- explain the relevance, quality and reliability of that information, and
- learn, adapt and evolve to changing conditions.

These services are often specified in terms of the following types of agents:

Cooperative Information Agents (CIA) are agents operating in such an environment. Cooperative Information Agents research and development focuses on accessing multiple, distributed and heterogeneous information sources. Current research also focus on integration and dissemination issues and includes agent negotiation, agent communities, agent mobility and agent collaboration for information discovery [Klusck, Kerschberg, 2000]. CIAs have been used to model systems where users share their preferences, and obtain recommendations for unknown and unseen objects [Delgado, 2000]. Such systems are also called Recommender Systems [Varian, Resnick, 1997] and are used in e-commerce to provide potential clients with such information as '*clients who bought this article also bought...*'

Personal Assistants Agents represent the interests of users within the system, and should adapt to the user's needs. A proactive personal assistant agent will not only perform the tasks given to it by the user, but will also suggest knowledge sources or other resources that are not explicitly requested if they match the user's interests. The personal assistant interacts with a human user to do tasks and learn user preferences [Kearney, 1998]. The most basic personal agents are those that simply automate some actions, like filtering emails. These are already available. The most complex agents are called ubiquitous. These form a dynamic, adaptive, self-organizing global information system.

2.4 Multi-agent systems

Multi-agent environments extend single-agent architectures with an infrastructure for interaction and communication. Ideally, MAS exhibit the following characteristics [Huhns, Stephens, 1999]:

- Are typically open and have no centralized designer;
- Contain autonomous, heterogeneous and distributed agents, with different 'personalities' (cooperative, selfish, honest, etc.);
- Provide an infrastructure to specify communication and interaction protocols.

Agents in a MAS are expected to coordinate by exchanging services and information, to be able to negotiate and agree on commitments, and to perform other complex social operations. Coordination and communication are therefore extremely important issues of MAS, but not really relevant in the case of single-agent systems. In MAS agents have to be able to find each other, announce their possibilities and

pose questions or requests. Furthermore, MAS infrastructure must provide security services, to ensure that agents do not misbehave.

Several architectures and models for MAS have been proposed that handle coordination in different ways. One of the initial and most widely used architectures is based on mediators. The concept of **mediator** was first introduced by Gio Wiederhold [Wiederhold, 1992] as a way to deal with the integration of knowledge from heterogeneous sources. Mediators are facilitation agents that can provide a number of intermediate information services to other agents. They may suggest collaboration between users with common interests, or provide information about tools and resources available. An example of a MAS infrastructure based on the concept of mediators is RETSINA. RETSINA was implemented based on the idea that agents in the system form a community of peers that engage in peer to peer relations. Coordination should emerge from the relations between agents rather than be imposed by the infrastructure, and as such does not employ centralized control but provides (mediation) services that facilitate the relations between agents [Sycara et al., 2003].

2.4.1 Agent Societies

The term *society* is used in a similar way in agent societies research as in human or ecological societies. The role of any society is to allow its members to coexist in a shared environment and pursue their respective roles in the presence and/or in cooperation with others. Main aspects in the definition of society are *purpose, structure, rules and norms*. Structure is determined by roles, interaction rules and communication language. Rules and norms describe the desirable behavior of members and are established and enforced by institutions that often have a legal standing and thus lend legitimacy and security to members. A further advantage of the organization-oriented view on designing multi agent systems is that it allows for heterogeneity of languages, applications and architectures during implementation.

Organizations can be seen as sets of entities regulated by mechanisms of social order and created by more or less autonomous actors to achieve common goals. Multi-agent systems that model and support organizations should therefore be based on coordination frameworks that mimic the structure of the particular organization and be able to dynamically adapt to changes in organization structure, aims and interactions. The structure of the organization determines important autonomous activities that must be explicitly organized into autonomous entities and relationships in the conceptual model of the agent society [Dignum et al., 2001].

In a business environment, the behavior of the global system and the collective aspects of the domain - such as stability over time, predictability and commitment to overall aims and strategies - must be considered. That is, the concept of desirable social behavior is of utmost importance when multi-agent systems are considered from an organizational point of view. This leads to a rising awareness that multi-agent systems and cyber-societies can best be understood and developed if they are inspired by human social phenomena [Artikis et al, 2001], [Castelfranchi, 2000], [Zambonelli et al., 2001a]. This is, in many ways, a novel concept within agent research, even if sociability has always been considered an important characteristic of agents.

When multi-agent systems are considered from an organizational point of view, the concept of desirable social behavior becomes of utmost importance. That is, from the organizational point of view, the behavior of individual agents in a society should be understood and described in relation to the social structure and overall objectives of the society. Until recently, multi agent systems were mainly viewed from an individualistic perspective, that is, as aggregations of agents that interact with each other, and how an agent can affect the environment or be affected by it [Ferber, Gutknecht, 1998]. This view looks at the behavior of multi-agent systems from the perspective of the agent itself, in terms of how an agent can affect the environment or be affected by it. Throughout this dissertation we will use the term **agent society** to refer to MAS considered from a social perspective.

In an individualistic view of Multi-Agent Systems, agents are individual entities *socially situated* in an environment, that is, their behavior depends on and reacts to the environment, and to other agents on it [Dautenhahn, 2000]. It is not possible to impose requirements and objectives to the global aspects of the system, which is paramount in business environments. However, organization-oriented agent societies require a collectivist view on the relation between agent and environment. That is, agents are considered as being *socially embedded* [Edmonds, 1999]. If an agent is socially embedded it needs to consider not only its own behavior but also the behavior of the system as a whole and how agents in the system influence each other.

Davidsson has proposed a classification for artificial societies based on the following characteristics [Davidsson, 2001]:

- *openness*, describing the possibilities for any agent to join the society,
- *flexibility*, indicating the degree agent behavior is restricted by society rules and norms,
- *stability*, defining the predictability of the consequences of actions, and
- *trustfulness*, specifying the extent to which agent owners may trust the society.

Depending on its purpose, a society needs to support these characteristics in different degrees. In one extreme, we have **open societies** that impose no restrictions on agents joining the society. Popper has defined open societies as systems in a state, far from equilibrium, that shows no tendency towards an increase in disorder [Popper, 1982]. That is, open societies support flexibility and openness very well but lack on stability and trustfulness. The most obvious example of an open society is the WWW. Open agent societies assume that participating agents are designed and developed outside the scope and design of the society itself and therefore the society cannot rely on the embedding of organizational and normative elements in the intentions, desires and beliefs of participating agents but must represent these elements explicitly. These considerations lead to the following requirements for engineering methodologies for open agent societies [Dignum, Dignum, 2001]:

- Agent societies must include formalisms for the description, construction and control of the organizational and normative elements of a society (roles, norms and goals) instead of just the agents' states [Artikis et al, 2001], [Zambonelli et al., 2001a]

- The methodology must provide mechanisms to describe the environment of the society and the interactions between agents and the society, and to formalize the expected outcome of roles in order to verify the overall animation of the society.
- The organizational and normative elements of a society must be explicitly specified since an open society cannot rely on its embedding in the intentions, desires and beliefs of each agent [Dellarocas, 2000], [Ossowski, 1998]
- Methods and tools are needed to verify whether the design of an agent society satisfies its design requirements and objectives [Jonker et al., 2000].
- The methodology should provide building directives concerning the communication capability and ability to conform to the expected role behavior of agents participating in the society.

In **closed societies**, on the other extreme, it is not possible for external agents to join the society. Agents in closed societies are explicitly designed to cooperate towards a common goal and are often implemented together with the society [Zambonelli et al., 2001a]. Closed societies provide strong support for stability and trustfulness properties, but only allow for very little flexibility and openness. The large majority of existing MAS are closed.

[Davidsson, 2001] introduces two new types of agent societies, **semi-open** and **semi-closed**, that combine the flexibility of open agent societies with the stability of closed societies. This balance between flexibility and stability results in systems where trust is achieved by mechanisms that enforce ethical behavior between agents:

- In semi-open societies the access of external agents is explicitly regulated. This allows to decide on the acceptance or not of new members and to monitor which agents are currently in the society. An example of a semi-open society is the Napster system⁵. Semi-open societies slightly limit the openness and flexibility characteristics of open societies, but are able to provide greater stability and trustfulness.
- Semi-closed societies do not allow for the participation of external agents but provide the possibility for external parties to initiate a new agent within the society to act on their behalf. This extends the flexibility and openness of the society, without losing on stability and trustfulness, since participating agents still are designed following the society requirements and the owner of the society still controls the overall architecture of the system. Semi-closed societies are as open as semi-open society but less flexible. This is the approach taken in the ISLANDER platform where external agents are provided with an API as interface to the institution, which regulates and controls all interaction [Esteva et al., 2002b].

⁵ <http://www.napster.com>

2.4.2 Coordination in MAS

Multi-agent systems that are developed to model and support organizations need coordination frameworks that mimic the coordination structures of the particular organization. The organizational structure determines important autonomous activities that must be explicitly organized into autonomous entities and relationships in the conceptual model of the agent society [Dignum et al., 2001]. Furthermore, the multi-agent system must be able to dynamically adapt to changes in organization structure, aims and interactions.

Coordination can be defined as the process of managing dependencies between activities [Malone, Crowston, 1994]. Organizational science and economics have since long researched coordination and organizational structures [Williamson, 1975], [Powell, 1990]. Drawing on disciplines such as sociology and psychology, research in organization theory focuses on how people coordinate their activities in formal organizations. On the other hand, it is also generally recognized that coordination is an important problem inherent to the design and implementation of multi-agent systems [Bond, Gasser, 1998].

The challenge of coordination in MAS has been recognized by many authors and several approaches have been developed and advocated. Such approaches take either a bottom-up (e.g. goal management in which members of the group take control of the definition of their work [Malone, Crowston, 1994]) or a top-down view of coordination (e.g. shared ontologies [Fox, Gruniger, 1998] and the hierarchical assignment of responsibilities used in many human organizations). Coordination is one of the cornerstones of agent societies and is considered an important problem inherent to the design and implementation of MAS [Bond, Gasser, 1988], [Dignum, Dignum, 2001]. However, the implications of coordination models to the architecture and design of agent societies are not often considered. Other examples of coordination theories in MAS are joint-intentions [Cohen, Levesque, 1991], [Dunin-Keplicz, Verbrugge, 2002], shared plans [Grosz, Kraus, 1996] and domain-independent teamwork models [Tambe, 1997].

Behavioral approaches to the design of multi-agent systems are gaining terrain in agent research and several research groups have presented models similar to our proposal. Recent developments recognize that the modeling of interaction in MAS cannot simply rely on the agent's own (communicative) capabilities. Furthermore, organizational engineering of MAS cannot assume that participating agents will act according to the needs and expectations of the system design. Concepts as organizational rules [Zambonelli, 2002], norms and institutions [Esteva et al., 2001] and social structures [Parunak, Odell, 2002] all start from the idea that the effective engineering of MAS needs high-level, agent-independent concepts and abstractions that explicitly define the organization in which agents live [Zambonelli et al., 2001a].

Relating society models to the organizational perception of the domain can facilitate the development of organization-oriented multi-agent systems. This means that the development of agent society models for organizations must be a concerted effort between MAS engineers and domain specialists. A common ground of understanding is therefore needed between MAS engineers and organizational

practitioners. Coordination aspects are relevant both in agent research as in organizational theory. Therefore, we propose to look at coordination as the way to bridge both communities and create an initial common ground for cooperation.

2.4.2.1 Closed approaches to coordination

In distributed Computer Science, coordination languages are a class of programming notations that offer a solution to the problem of specifying and managing the interactions among computing agents. From this point of view, coordination models can be divided into two classes: **control-driven** and **data-driven** [Papadopoulos, Arbab, 1998]. Control-driven models are systems made up of a well-defined number of entities and functions, in which the flow of control and the dependencies between entities need to be regulated. The data-driven model is more suited for open societies where the number of entities and functions is not known a priori and cooperation is an important issue. While the classification of cooperation provided by organizational theory stems from social considerations and transaction costs, this classification is concerned with the way interaction between agents happens.

In Distributed Artificial Intelligence (DAI), coordination approaches are often based on contracting. The most famous example of these is the Contract Net Protocol (CNP) [Smith, 1980] for decentralized task allocation. CNP was designed to handle applications with a natural spatial distribution. It assumes a network of loosely coupled asynchronous nodes (agents), each containing a number of distinct knowledge sources. The agents are interconnected so that each agent can communicate with every other agent by sending messages. Agents can either execute tasks or have tasks that need to be executed. CNP provides a simple language to describe contracts for task execution in messages between agents. Furthermore, matchmaking and monitoring services are available. In short, CNP acts as follows:

- All agents must register with the matchmaker.
- When an agent needs to locate other agents, it must send a request message to the matchmaker describing the requested service.
- Other agents can then make bids.
- Once bids have been received, the request will select one (according to some criteria) and allocate the task to that bidder.
- The bidder can then accept the task.

The CNP protocol assumes that all agents are eager to contribute, and the most appropriate bid is the bid of the agent with the best capability and availability. A more sophisticated version of the CNP is the TRACONET model [Sandholm, Lesser, 1995]. In this model, agents are supposed to be self-interested. This means that contractors have to pay a price for the service performed. Contractors try to minimize the costs by selecting the bidder with the lowest price (all things being equal). Potential subcontractors try to maximize their benefit. If they read an announcement of a contractor that offers a price lower than their minimum price, they will discard it. It is also possible to respond by a counter offer.

Contractual Agent Societies (CAS) apply the concept of contracting to the coordination of MAS, and are inspired by work in the areas of organizational theory,

economy and interaction sociology, which model organizations and social systems after contracts [Dellarocas, 2000]. Crucial to the CAS model is the distinction between mutually trusted agents and mutually untrusted agents. A market place is a set of mutually trusted agents; when an untrusted agent wants to join the market place, it applies at a socialization service that not only plugs in the agent technically, but also makes him agree on a social contract. Social contracts govern the interaction of a member with the society. A social contract is a commitment of an agent to participate in a society (or market place), and includes beliefs, values, objectives, protocols and policies that agents agree to obey in the context of their social relationship. CAS defines a general set of principles for MAS coordination. These principles can be described as follows:

- New agents are admitted through a process or socialization during which the agent negotiates with the society the terms of its membership. As a result the terms of the social contracts of existing members may need to be renegotiated as well
- Members of a CAS may form sub communities in the context of a CAS by negotiating private contracts on a bilateral basis, for example, using CNP or TRACONET.
- The society commits itself to enforce the agent's private contracts. To this end, two special agents are defined: a notary agent, responsible for storing contracts and resolving potential disputes, and a reputation agent, responsible for keeping records of all contracts formed by members of the market place. The society also contains a matchmaker agent that helps registered agents to locate other members.
- A mechanism of social control may be negotiated as part of the social contract, defining deviations from agreed 'normal' behavior and corresponding sanctions. For instance, misbehaving agents can be banned from a society, if this is specified in the social contract.

The above application of contracts are mainly geared to the modeling of market places. However, contracts have also been used to model the interaction in Information Systems, in terms of Cooperative Information Agents [Verharen, 97]. This work assumes that agent's behavior is not predefined but based on commitments to other agents. These commitments are specified in contracts. The semantics of these contracts are described by means of illocutionary deontic logic, the logic of obligations, authorizations and speech acts [Verharen, Dignum, 97].

2.4.2.2 Open approaches to coordination

Usually human organizations and societies use norms and conventions to cope with the challenge of social order. Norms and conventions specify the behavior that society members are expected to conform to and are suitable means for decentralized control. In most societies, norms are backed by a variety of social institutions that enforce law and order (e.g. courts, police), monitor for and respond to emergencies (e.g. ambulance system), prevent and recover from unanticipated disasters (e.g. coast guard, fire-fighters), etc. In this way, civilized societies allow citizens to utilize relatively simple and efficient rules of behavior, offloading the prevention and

recovery of many problem types to social institutions that can handle them efficiently and effectively by virtue of their economies of scale and widely accepted legitimacy.

Several researchers have recognized that the design of agent societies can benefit from abstractions analogous to those employed by our robust and relatively successful societies and organizations. There is a growing body of work that touches upon the concepts of norms and institutions in the context of multi-agent systems (cf. [Dignum, 1999], [Dignum, 2001], [Esteva et al., 2001]).

The benefit of an institution resides in its potential to lend legitimacy and security to its members by establishing norms. The electronic counterpart of the physical institution does a similar task for software agents: it can engender trust through certification of an agent and by the guarantees that it provides to back collaboration. However, the electronic institution can also function as the independent place in which all types of agent independent information about the interaction between the agents within the society is stored. E.g. it defines the message types that can be used by the agents in their interactions, the rules of encounter, etc. In general, institutions enable to:

- Specify the coordination structure that is used
- Describe exchange mechanisms of the agent society
- Determine interaction and communication forms within the agent society
- Facilitate the perception of individual agents of the aims and norms of an agent society
- Enforce the organizational aims of the agent society

In an agent society, the institution acts as mediator and animator for the members, who bring various skills and services, and customers (or groups of customers) who bring their problems and requirements. The most important service the institution provides is to regulate the interaction between members.

Although social issues are gaining importance in agent coordination research, MAS still provide a limited approach to coordination in the sense that coordination in MAS is mainly a matter of coordination of actions within the system. That is, it does not consider the ‘macro’ motivations of the users and stakeholders. However, organizational theory and social economics have devoted a great deal of research to this type of coordination which we think can be of value for the improvement of coordination issues in MAS. In section 2.5, these approaches are discussed in detail. Nevertheless, communication remains an important tool for coordination, both in human as in artificial systems. Communication issues in MAS are discussed in the following subsection.

2.4.3 Communication

The main challenge of coordination and collaboration among heterogeneous and autonomous intelligent systems (we mean here both humans and software) in an open, information-rich environment is that of mutual understanding. Only by sharing a mutual understanding of the domain will agents be able to exchange and combine information from heterogeneous sources. Communication and social interaction are

therefore the core characteristics of autonomous agents. A mechanism for communication must include both a **knowledge representation language** (to specify the internal behavior of agents) and a **communication protocol** (to specify the interactions among agents). Knowledge representation models are based on *ontologies* that define the domain model and vocabulary of a particular domain of discourse, and shared using *content languages* that represent the agent's mental model of the world (e.g. beliefs, desires, intentions). Given a particular domain of discourse, and a particular community of agents that know and do something in this domain, a communication language is needed that models the flow of knowledge and attitudes about such knowledge within the agent community. In the following we describe communication protocols and knowledge representation languages in more detail.

2.4.3.1 Communication Protocol

An Agent Communication Language (ACL) provides language primitives that implement the agent communication model. ACLs are commonly thought of as *wrapper languages* in that they implement a knowledge-level communication protocol that is unaware of the choice of content language and ontology specification mechanism. Most work done in the area of agent communication languages is based on the Language Action Perspective [Winograd, Flores, 1986] and Speech Act Theory [Searle, 1969], a formal model of human communication developed by philosophers and linguists.

2.4.3.1.1 Speech Act Theory

Speech Act Theory [Austin, 1962], [Searle, 1969] sees human natural language as actions, such as requests, suggestions, commitments and replies. Speech Act theory states that a language is used not only for making a statement but it also performs actions. For example, when someone asks someone else to do something, he/she is already causing an action. In Speech Act Theory, organizational communication is seen as the exchange of speech acts for the purpose of coordinating organizational activities. The theory provides the means to analyze communication in detail at three levels: content (locution), intention (illocution) and effect (perlocution). **Locution** is the information contained in an utterance. **Illocution** is the purpose that an utterance has, like informing, convincing, requesting, or demanding. **Perlocution** is the actual effect that a statement has. Form (syntax) of communication is less important than 'why' and 'what' is communicated.

Speech Act Theory is relevant to agent communication in that it serves as one (but not the only) formal basis for deciding on agent communication language primitives. Using speech act theory eases ambiguous semantic resolution, as compared to the natural languages. Speech acts are useful in that one can formally represent the intent of the speaker and the effect on the hearer. It is up to the agent theory and the agent infrastructure to ensure that agents in the community are ethical and trustworthy, and therefore that the perlocutionary behavior of a speech act on the hearing agent is predictable. All this is not the concern of ACLs, which are merely providing the language primitives. Still, the semantics of speech acts for a particular agent completely depends on the agent's belief, intention, knowledge about how to carry

out the operation, and the society to whom an agent belongs. These semantics are represented using the knowledge representation language.

The Language Action Perspective (LAP) is a practical application of the Speech Act Theory, which is used as a linguistic tool to model communication in Cooperative Information Systems [Flores, Ludlow, 1980]. The basic assumptions underlying the Language Action Perspective are [Verharen, 1997]:

- The primary dimension of human cooperative activity is language. Action is performed through language in a world constituted by language
- The meaning of sentences for the actors in a social setting is revealed by the kinds of acts performed
- Cooperative work is coordinated through language acts.
- The speech act is the basic unit of communication
- Speech acts obey socially determined rules
- The design of IT systems has a focus on getting things done, whenever work involves communication and coordination among people. The act of doing something, the patterns of interaction and their articulation are the primary concern of information systems design

2.4.3.1.2 Agent Communication Languages

Recent developments in the area of agent communication have resulted in the definition of two different ACLs based on the Speech Act Theory. The first one is KQML (Knowledge Query and Manipulation Language) developed in the context of the ARPA Knowledge Sharing Effort [Finin et al., 1997]. KQML consists of a set of communication primitives (called performatives, in accordance to Speech Act Theory terminology) which aim to support cooperation among agents in distributed applications. The KQML performatives enable agents to exchange and request knowledge, and to cooperate during problem solving. KQML doesn't care about the content language used to represent the mental. Its goal is to provide knowledge transportation protocol for blobs of content, in some ontology that the sending agent can point to and the receiving agent can access.

The second language is FIPA-ACL, the Agent Communication Language framework proposed by the Foundation for Intelligent Physical Agents [FIPA, 2002]. FIPA ACL is associated with FIPA's open agent architecture. As with KQML, FIPA-ACL is based on Speech Act Theory and is independent from the content language and is designed to work with any content language and any ontology specification approach. Furthermore, FIPA-ACL limits itself to primitives that are used in communications between agent pairs. The FIPA architecture has an Agent Management System that specifies services that manage agent communities.

Both FIPA-ACL and KQML are languages similar to those in the family of so-called coordination languages [Carriero, Gelernter, 1992]. These extend sequential languages with constructs to support concurrency and coordination. In a similar way, FIPA-ACL and KQML extend knowledge representation formalisms with knowledge communication primitives, and focus on defining knowledge level coordination

languages, which can be used to specify a range of cooperation strategies. Knowledge level coordination languages are situated at a higher level of abstraction with respect 'normal' coordination languages of distributed computing, as they support coordination not at the symbol-level but at the knowledge-level [Newell, 1993].

2.4.3.2 Representing and sharing knowledge

A specific feature of multi-agent systems is sociability which requires that agents should communicate with each other to cooperate, compete, or use services. In heterogeneous agent communities, where agents designed based on different architectures and internal representations interact, it is necessary to provide a means for agents to share their knowledge, which is represented in their internal state. The internal state of an agent is also referred to as mental agency, which refers to the mental concepts of an agent such as beliefs and intentions.

Languages are needed to describe things in a way that agents can understand. Natural languages such as English and Japanese are very powerful for building descriptions but the meaning of a natural language statement is not always clear and subject to different interpretations. (which is of course one of the reasons for the existence of lawyers). Many computer languages and systems have been built whose purpose is to define and describe things and situations. Specialized languages have been developed which are particularly good at describing certain fields. For example, STEP (Standard for the Exchange of Product Model Data) is an ISO standards project to develop mechanisms for the representation and exchange of computerized models of products in a neutral form. The goal is to enable a product representation to be exchanged without any loss of completeness or integrity. SGML is an example of a language that is designed to describe the logical structure of a document. There are other special languages for describing workflow, processes, chemical reactions, etc. However, it would be nice if there were some expressive languages and related computer systems which were good at representing a very broad range of things, like the natural languages, but which do not suffer the problems of imprecision and ambiguity. Agents can use such languages to share their knowledge, independently from the internal representation of that knowledge. Database systems and their languages (e.g., SQL, OQL) offer one general approach and certain object-oriented languages offer perhaps another. However, it is difficult or impossible to capture all kinds of information and knowledge in most of these general languages.

2.4.3.2.1 Content interchange languages

Content languages, in ACL terminology, are languages used by agents to exchange their information content while conversing. An ACL message's content, which contains descriptions in the content language, is distinct from the *propositional-attitude* of the message that defines the intention of the message, that is, the speech act type of primitive of the ACL message [Grosz, Labrou, 1999].

Such a content language is the Knowledge Interchange Format (KIF) which was developed within the DARPA knowledge sharing effort that also produced KQML as a communication protocol (cf. section 2.4.3.1.2). KIF is meant as a general-purpose content language. However, because agents are developed using different

frameworks, it is important for ACLs to support multiple, special-purpose, content languages. KIF defines a common language for expressing the context of a knowledge base to exchange. KIF proposed to use first order predicate logic to describe things within computer systems so that it can be used as a 'interlingua'. The syntax of KIF is a prefix version of first order predicate calculus and provides supports for non-monotonic reasoning and definitions. KIF can be used to encode knowledge about knowledge.

With the upcoming of the web, other languages appeared that also can be seen as content languages. HTML (Hyper Text Markup Language), the underlying language of most Web documents today, is a tag set that has been specifically designed to support display and hypertext linking. The use of HTML has grown exponentially because it is so easy to learn and to use. However, HTML is a "flat" tag set where all data is on an equivalent level of importance, and which main purpose is to describe the style and format of a document such that it can be read and displayed on different platforms. In order to be able to make document content understandable by machines, XML, the eXtensible Markup Language, was developed. XML is intended to make content more usable for distributing materials on the World Wide Web. A human may be able to tell the difference between a subtotal and a total, or a billing address and a shipping address, or a retail price and a sale price, but software agents, softbots and other programs need extra help. Indeed, XML is intended mainly to benefit computer programs. Although the tags created with XML resemble the HTML tags used today to create Web pages, there are two important differences: XML tags separate content from presentation, and XML is extensible, that is, it allows the creation of new tags for new and unforeseen purposes. ACML, Agent Communication Markup Language is a specification of a content language for FIPA-ACL that has been defined in XML [Grosz, Labrou, 1999].

An application of XML is RDF, the Resource Description Framework [Lassila, Swick, 1999]. RDF is a World Wide Web Consortium (W3C) recommendation that provides description facilities for (web-based) knowledge items. The objective of RDF is to support the interoperability of metadata. RDF allows descriptions of Web resources to be made available in machine understandable form. This enables the semantics of objects to be expressible and exploitable. That is, RDF provides support for the modeling of ontological concepts and relationships [Staab et al, 2000]. Once highly deployed, this will enable services to develop processing rules for automated decision-making and knowledge sharing.

2.4.3.2.2 Ontologies

Mechanisms to describe the meaning of the exchanged information are needed for meaningful interaction among agents. Possible basis for such a language for meaning description is the concept of **ontology**. In this sense, ontology is a specification of a conceptualization. That is, an ontology is a description of the concepts and relationships that can exist for a community of agents [Gruber, 1993]. Ontologies aim at capturing domain knowledge in a generic way and provide a commonly agreed understanding of a domain, which may be reused and shared across applications and groups. Ontologies provide a common vocabulary of an area and define - with

different levels of formality - the meaning of the terms and the relations between them. [Gomez-Perez, Benjamins, 1999].

Ontology – as a field of Philosophy – has a tradition of approximately 2500 years. Its underlying question "What exists? What is?" has found its way into cognitive sciences during the last decades in more specific forms related to a cognitive agent: For example in linguistics a variant of this theme is "What are the entities we speak about using natural language?" Cognitive Psychology is concerned with the question "What are the entities we perceive and reason about?" and Artificial Intelligence has to solve the problem "What is represented in a formal system?" In all these areas, research and answers have to be based on terms of languages (natural or formal) or concepts as the building blocks of categorization and reasoning.

Ontologies can be seen as the semantic middleware between knowledge sources and applications, in the same way as wrappers provide a 'physical' middle level between computer resources and applications. Construction of ontologies is a complex and lengthy process. Every knowledge item is described by a number of attributes, characterizing its context, content and format [Liao et al, 1999]:

- At the **format** level, each knowledge source is described in terms of its structure, access and format properties.
- **Context** should be expressed in terms of organizational structure and process models. Both the context and rationale for creation and intended use are important properties for some knowledge item.
- The **content** description of a knowledge item is typically highly specific, and based on its domain of application.

Knowledge sources, in all its different forms, are composed of signs. By sign we mean something that stands for something else, when interpreted by some individual interpreter in some individual situation. Semiotics is a cognitive framework, concerned with the study of signs. In its simplest form, a sign consists of two parts: the form of the sign and the meaning of the sign, that is, what it stands for [van Schooten, 1999]. For an agent, anything that involves interpretation may be called a sign: for example, smoke may be a sign of fire, a closed door may be a sign of a certain person's absence. This also includes signs of culture and convention, such as language, road signs, etc., at least when they indeed are interpreted as such.

Around the concept of sign, there are general classifications, such as syntax, semantics, and pragmatics. This formal classification corresponds roughly to the above concepts of format, content and context. In semiotics [Sowa, 2000], **syntax** refers to the rules governing the structure of a knowledge item, and the relations between symbols. **Semantics** is the relation between the symbols and things in the real world. **Pragmatics** refers to the relation between sign and sign user, in other words, why does the user use a sign, and what happens when a user uses that sign? Pragmatics relates symbols to the agents who use them to refer to things in the world, and to communicate their intentions about those things to other agents. One well-known pragmatic classification of sign transmissions (and language utterances in particular) is the classification into locution, illocution and perlocution, originally

proposed in the Speech Act Theory. Any ontology describing knowledge sources must consider syntax, semantics and pragmatics of that knowledge.

2.4.3.2.3 Context

Communication and social interaction are always embedded in a social context. The notion of *context* is called to account for a multifarious variety of phenomena, and includes syntactic, semantic and pragmatic aspects. In Artificial Intelligence (AI), McCarthy was the first to argue that formalizing context was a necessary step toward the designing of more general computer programs [McCarthy, 1987]. Other work comes from cognitive science where context is viewed as a way of structuring knowledge and its usage in problem solving tasks.

In a very general way, context can be seen as a collection of *things* (parameters, assumptions, presuppositions, etc.) a representation depends upon. The fact that a representation depends upon these things is called *context dependence*. The basic intuition is that locally produced knowledge (personal knowledge or the knowledge of a group or department) cannot be represented in a universal structure because we cannot be sure that this structure is understood in the same way by different agents (people, groups or software agents). To integrate knowledge from different sources, a process of *meaning negotiation* is needed [Bonifacio et al, 2000]. Integration of knowledge is therefore a mechanism of social agreement. A consequence of this is that since knowledge ‘exists’ in the context of a negotiation process, it has no existence when considered apart from its context. The motivations and the approaches to the problem of context are very different, and one might even wonder whether there is something as *the* problem of context, or rather a multiplicity of different problems very loosely related by the word *context* [Giunchiglia, Bouquet, 1997]. [Weigand et al., 1999] argues that context can be viewed according at three levels:

- *Locational level*, the physical or virtual location in which the message is represented.
- *Informational level*, the total of background knowledge relevant to the message that the communicative agents share.
- *Social level*, dependent on social institutions and conventions.

2.5 Coordination in Organizational Studies

The use of coordination in the remainder of this dissertation has been influenced by research on coordination in several other research fields. In the following, we highlight the views on coordination that currently hold in economics and organizational sciences, which are somewhat different but complementary to those taken in computer science, and distributed artificial intelligence, discussed in section 2.4.2.

2.5.1 Organizational Forms

Economics and organizational theory consider that relationships between and within organizations are developed for the exchange of goods, resources, information and so

on. Williamson argues that transaction costs are determinant for the choice of organizational model [Williamson, 1975]. Transaction costs will rise when the unpredictability and uncertainty of events increases, and/or when transactions require very specific investments, and/or when the risk of opportunistic behavior of partners is high. When transaction costs are high, societies tend to choose a **hierarchical** model in order to control the transaction process. If transaction costs are low, that is, products are straightforward, non-repetitive and require no transaction-specific investments, then the **market** is the optimal choice. Powell introduces **networks** as another possible coordination model [Powell, 1990]. Networks stress the interdependence between different organizational actors and pay a lot of attention to the development and maintenance of (communicative) relationships, and the definition of rules and norms of conduct within the network. At the same time, actors are independent, have their own interests, and can be allied to different networks. That is, transaction costs and interdependencies in organizational relationships determine different models for organizational coordination.

Table 2-1: Comparison of organizational forms

	MARKET	NETWORK	HIERARCHY
Coordination	Price mechanism	Collaboration	Supervision
Relation form	Competition	Mutual interest	Authority
Primary means of communication	Prices	Relationships	Routines
Tone or Climate	Precision/ suspicion	Open-ended/ mutual benefits	Formal/ bureaucratic
Range of cooperation	No cooperation expected	Negotiation of cooperation	Absolute cooperation expected
Conflict Resolution	Haggling (Resort to courts)	Reciprocity (Reputation)	Supervision

Coordination in markets is achieved mainly through a price mechanism in which independent actors are searching for the best bargain. Hierarchies are mainly coordinated by supervision, that is, actors that are involved in power-dependent relationships act according to routines. Networks achieve coordination by mutual interest and interdependency. The characteristics of the different forms of organization are summarized in Table 2-1 (adapted from [Nouwens, Bouwman, 1995]).

2.5.2 Social Structures

Social structures, or artificial social systems ([Moses, Tennenholtz, 1995], [Shoham, Tennenholtz, 1995]), define a social level where the multi-agent system is seen as a society of entities which define a structured pattern of behavior that enhances the coordination of agent activities [Vázquez-Salceda, 2003]. Social structures reduce the danger of combinatorial explosion in agent interaction, as they impose restrictions on the actions of agents. Social structures have been classified into the following groups [Findler, Malyankar, 2000]:

- **Alliance**: temporary group formed voluntarily by agents whose goals are similar enough. While in the alliance, agents give up some of their own goals and fully

cooperate with the other members of the alliance. They stay in the alliance as long as it is in their interest.

- **Team:** formed by a (possibly self-appointed) team leader that has some problem solving to do and recruits qualified members under its leadership.
- **Coalition:** similar to an alliance except that members of a coalition do not have to abandon their individual goals but engage only in those joint activities whose goals are not in conflict with their own.
- **Convention:** is a formal description of forbidden or preferred goals or actions in a group of agents
- **Market:** defines the mechanisms for transacting business by introducing two prominent roles: buyer and seller.

Apart from these types of social structures, multi-agent systems also make use of **referral networks** to model emerging structures [Yu, Singh, 2002]. In this case, the structure of a group of socially situated agents is not specified a priori but emerges from the interactions between agents. The types of social structures classified by Findler and Malyankar and referral networks are specific of multi-agent systems, and can be covered by the more generic types described in section 2.5.1: market, hierarchy and network. Teams are a sort of hierarchy, and alliances, conventions and coalitions, as well as referral networks can be seen as special cases of networks. Furthermore, it can be argued whether the type convention in Findler and Malyankar's classification really is a social structure, or rather a characteristic of social structures that can actually apply to any of the other types.

2.6 Discussion

KM tasks have often a collaborative aspect, that is, individuals best acquire and use knowledge by reusing information already collected and annotated by others, or by making use of existing relations among people (or communities). Furthermore, a KM system must be able to adapt to changes in the environment, to the different needs and preferences of users, and to integrate naturally with existing work methods, tools and processes. That is, the suitability of agent technology in the KM area arises from the need for KM systems to be *reactive* (able to respond to user requests or environment changes) and *proactive* (able to take initiatives to attend to user needs).

Agent-based models for knowledge management use agents as autonomous entities (like employees in a company) that are endowed with certain behaviors, and the interactions among these entities give rise to complex dynamics. In this context, agents can be defined as '*one that acts or has the power or authority to act*' or '*one that takes action at the instigation of another*'. The concept of agent in this sense is not new, nor restricted to software. In this perspective, agents are autonomous social entities that exhibit flexible, responsive and proactive behavior. There is currently an increasing interest in the use of multi-agent concepts for KM, mainly motivated by the fact that, like multi-agent systems, KM domains involve an inherent distribution of sources, problem solving capabilities and responsibilities [van Elst et al., 2003a], [Bonifacio et al., 2002], [Gandon et al., 2000]. That, is, the integrity of the existing organizational structure and the autonomy of participants must be maintained, which

calls for a autonomous and distributed representation of KM systems. Interactions in KM environments are fairly sophisticated, including negotiation, information sharing and coordination, and require complex social skills with which agents can be endowed. Furthermore, solutions for KM problems cannot be entirely prescribed from start to finish and therefore reactive and proactive problem solvers are required that can respond to changes in the environment, react to the unpredictability of business processes and act on opportunities when they arise.

In our opinion, the agent paradigm is particularly well suited to model KM support systems due to the autonomous, re- and proactive character of agents which meet the characteristics of KM [Van Elst et al., 2003b], [Dignum, 2003]:

- Knowledge in organizations is distributed. That is, KM domains involve an inherent distribution of data, problem solving capabilities and responsibilities. Agents are suitable here due to their characteristics of autonomy and social ability.
- KM should follow the existing organizational structure and maintain the autonomy of its divisions. Again here the autonomous nature of the agents is suitable.
- KM is a social process. Interactions in KM environments are fairly sophisticated, including negotiation, information sharing, and coordination. This can make use of the complex social skills with which agents are endowed.
- Business processes and knowledge processes are often in conflict. The maintenance and use of knowledge sources is often not seen as a main activity, and primary business processes will take priority on the attention of a worker. That is, KM domains call for a functional and dynamic separation between knowledge use and knowledge sources. Agents can act as mediators between maintenance and application of knowledge.
- KM must deal with a changing environment. Often, KM systems are directed to environments where changes are frequent. Centralized solutions are therefore not suitable, due to maintenance costs and lack of flexibility. Agents are suitable here due to their reactive and proactive characteristics.
- The solution for KM problems cannot be entirely prescribed from start to finish and therefore problem solvers are required that can respond to changes in the environment, to react to the unpredictability of business process and to proactively take opportunities when they arise. This characteristic requires the reactive and proactive abilities of agents.
- KM must deal with individual recognition and requirements. That, one solution does not fit all, and systems must be adaptable to user preferences and profiles.

In our opinion, agent concepts can lead, on the one hand, to advanced functionality of KM systems (e.g. personalization of knowledge presentation and matching supply and demand of knowledge), and on the other hand, the rich representational capabilities of agents as modeling entities allow faithful and effective treatment of complex organizational processes. Currently, the use of agents in KM falls basically into two types of approaches: implementation technique or conceptual modeling.

In agent-based implementations of KM systems, software agents are employed as tools to manage loosely coupled information sources, to provide unifying presentation of distributed heterogeneous components and to personalize knowledge presentation and navigation. Possible agent-based services in an KM system are [Klusck, 1999]:

- search for, acquire, analyze, integrate and archive information from multiple heterogeneous sources,
- inform us (or our colleagues) when new information of special interest becomes available,
- negotiate for, purchase and receive information, goods or services,
- explain the relevance, quality and reliability of that information, and
- learn, adapt and evolve to changing conditions.

However, current agent society models are not always well suitable for KM because either they take a centralist approach to organizational design (cf. for example [Wooldridge et al., 2000]), or have a completely emergent view on agent interactions. KM support systems require however the integration of individual desires with organizational requirements.

One of the main contributions of agent-based modeling of KM environments (often referred to as **Agent-Mediated Knowledge Management, AMKM**) is that it provides a basis for the incorporation of individual initiative and collaboration into formal organizational processes. That is, a system does not need to be completely designed and fixed a priori but it is developed as a set of components and interaction processes that can be adjusted to the needs and requirements of the specific participants. This implies that the development AMKM systems requires a theory of organization design, and knowledge on how organizations may change and evolve over time. Sociological organizational theory and social psychology are clearly important inputs to the design of such systems. Moreover, for the design of open societies, concepts from political theory may be necessary. Open systems permit the involvement of agents from diverse design teams, with diverse objectives, which may all be unknown at the time of design of the system itself. How the system as a whole makes decisions or agrees on joint goals will require the adoption of specific political philosophies, for example whether issues are subject to simple majority voting or transferable preference voting, etc. [Luck et al., 2003]. The OperA model described in this dissertation is a proposal for a framework for AMKM that follows these ideas and integrates research from several disciplines.

2.7 Conclusions

In this chapter we have presented the state of the art in research related to the subject of this dissertation. In particular, research in KM, agent and agent societies, and coordination were presented and the contributions and cross-relations discussed. We have described the main aspects of each research area, that are relevant for the dissertation and discussed how integration between areas can be achieved. The realization of such integration is the objective of the OperA framework that will be presented in the remainder of this dissertation.

Chapter 3

The OperA Model for Agent Societies

'The problem is to find a form of association which will defend and protect with the whole common force the person and goods of each associate, and in which each, while uniting himself with all, may still obey himself alone, and remain as free as before.'
– J.J. Rousseau, Le Contract Social, 1762

In this chapter we introduce the model for agent societies **OperA** (**O**rganizations **per** **A**gents).⁶ This framework emerges from the realization that in organizations interactions occur not just by accident but aim at achieving some desired global goals, and that participants are autonomous, heterogeneous and not under the control of a single authority. The main focus of this research is thus on designed societies, with explicit objectives and structure, in opposition to emergent societies, that result from the 'ad hoc' interaction of agents. An OperA model can be thought of as a kind of abstract protocol that governs how member agents should act according to social requirements. Interaction is specified in contracts, which can be translated into formal expressions (using the Logic for Contract Representation, described in chapter 4), and therefore ensure that compliance can be verified.

The chapter is organized as follows. Section 3.1 provides background motivation to the concept of agent societies, and in particular to the idea that society design must be independent from the design of its members. Related work is discussed in section

⁶ The name illustrates the dual relation between organizations and agents, the fact that organizations are outmost dependent on its agents, but, as in a musical opera, a script is needed that guides and constrains the performance of the actors, according to the motivations and requirements of the society designer.

3.2. The basic architecture of the OperA framework is described in section 3.3. The next three sections give a detailed description of each of the three models of OperA: the Organizational Model is described in section 3.4, the Social Model in section 3.5 and the Interaction Model in section 3.6. Finally, section 3.7 indicates directions for future research and presents our conclusions.

This chapter is a modified and extended version of [Dignum et al., 2002a], [Weigand, et al., 2003], [Dignum et al., 2002b]. Section 3.5.2 includes and extends ideas from [Dastani et al., 2002] and [Dastani et al., 2003].

3.1 Motivation

The purpose of any society is to allow its members to coexist in a shared environment and pursue their respective goals in the presence or in co-operation with others. A collection of agents interacting with each other for some purpose and/or inhabiting a specific locality can be regarded as a society. Societies usually specify mechanisms of social order in terms of common norms and rules that members are expected to adhere to [Davidsson, 2000]. An organization can be defined as a specific solution created by more or less autonomous actors to achieve common objectives. Organizational structure can therefore be viewed as a means to manage complex dynamics in (human) societies. This implies that approaches to organizational modeling must incorporate both the structural and the dynamic aspects of such a society.

A social system, or society, consists of both a social structure of interrelated institutions, statuses, and roles and the functioning of that structure in terms of social actions and human interactions. Based on the ideas of Auguste Comte (1798-1854), a social system is assumed to include both social change (Comte's dynamics) - the processes and patterns of action and interaction - and social stability (Comte's statics) - the stable structural form of the society [Comte, Lenzer, 1998]. Furthermore, a social system constitutes a unitary social whole reflecting a real value consensus - the sharing of common values, social norms, and objectives. Organizations, as social systems, therefore comprise a *factual* and a *procedural* dimension [Sichman, Conte, 1998]. These dimensions are a fundamental basis for organizational research, and come back in studies from different disciplines, such as organizational theory [Lawrence, Lorsch, 1967], [Scott, 1987], AI and multi-agent systems [Gasser, 2001], [Rao, Georgeff, 1995] and distributed computing [Minsky, Rozenshtein, 1989]. The factual dimension consists of the observable behavior of the organization, that is, high level goals, inputs and outputs. The procedural dimension has to do with how this behavior is obtained, that is, the division of labor into roles, the determination of authority lines and the establishment of communication links.

An organization can be seen as a set of entities and their interactions, which are regulated by mechanisms of social order and created by more or less autonomous actors to achieve common goals. Business environments must furthermore consider the behavior of the global system and be able to incorporate the collective characteristics of an organization such as stability over time, some level of predictability, and clear commitment to aims and strategies. This indicates the need to extend current approaches to MAS to include an organizational perspective. That is,

agent societies must be able to define the global aims as well as the roles and responsibilities of participants. When these requirements are met, agent societies will be an effective platform for virtual organizations because they provide mechanisms to allow organizations to advertise their capabilities, negotiate their terms, exchange rich information, and synchronize processes and workflow at a high-level of abstraction [Preece et al., 1999].

From an organizational perspective, the main function of an individual agent is the enactment of a role that contributes to the global aims of the society. That is, society goals determine agent roles and interaction norms. Agents are actors that perform role(s) described by the society design. The agent's own capabilities and aims determine the specific way an agent enacts its role(s). However, the society is often not concerned about which individual agent will actually play a specific role as long as it gets performed. Several authors have advocated role-oriented approaches to agent society development, especially when it is manifest to take an organizational view on the application scenario [Dignum et al, 2002b], [Zambonelli et al., 2001a].

Organizational design can be either pre-established or dynamically formed. In the pre-established case, organizational design is established offline. That is, the participating agents are not responsible for the basic definition of the organization. In dynamically formed organizations, design is the result of 'negotiation' between participating agents. In multi-agent research, organizations are often viewed as a structural relationship between agents. Current multi-agent approaches to organizational design often consist of rigid structures where agents cannot really deviate from the expected behavior. That is, interaction is dictated by protocols, at 'instruction' level [So, Durfee, 1998]. Although powerful and robust, such approaches leave very little room for individual initiative of their populations and therefore cannot easily react to changes and extensions to the environment or incorporate 'foreigner' agents. For instance, [Esteva et al., 2002b] state that in agent organizations, agents must be '*governed through a formal and computational device that will supervise all the individual agent interactions and act as a dynamic two way illocution filter that is consistent with the role that the agent is playing*'. One of the first attempts to design more flexible systems, where agents can reason about deviation from expected behavior can be found in [Dignum, 1999].

Current work on distributed software systems, proposes models based on (semi) autonomous heterogeneous software entities [Finkelstein, Kramer, 2000]. In open systems there are no central control or expectations on the architecture of the different agents, and therefore society design cannot be based on the internal characteristics of the agents. Within an open society, the structure is typically just a generally accepted communication language and a limited set of roles [Davidsson, 2001]. The question that arises in such scenario is then: if agents are autonomous and independently designed, how can it be guaranteed that their interaction will reach any desired state, that is, the society will behave in an expected way?

Traditional multi-agent models often assume an individualistic perspective on the environment. Agents are taken as autonomous entities pursuing their own individual goals based on their own beliefs and capabilities. In this perspective, global behavior emerges from individual interactions and cannot easily be managed or specified

externally. However, in business environments, the behavior of the global system must be taken into account and structural characteristics of the domain have to be incorporated. That is, the design of the agent society must consider organizational characteristics such as stability over time, some level of predictability, and commitment to aims and strategies, etc. These are the rules and global objectives that govern the activity of an enterprise, group, organization or nation. Such characteristics are often specified top-down and imposed on the participants. Global characteristics are external to each individual agent and independent from the own goals and behavior of the agent, and therefore cannot easily be incorporated in a multi-agent architecture that starts from an individualistic perspective. This implies that, to a certain degree, agent societies must be pre-established, and organizational design cannot be completely left to the result of autonomous interaction. On the other hand, the volatility of business environments stresses the need for models and systems that accommodate changes required by new or different organizational aims with a minimum impact on the already existing services. From the organizational point of view this creates a need to check conformance of the actual behavior of the society to the behavior desired by the organization [Dignum et al., 2002a].

The above considerations can be summarized in the recognition that there is a clear need for multi-agent frameworks that combine and use the potential of a group of agents for the realization of the objectives of the whole, without ignoring the individual aims and ‘personalities’ of the autonomous participant agents. That is, as already mentioned in chapter 1, in order to represent interactions between agents in such an open context, a framework is needed that meets the following requirements:

- **internal autonomy requirement:** interaction and structure of the society must be represented independently from the internal design of the agents
- **collaboration autonomy requirement:** activity and interaction in the society must be specified without completely fixing in advance the interaction structures.

The development of such a framework, with furthermore a formal logical semantics, is the objective of our research. Our approach consists of a 3-layered model that separates the concerns of the organization from those of the individual. The top layer describes the structure and objectives of a system as envisioned by the organization, and the bottom layer the activity of the system as realized by the individual agents. In order to connect individual activity with organizational structure we add a middle layer that describes the agreed agent interpretation of the organizational design. This framework, **OperA**, is described in the remainder of this chapter. A formal model of OperA, based on the temporal deontic logic described in chapter 4, is presented in chapter 5.

3.2 Related work

Traditional approaches to multi-agent systems take an individual perspective on the system, that is, are mainly interested in the effects of interaction on the internal architecture of the agents. On the other hand, current multi-agent approaches either take a centralist approach to organizational design (cf. for example [Wooldridge et al., 2000]), or take an emergent view in which agent interactions are not pre determined,

thus making it impossible to make any predictions on the behavior of the whole systems.

Organizational approaches to the design of multi-agent systems are however gaining terrain in agent research and several research groups have presented models for agent societies in the line of our proposal. Recent developments recognize that the modeling of interaction in MAS cannot simply rely on the agent's own architectures and (communicative) capabilities. Furthermore, organizational engineering of MAS cannot assume that participating agents will act according to the needs and expectations of the system design. Concepts as organizational rules [Zambonelli, 2002], norms and institutions [Dignum, Dignum, 2001], [Esteva et al, 2001], and social structures [Parunak, Odell, 2002] arise from the idea that the effective engineering of MAS needs high-level, agent-independent concepts and abstractions that explicitly define the organization in which agents live [Zambonelli et al, 2001a].

One of the first works in this area is the Agent/Group/Role model, AGR for short, proposed by Ferber and Gutknecht [Ferber, Gutknecht, 1998]. This organization model structure includes high level concepts such as groups and roles within groups, and (intra-group and inter-group) role interaction. The model is well suitable to model social structures but not geared to the modeling of dynamic interactions. AGR models interaction as part of a role description and does not consider the normative aspects of interaction. Furthermore, all interactions are represented as definite, fixed protocols involving roles, which leaves no room for individual interpretation by the agents fulfilling the roles. This model was used as basis for a proposal for representation of social structures in AUML that does describe interaction between roles [Parunak, Odell, 2002].

The model developed by the Alfebiite consortium is meant for the design of open agent societies and focuses in aspects of security and legal consequences of agent action in agent societies [Artikis, Pitt, 2001], [Artikis et al., 2002]. The model includes representation primitives for agents, constraints, communication language, roles, social states and agent owners. This model presents a very interesting extension to agent society modeling by providing explicit primitives to represent stakeholders, as owners of the agents participating in the society. Furthermore, the model has a logical semantics, comparable to the one described in this thesis. A main difference with OperA lies in the way roles and agents are associated and their lack of primitives for the representation of groups and complex interaction and coordination in a society (for which OperA uses scenes scripts and transitions).

Esteva and his colleagues [Esteva et al., 2001] have devised a formal specification language to design open agent systems as *electronic institutions* with focus on the normative aspects of societies. This proposal aims at the modeling of institutionalized electronic organizations (institutions). In this approach, roles are defined as patterns of behavior, normative rules are specified to limit or enlarge the space of agent actions and scenes are defined in order to represent the different contexts within an organization in which agents can interact. However, this framework takes a very low level approach to abstract interaction, by demanding that all interaction be expressed in terms of fully specified protocols.

Although all the above models make a formal distinction between role and agent, this distinction is in our opinion not really used to its full potential. Either agents are treated as ‘pure’ performers of roles, or roles are seen as placeholders for agents. That is, there is no room to describe individual performances and particularities of a specific agent enacting a specific role. This is also the case with MAS methodologies, such as MaSE [DeLoach, 1999], Gaia [Wooldridge et al., 2000] and SODA [Omicini, 2001], that have adopted role and other social concepts to the analysis and design phases of the development of multi-agent systems. These methodologies are based on a centralized approach to system design, in which agents are designed to fulfil the roles, and do not allow for open development of societies and incorporation of agent identity in role enactment. We will further discuss MAS development methodologies in chapter 6.

Finally, recently a normative model for organizations has been presented: HARMONIA [Vázquez-Salceda, Dignum, 2003]. HARMONIA is a framework to model electronic organizations from the abstract level where norms usually are defined to the final protocols and procedures that implement those norms. Its objective is to fill the gap in previous approaches, which work either at the level of norm formalization or at the procedural level. It is composed of four levels of abstraction:

- **Abstract Level:** where the statutes of the organization are defined in a high level of abstraction along with the first abstract norms.
- **Concrete Level:** where abstract norms are iteratively concretized into more concrete norms, and the policies of the organization are also defined.
- **Rule Level:** where concrete norms and policies are fully refined, linking the norms with the ways to ensure them.
- **Procedure Level:** where all rules and policies are translated in a computationally efficient implementation easy to be used by agents.

Although HARMONIA has been partially formalized, using multi-modal logics and some existing implementations (such as the ontology and procedure level) it still lacks a formal description of the semantic relations between the levels.

3.3 OperA Architecture

OperA is a framework for the specification of multi-agent systems that distinguishes between the mechanisms through which the structure and global behavior of the model is described and coordinated, and the aims and behavior of the service-providers (agents) that populate the model [Dignum et al., 2002a], [Dignum, Dignum, 2001]. The framework is meant to describe systems at conceptual level. It allows for the formal specification of agent societies in a way that is, on the one hand, easily understandable, such that it can be used to discuss with domain experts that are not knowledgeable in agent theory, but, on the other hand, is based on a formal semantics (cf. chapter 5) that make verification possible. Furthermore, in chapter 6, we provide a methodology for the domain directed development of systems based on the OperA framework. An important area for future research is the development of a computer implementation to automate and support the specification of OperA models.

The OperA framework represents interaction between agents in a way that:

- (1) is independent of the internal design of the agents
- (2) distinguishes organizational characteristics from agents' own goals
- (3) creates dynamic links between organizational design and agent populations
- (4) allows for the adaptation of interaction patterns to the characteristics of specific populations.

In our view, the development of agent societies is based on two competing goals. On the one hand, the structure and requirements of the society owners must be captured in the society design (OM), and on the other hand, agents must be available that are able and interested in enacting society roles. From the point of view of society design, the reasons why an agent wants to enact a role are not relevant. However, from the agent's perspective, mechanisms must be developed that allow the incorporation of role characteristics into the agent's architecture. Even though this is not the focus of this thesis, we have done some investigative work in this area and will present some preliminary ideas and results in section 3.5.2 [Dastani et al, 2002, 2003]. Contracts are introduced as a means to integrate top-down specification of organizational structures with the autonomy of the participating agents. OperA consists of three interrelated models, depicted in Figure 3-1.

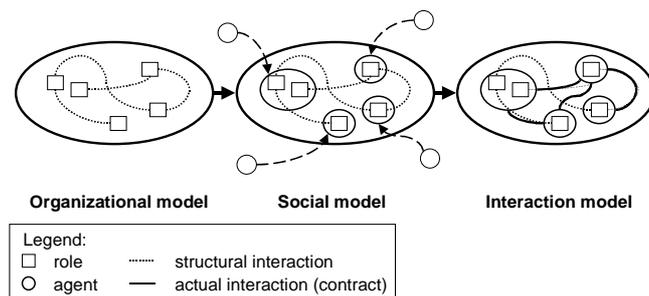


Figure 3-1: Organizational framework for agent societies

The three components of an OperA model are:

- The organizational structure of the society, consisting of roles and interactions, as intended by the organizational stakeholders, is described in the **Organizational Model (OM)**.
- Roles in the OM are 'filled in' by active populations of agents, specified in the **Social Model (SM)** in terms of agreements concerning the enactment of roles by individual agents.
- Finally, given an agent population for a society, the **Interaction Model (IM)** describes the possible interaction between agents.

The organizational model of a society reflects the requirements of the organization's owners. In OperA, agents are seen as autonomous communicative entities that will perform society role(s) as a means to realize their own goals according to their own internal aims and architecture. Whereas constrained by the

organizational design, activity is dependent on the capabilities of actual agents present in the society at a given moment. This means that several agent populations are possible for each organizational model, and the objectives of the society will be achieved in different ways. The characteristics and requirements of the society specified in the society model are then incorporated in the software agents themselves. Agents will thus contain enough information and capabilities to interact with others according to the society specification.

We have also developed a generic methodology to analyze a given domain and determine the type and structure of the agent society that best models that domain [Dignum, Weigand, 2002], described in chapter 6. This methodology, which is tailored to OperA but can as well be generically used to specify any agent society, describes generic facilitation and interaction frameworks for agent societies that implement the functionality derived from the coordination model applicable to the problem domain. Standard society types as market, hierarchy and network, can be used as starting point for development and can be extended where needed and determine the basic norms and facilitation roles necessary for the society. These coordination models describe the different types of roles that can be identified in the society and issues such as communication forms, desired social order and co-operation possibilities between partners. OperA combines results from several authors and draws from research in many disciplines. The main contribution of our work is that it presents a comprehensive model for organizational multi-agent systems in which the concerns of both organization and individual are acknowledged, integrated and respected. In this chapter we present the architecture of OperA.

3.3.1 Road Map

As described above, OperA models are based on the organizational characteristics of a society as envisioned by the society owner. The link between OperA models and individual agents is made through the role enacting agreements specified in the social model. Commitments concerning task-directed interaction between role enacting agents are then specified in the interaction model. The global architecture of OperA is depicted in Figure 3-2.

The **Organizational Model (OM)** specifies the organizational characteristics of an agent society in terms of four structures: social, interaction, normative and communicative. The **social structure (SS)** specifies objectives of the society, its roles and what kind of model governs coordination. The **interaction structure (IS)** describes interaction moments, as scene scripts, representing a society task that requires the coordinated action of several roles, and gives a partial ordering of scene scripts, which specify the intended interactions between roles. Society norms and regulations are specified in the **normative structure (NS)**, expressed in terms of role and interaction norms. Finally, the **communicative structure (CS)**, specifies the ontologies for description of domain concepts and communication illocutions. The way interaction occurs in a society depends on the aims and characteristics of the application, and determines the way roles are related with each other, and how role objectives and norms are ‘passed’ between related roles.

In the **Social Model (SM)**, the enactment of roles by agents is fixed in social contracts that describe the capabilities and responsibilities of the agent within the society, that is the agreed way the agent will fulfil its role(s). A social contract defines a **role-enacting agent (rea)**. We call a set of agents enacting society roles at a given moment, an agent **population**. Because the society designer does not control the design and behavior of individual agents, there is a need to verify the actual behavior of a society population. This is done by analyzing the agreements specified in the social contracts. The use of contracts to describe activity of the system allows, on the one hand, for flexibility in the balance between organizational aims and agent desires, and, on the other hand, for verification of the outcome of the system. The social contract provides a ‘window’ to the agent, through which other agents know what to expect and how to interact with the agent.

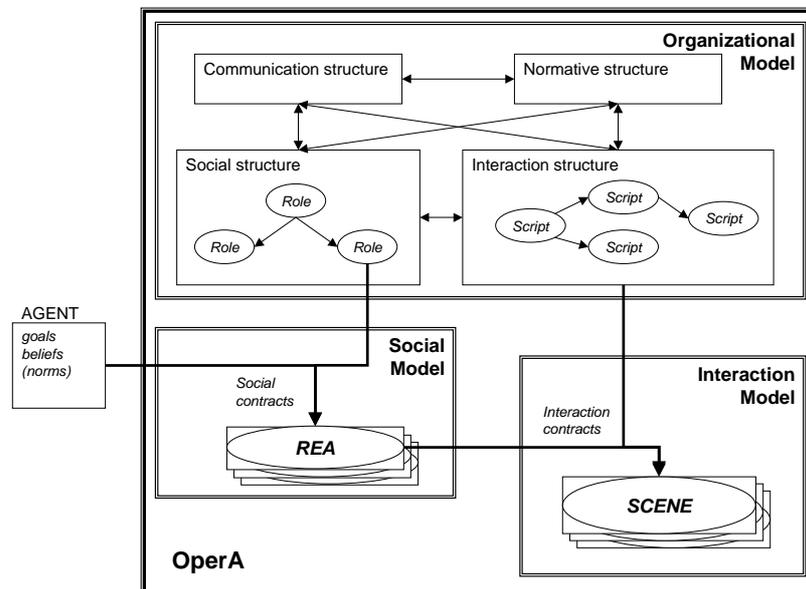


Figure 3-2: OperA Architecture

In the **Interaction Model (IM)** concrete interaction scenes are dynamically created by role-enacting agents, based on the interaction scripts specified in the OM. Role enacting agents negotiate specific interaction agreements with each other. Such interaction commitments are fixed in interaction contracts. As in the SM, interaction contracts allow on the one hand for flexibility and personalization of the organizational design, and on the other hand, for the verification of design and activity. That is, it can be verified whether the interaction agreements between a specific population satisfy and are sufficient for the organizational interaction aims specified in the OM.

3.3.2 Working example

In the following, we will illustrate the different components of a society using the example of a multi-agent system that models a conference organization⁷. The global objective of this society is to realize a conference. Stakeholders in this society, are organizer, authors, PC members and participants. The objective of the organizer is to organize a successful conference, authors want to get their papers accepted, the PC member aims at assuring the quality of the program, and the participant hopes for a high quality conference. Facilitation activities can be described in terms of an organizer role that administrates the conference, a chairperson role responsible to regulate conference sessions, etc. The example will be further detailed as the components of OperA are presented.

3.4 Organizational Model

The **Organizational Model** (OM) of OperA describes a social system from the perspective of the organization, that is, it describes what are aims and concerns of the organization with respect to the social system. Organizations are usually designed with a specific result and structure in mind. From this perspective, an organization is defined by its externally observable *objectives* and by the *means* needed to reach such objectives. We identify four areas in the specification of societies: social structure, interaction, normative behavior and communication. The OM combines therefore aspects from role theory with ontologies, normative description and process specification.

The OperA specification of a society includes the description of concepts and relationships holding in the domain, and of the holding social values and norms, that is, what is to be accepted as ‘good’ social behavior. The former is the aim of the **communicative structure**, which specifies the ontology and the communication language that is used in the society. The latter is achieved by the **normative structure**, which describes the expectations and boundaries for agent behavior as envisioned by the organization. Such explicit specification of normative elements is necessary because agents are assumed to be autonomous and therefore their activity cannot be enforced.

Furthermore, society structure must include the description of society objectives and the means for their realization. Objectives of a society specify the aims of the society, that is, the desired states of affairs in the life of the society. The objectives of an organization are usually quite large and complex. According to the principle of bounded rationality, people (and artificial agents) are rational within limits, that is, there are boundaries on the information that can be taken in account and on the processing capabilities of the decision makers [Simon, 1957]. In this light, performance in organizations increases when organization objectives are split into smaller, less complex, units or roles. Therefore, the OM includes a component, the

⁷ This example has been often used in MAS literature, e.g. [Zambonelli et al., 2001b]

social structure, where the description of relationships and capabilities of roles and activities is specified. In parallel to the description of the roles, the OM must also indicate the abstract processes that according to the organization's view must be used to achieve its objectives. Taking into account the collaboration autonomy requirement (discussed in section 3.1), such abstract processes are not procedural but declarative in nature. That is, the **interaction structure** component of the OM describes the states that agents must strive to achieve, instead of the activities to perform.

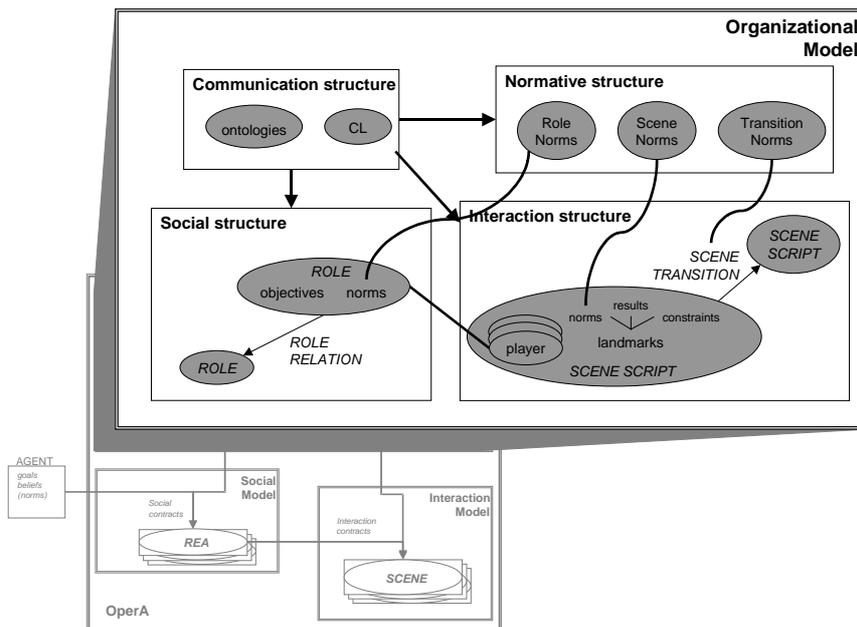


Figure 3-3: OM Architecture

The architecture of the OM is presented in Figure 3-3. The components of the OM description are summarized Figure 3-4. In the remainder of this section the four structures are described in detail.

Organizational Model definition	
Social structure:	The social structure definition
Interaction structure:	The interaction structure definition
Normative structure:	The normative structure definition
Communicative structure:	The communicative structure definition

Figure 3-4: Elements of OM definition

3.4.1 Social Structure

The social structure of an organization describes the roles holding in the organization, including their objectives, rights and requirements, possible groups of roles, and the

relations between roles. The elements of the social structure of the OM are described in Figure 3-5.

Social Structure definition	
Roles:	A list of role definitions
Role dependencies:	A list of triples of two role names and the name of the relationship between them
Groups:	A list of sets of roles

Figure 3-5: Elements of the definition of a social structure

Roles identify activities and services necessary to achieve social objectives and enable to abstract from the specific individuals that will eventually perform them. From the society design perspective, roles provide the building blocks for the agent systems that can perform the role, and from the agent design perspective, roles specify the expectations of the society with respect to the agent's activity in the society. Roles also define normative behavioral repertoires for agents [Odell et al., 2003]. Our specification of the social structure of the OM is furthermore based on ideas from role theory [Biddle, 1979].

Role theory bridges social psychology, sociology, and anthropology, and has recently generated interest among agent researchers. Its central concern has been with patterns of conduct, that is, expectations, identities, and social positions; and with context and social structure. It rests on a theatrical analogy [Goffman, 1959]: Individuals in a society occupy *positions*, and their *role* performance in these positions is determined by social norms, demands, and rules. Their position is influenced by the role performances of others in their respective positions and possibly by those who observe and react to the performance as well as by the individual's particular capabilities and personality. The social "*script*" may be as constraining as that of a classical theater play, but it frequently gives the actors some freedom of interpretation; the "*director*" is often present in real life as a supervisor, parent, teacher, or coach; the "*audience*" in life consists of all those who observe the behavior of that individual; the "*performance*" of an individual in a society, as in the play, is attributable to one's familiarity with the "*part*," one's personality and personal history in general, and more significantly, to the "*script*" which others define in so many ways. The role perspective assumes that performance results from the social prescriptions and behavior of others, and that individual variations in performance, to the extent that they do occur, are expressed within the framework created by these factors [Biddle, 1979]. In essence, role models deal with collaboration and coordination and specify collaboration relationships between entities without fixing a priori the complete interaction process. The organizational model of a society therefore must describe the 'parts' to be played in order to achieve the objectives of the society in the *ideal* abstract way envisioned by society design.

Society objectives form the background for the OM and are the basis for the definition of the objectives of roles, in the sense that the society objectives will be realized by the realization of role objectives. In the same way, society values determine which norms hold for different roles or groups of roles. In a sense, the whole society can be seen as a super-role whose objectives are delegated to

organizational roles and whose norms hold for all roles. The way society objectives are ‘split’ into role objectives and what kind of ways are possible to realize those objectives depends on the requirements and characteristics of the domain.

Roles can be organized into **groups**. In its most basic form, groups are just a way to refer to a set of roles. This can be useful when in an interaction scene, a participant is not an actor of a specific role but can be acting one of several roles. However, the most relevant feature of a group is that it can specify norms that must hold for enactors of roles in the group. For any society, the trivial group of roles is the group that contains all roles in the society.

Finally, **role dependency** between two roles means that one role is dependent on another role for the realization of its objectives. How role objectives are actually passed between two roles depends on the type of coordination.

3.4.1.1 Roles

A role is the abstract representation of a policy, service or function. Roles typically describe an organizationally-sanctioned structured bundle of activity types [Gasser, 2001]. In the OM, role descriptions identify the activities and services necessary to achieve society objectives and enable to abstract from the individuals that will eventually perform the role.

Role Definition	
Role id:	A unique name with which to refer to this role
Objectives:	A set of landmarks that describe the desired results of this role
Sub-objectives:	A set of landmarks that described desired intermediate states for role objectives
Rights:	A set of expressions identifying the rights of this role
Norms:	A list of normative expressions that apply to this role
Type:	Either <i>external</i> or <i>institutional</i> , indicating which type of agents can apply for this role

Figure 3-6: Elements of a role definition

Roles are described in terms of **objectives** (what an actor of the role is expected to achieve) and **norms** (how is an actor expected to behave). Furthermore, role descriptions also specify the **rights** associated with the role and the **type of enactment** of the role, that is, whether it is an institutional role or a external role. The elements of role definition are summarized in Figure 3-6.

The objectives and sub-objectives of a role are expressed in the society language. Such expressions can be more or less restrictive on the actor performance: that is, the more aspects that are fixed in the expressions, the less freedom an agent enacting the role has to decide on how to achieve the role objectives and interpret its norms. Following the ideas of [Smith et al., 1998], we call such expressions **landmarks**. Formally, landmarks are conjunctions of logical expressions that are true in a state [Kumar et al., 2002]. Landmarks will be further specified in chapter 5. Intuitively, landmarks provide a description of a place or situation, which is enough to identify it but without prescribing any specific process. For example, if some one tells you, ‘I

will meet you in the park to the left of the high red building on Main Street', if you are knowledgeable of the context (that is of the place I'm referring to), you will usually know what is expected of you. However, how to get there and which path to take, depends on your own wishes, constraints and present location. In the same way, we use landmarks in OperA to describe the characteristics of a role or scene. Several different specific actions can bring about the same state, and therefore, landmarks actually represent families of protocols. The use of landmarks to describe activity enables the actors to choose the best applicable actions, according to their goals and capabilities. The level of specification of landmarks determines the degree of freedom the actors have about their performance. In the following, we will describe in more detail the elements of a role.

Role Objectives. Role objectives are defined as states of affairs expected to be achieved in the environment. Once a society model is animated, the objectives of a role are expected to be executed by the agent(s) enacting that role, that is, role objectives should become part of the goals of the enacting agent. Intuitively, role objectives enable the 'translation' between society objectives and agent goals. At this level of specification, role objectives do not really have a fixed semantics since roles are not performative entities but mere 'placeholders' for actors. The actual semantics of objectives depend on the way objectives are treated and assumed by the agent acting the role and on the semantics of agent goals in the agent model.

Definition 3.1. Role Objective

A role objective is a predicate describing the ideal state for the role. Role objectives are represented by $\rho = p(t_1, \dots, t_n)$, where $p(t_1, \dots, t_n)$ is a predicate in the domain language⁸. P_r is the set of objectives of role r .

Roles are identified by its objectives (that is, different roles have different sets of objectives) and all roles must have at least one objective. Formally:

$$(1) \forall r_1, r_2 \quad r_1 = r_2 \Leftrightarrow P_{r_1} = P_{r_2}, \text{ and}$$

$$(2) \forall r, P_r \neq \{\}$$

For example, in the Conference Society, the objective of the PC-member role is to review papers submitted to the conference, that is, each enactor of the role PC-member is expected to aim to be in a state where there are review reports for all the papers assigned to that enactor.

Role Sub-objectives. A role objective γ can be further described by specifying a set of **sub-objectives** that must hold in order to achieve objective γ . Sub-objectives give an indication of how an objective should be achieved, that is, describe the states that must be part of any plan that an agent enacting the role will specify to achieve

⁸ Cf. chapter 5 for more details on the domain language specification.

that objective. However, sub-objectives abstract from any temporal issues that must be present in a plan, and as such must not be equated with plans.

Intuitively, sub-objectives are objectives that contribute to the realization of another objective. That is, if $\Pi\gamma = \{\gamma_1, \dots, \gamma_n\}$ is a set of sub-objectives for γ , the realization of all sub-objectives in $\Pi\gamma$ yields the realization of γ . Furthermore, for each objective γ , the trivial set of sub-objectives $\{\gamma\}$ is defined.

For example, sub-objectives of the objective of reviewing papers of the PC-member role are (a) to have read the paper, (b) to have written the review report, and, (c) to have sent the report to the program chairs. How an actor of the PC-member role is going to achieve this, and indeed if she herself will do it (e.g. she can ask a student to read the paper and make the review report) is not, in this situation, a concern of the society.

Role Rights. Role rights indicate the capabilities that actors of the role receive when enacting the role. These are capabilities that an agent usually does not possess but which are inherent to the role. For example, in the conference society, actors of the PC-member role are given the right to access a conference organization system online. The social contract between the agent and the society will indicate how the agent can access such rights. Rights are represented by (atomic) expressions in LCR.

Role Norms. Role norms specify the rules of behavior for actors performing that role, irrespective of the interaction scene. As stated before, norms are necessary because of the assumption of society openness, that is, that agent design is not under direct control of the society. That is, actors are assumed to act independently from society control and therefore can, in principle, reach states which are not desired from the society perspective. Norms allow for such deviations and provide the means to return to more acceptable states. Furthermore, norms can impose constraints on the capabilities of agents pretending to enact the role. For example, a norm of the role of PC-member can state that, in all cases, agents pretending to play that role must be honest. We represent role norms as deontic expressions in LCR. Norms are further described in section 3.4.3.

Role Types. In OperA, roles can be of two different types: **institutional** roles and **external** roles. Actors of institutional roles are fixed and controlled by the society and are designed to enforce the social behavior of other agents in the society and to assure the global activity of the society. External, or operational, roles can in principle be enacted by any agent, according to the access rules specified by the society, and describe the overall (domain related) objectives of the society. Typically, actors of institutional roles are mutually trusted agents, whereas operational role actors do not necessarily trust each other. Moreover, players of institutional roles are unselfish, that is, either they don't have own goals (in the case they have been designed just for the role) or, otherwise, they can never put their own goals first, whereas players of external roles will possibly at each moment decide on the course of action to take and weight the objectives of role against its own goals. The operational layer is always domain and application dependent whereas the facilitation layer depends on the cooperation characteristics of the environment and can be reused across domains. The methodology described in chapter 6, facilitates the design of agent societies, by using

the coordination type of the domain to determine roles and interactions in the society [Dignum, Weigand, 2002]. Different coordination types correspond to specific facilitation roles that can be used as a starting type for the organizational model.

Figure 3-7 shows the specification of the role PC member in the Conference Society example (for reasons of readability, the example assumes the case that a PC member will only review one paper). In the specification of role norms in a role definition, often the subject of a norm is left out, when it refers to the role itself. In the example above the formal specification of the obligation for a PC member to be able to understand English is: OBLIGED(PC-member, understand(English)).

Role: PC Member	
Objectives	paper_reviewed(Paper, Report)
Sub-objectives	{ read(P), report_written(P, Rep), review_received(Org, P, Rep) }
Rights	access-confman-program(<i>me</i>)
Norms	OBLIGED understand(English) IF DONE assigned (P, <i>me</i> , Deadline) THEN OBLIGED paper_reviewed(P, Rep) BEFORE Deadline IF DONE paper_assigned(P, <i>me</i> , _) AND is_a_direct_colleague(author(P)) THEN OBLIGED review_refused(P) BEFORE TOMORROW
Type	external

Figure 3-7: Role definition for PC Member

3.4.1.2 Groups

The basic idea behind the notion of role groups is to provide means to collectively refer to a set of roles. Moreover, groups are used to specify norms that hold for all roles in the group. The elements of a group definition are summarized in Figure 3-8.

Group Definition	
Group id:	A unique name with which to refer to this group
Roles:	A list of role ids, specifying the members of the group
Norms:	A list of normative expressions that apply to this role

Figure 3-8: Elements of a group definition

Members of a group must be existing roles in the society. A basic group does not specify any group norms, and is just an efficient way to refer to a group of roles, for example, when a certain interaction scene (cf. section 3.4.2.1) requires an enactor of one of the roles in the group to be present, without really having to specify which role. A trivial group is *all*, which refers to all roles in the society. However, when norms are specified for a group, these must be consistent with the norms of the roles in the group. Verification of norm consistency is dealt with in chapter 5.

Group	
Group id:	Organizers
Roles:	{PC-Chair, website manager, general chair, local organizer}
Norms:	FORBIDDEN submit_paper

Figure 3-9: The organization group

An example of a group in the conference society is the organization team, referred to collectively as the Organizers, which contains the program chair, the local organizer, the website manager and the general chair. Figure 3-9 illustrates this group.

3.4.1.3 Dependencies between roles

The notion of role is closely related to those of cooperation and coordination. Societies establish dependencies and power relations between roles, indicating relationships between roles. These relationships describe how actors can interact and contribute to the realization of the objectives of each other. That is, an objective of a role can be delegated to, or requested from, other roles. The dependency relation between roles r_1 and r_2 for objective γ of r_1 , represented by $r_1 \underline{\phi}_\gamma r_2$, indicates that objective γ can be passed to r_2 , that is, that r_2 can realize objective γ for r_1 .

Definition 3.2. Role dependency

A dependency relation describes the fact that role r_1 depends on role r_2 to realize its objective γ . $r_1 \underline{\phi}_\gamma r_2$ indicates a dependency relation between roles r_1 and r_2 for objective γ , where $\gamma \in \text{plans}(r_1)$. The relation $\underline{\phi}_\gamma \subseteq R \times R$ is reflexive and transitive.

That is, for all $r_1, r_2, r_3 \in R$:

- $r_1 \underline{\phi}_\gamma r_1$
- $r_1 \underline{\phi}_\gamma r_2$ and $r_2 \underline{\phi}_\gamma r_3$ implies $r_1 \underline{\phi}_\gamma r_3$.

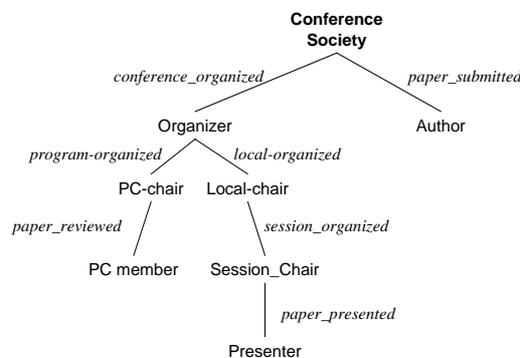


Figure 3-10: Role dependencies in the conference society

In OperA, roles are organized as a pre-order set, represented as $\mathfrak{R} = (R, \underline{\phi})$ that reflects role dependencies. A dependency graph represents the dependency relations between roles. Nodes in a dependency graph are roles in the agent society. Arcs are labeled with the objectives of the parent role for whose realization the parent role depends on the child role. There can be more than one arc between two nodes, representing the fact that the parent role depends on the child role for more than one of its objectives. The root of the graph is the society itself, represented as a super-role,

which contains the global objectives of the society, which are then decomposed into role objectives distributed along the role tree. Part of the dependency graph for the conference society is displayed in Figure 3-10. For example, the arc between nodes *PC-Chair* and *PC-member* represents the dependency $PC-Chair \underset{\text{paper-reviewed}}{\phi} PC-member$.

Dependencies between two roles can be established in different ways, and therefore a model must describe how this interaction occurs. The way the objective γ in a dependency relation $r_1 \underset{\gamma}{\phi} r_2$ is actually passed between r_1 and r_2 depends on the coordination type of the society, which have been discussed in chapter 2 (cf. Table 2-1). In OperA, we therefore identify three types of role dependencies, related to the coordination types hierarchy, market and network. In [Dignum, Weigand, 2002] we analyzed the implications of coordination to the architecture of agent societies, which are summarized in Table 3-1 below.

Table 3-1: Coordination types

	Market	Network	Hierarchy
Type of society	Open	Trust	Closed
Agent 'values'	Self interest	Mutual interest/ Collaboration	Dependency
Facilitation roles	Matchmaking Banking	Gate keeping Registry Matchmaking	Interface Control
Dependency relation	Bidding	Request	Delegation

In hierarchies, the parent role will delegate its sub-objectives to its children. In this case, the enactor of a child role cannot decide which objectives it will get but must accept whichever objectives are delegated to it by its parent role. In markets, the child role can request the assignment of objectives from the parent role. That is, the enactors of a child role can choose to take the objectives of its parent such that it best fit its own private goals. In a network, both situations can happen. An objective can either be delegated by the parent role or requested by the child role, which defines an equivalence relation between related roles in a network. In summary, we identify three types of role dependencies:

1. $r_1 \underset{\gamma}{\phi}^H r_2$, representing the hierarchical relation
2. $r_1 \underset{\gamma}{\phi}^M r_2$, representing the market relation
3. $r_1 \underset{\gamma}{\phi}^N r_2$, representing the network relation

The role dependencies illustrated in Figure 3-10 are therefore interpreted in different ways depending on the coordination type holding in the society. For instance, in the case of a hierarchy, the relation $PC-Chair \underset{\gamma}{\phi} PC-member$, (where γ is *paper-reviewed*) indicates that the enactor of role *PC-Chair* can delegate the objective *paper-reviewed* to an enactor of role *PC-member*, that is, enactors of *PC-member* will get papers to review assigned to them. In a market, enactors of the role *PC-member* can request the objective *review-paper* to the enactor of role *PC-Chair*,

that is PC members can choose which papers they want to review and apply for those to the enactor of role *PC-Chair*. In a network, a dependency relation in a network is a request which can be initiated either by the parent or by the child roles. In some cases PC-Chair will request PC-members to review papers, and, in other cases, PC-members will ask specific papers. Depending on the domain, specific mechanisms for a network dependency relation can be specified.

3.4.2 Interaction structure

As described in the previous subsections, society objectives are realized by the interaction between actors. Roles represent different skills, entities or interests relevant for those global objectives. Activities in a society are the composition of multiple, distinct and possibly concurrent interactions, involving different actors, playing different roles. For each activity, interaction is articulated through **scenes** that follow pre-defined abstract **scene scripts**. A scene script is described by its players (roles), its desired results and the norms regulating the interaction. In the OM, scene scripts are specified according to the requirements of the society. The results of an interaction scene are achieved by the joint activity of the participating roles, through the realization of (sub-)objectives of those roles. A scene script establishes the desired interaction patterns between roles, that is, a desired combination of the (sub-)objectives of the roles. Examples of scenes are the registration of participants in a conference, which involves a representative of the organization and a potential participant, or paper review, involving program committee members and the PC chair.

However, because in organizations more complex activities can take place, scenes must be embedded in a broader context, that allows to represent how the overall society objectives can be achieved. OperA therefore enables the description of ordering and synchronization of interaction scenes. Scene scripts are organized into an **interaction structure** that specifies the coordination of the scene scripts. **Transitions** describe a partial ordering of the scenes, plus eventual synchronization constraints. Furthermore, the enactment of a role in a scene has consequences for the further enactment of roles in following scenes. That is, the **evolution relations** between roles must be described. Evolution relations specify the constraints that hold for the role-enacting agents as they move from scene to scene in the animated society. Note that several scenes can be happening at the same time and one agent can participate in different scenes simultaneously. Transition scripts must furthermore also describe the conditions for the creation of a new instance of the scene script. For each scene, the interaction structure also specifies an upper bound for the number of instances of that scene that are allowed simultaneously. Finally, each interaction structure definition must include the description of the initial and final scenes. Such scenes play a special role in the animation of the society and will be further discussed in sections 3.5.1.1 (start scene) and 3.5.1.2 (final scene). The elements of an OperA interaction structure are described in Figure 3-11.

Scene scripts are described in section 3.4.2.1, transition scripts in section 3.4.2.2 and role evolution relations in section 3.4.2.3. An interaction structure specifies the ‘minimal’ group of scenes that is needed to accomplish society objectives, according to the organization’s requirements. However, when roles are instantiated to concrete

agents, other scenes can be added to the interaction structure in order to meet their specific agreements and requirements. For example, in the conference society the scene ‘fix conference program’ can be added between ‘paper acceptance’ and ‘conference’, which will describe possible interaction between authors and PC-Chairs with the aim to meet author’s requirements relative to the presentation slot of their papers. This dynamic modification of interaction structures is further described in the Interaction Model (cf. section 3.6).

Interaction Structure definition	
Scenes:	A list of scene script definitions
Simultaneous scenes:	A list of pairs (scene identifier, max)
Transitions:	A list of scene transition definitions
Evolution relations:	A set of expressions describing the evolution of roles
Initial Scene:	Name of the initial scene
Final Scene:	Name of the final scene

Figure 3-11: Elements of the definition of an interaction structure

The graphical representation of interaction structures shows scenes as boxes labeled with the scene name, and transitions as directed arcs between two scenes. The initial and final scenes are represented by boxes with a black background. The interaction structure of the Conference society is depicted in Figure 3-12. Forked arcs indicate that all incoming/outgoing scenes must be realized (AND), and directed arcs indicate that either of the paths can be chosen (OR). A numbered circle connected to a scene indicates the upper bound of scene instances. For example, in the conference society, maximal M conference sessions, that is, instances of the conference session script, can occur simultaneously. An empty circle indicates that a new scene instance will be created every time. More details on the relations between scenes are given in section 3.4.2.2.

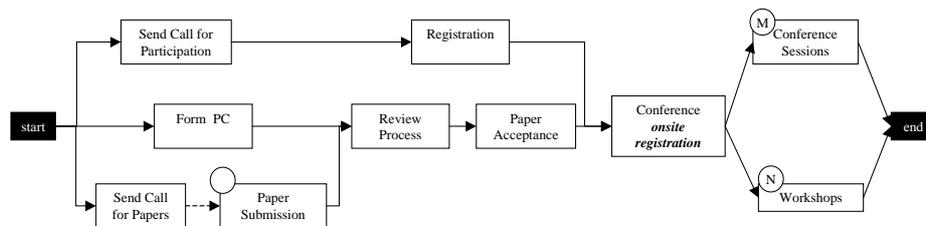


Figure 3-12: Interaction Structure for ‘Conference’ Society

3.4.2.1 Scene script

Interaction scripts serve as a blueprint for the actual interactions between actors. An interaction **scene script** describes a scenario of activity, that is, how roles can interact and evolve in the context of a scene. The elements of a scene script definition are summarized in Figure 3-13.

In general, scripts can be strictly or loosely specified. In a strict specification, the activity of the agents is completely fixed as a protocol, and the agent’s performance

has very little room for variance. Most current MAS are designed in this way, e.g. [Esteva et al., 2002b]. In a loose script specification, interaction is described in general terms and objectives, and agents are free to fulfil their interaction at will. In between these extremes exists a range of possibilities. Landmarks are used in OperA to describe both the objectives (and its patterns) as the norms of an interaction scene. In the following, we described in more detail the elements of a scene script.

Roles. Scene scripts indicate which roles participate in the scene. In many cases, the participants in a scene do not need to enact a specific role but can be one of several roles. These sets are specified in the social structure as groups. When a group is indicated as participant in a scene, actors of any of the group roles are entitled to participate in the scene, given that access conditions are met⁹. Often, the number of actors of a certain role in a scene is not fixed, in which case a minimum and maximum number of actors can be indicated. Moreover, the expression $s.r$ denotes the fact that role r is a role of scene s .

Scene script definition	
Scene id	A unique name to refer with which to refer to this scene
Roles	A list of role or group ids, indicating the participants in this scene, plus minimum and maximum constraints on actor populations per role
Results	A list of landmarks that describe the desired results of this scene
Interaction Patterns	A set of patterns indicating the desired intermediate states possibly partial-ordered for scene objectives
Norms	A list of normative landmarks applying to this scene

Figure 3-13: Elements of a scene script definition

Results. Scene results are declarative expressions that describe the desired final states for the scene, that is, the states in which the scene ends and actors can leave it successfully. This does not mean however, that all actors must leave the scene at the same moment. Since scene results are related to role objectives, when an actor has achieved all the objectives of its role in a scene, and all the commitments expressed in its interaction contract, it can in principle leave the scene, except if explicitly stated otherwise (for example, in the scene norms). The relation between scene results and role objectives can be expressed as:

$$\forall \rho \in results(s), \exists r: s.r, \rho \subseteq (objectives(r) \cup sub-objectives(r))$$

where s is a scene, meaning that every result in a scene must be an objective or sub-objective of one of the participating roles. Note that, in this definition, the inclusion of $objectives(R)$ is redundant, as every objective is also a sub-objective (cf. the definition of sub-objectives in section 3.4.1.1). However, for the clarity of the definition of results, we choose to explicitly refer to role objectives.

⁹ Access to scenes will be discussed in section 3.4.2.2.

Interaction Patterns. An interaction pattern describes the way to achieve a scene result, as desired by the society. Interaction patterns give an indication of how a result can be achieved, that is, describe the states that must be part of any protocol that will eventually be used by actors to achieve that result. Moreover, interaction patterns give an indication of partial ordering by describing temporal relationships between expressions. Furthermore, elements of an interaction pattern, which refer to the activity of an actor, must be of the form $D_r\gamma$, where r is a role and γ a (sub-)objective of r . For example, the interaction pattern specified in Figure 3-14 describes the expected landmarks for the scene script ‘Review Process’. In this case, a paper is to be assigned for review to two enactors of the PC-member role before a certain deadline (DeadlineA), after which, and before another deadline (DeadlineR) the reviews are expected back from the PC-members.

Interaction Scene: Review Process	
Roles	PC-Chair (1), PC-member (2..Max)
Results	$r_1 = \forall P \in \text{Papers, reviews_done}(P, \text{review1}, \text{review2})$ $r_2 = \forall p \in \text{Papers, decision_on_paper}(\text{paper}, \text{decision}, \text{review1}, \text{review2})$
Interaction Patterns	PATTERN(r_1) = { DONE(O, paper_assigned(P, PC1, DeadlineR) BEFORE DeadlineA), DONE(O, paper_assigned(P, PC2, DeadlineR), BEFORE DeadlineA), DeadlineA BEFORE DeadlineR, DONE(PC1, paper_reviewed(P, Rev1) BEFORE DeadlineR), DONE(PC2, paper_reviewed(P, Rev2) BEFORE DeadlineR) }
Norms	PERMITTED(O, paper_assigned(P, PC, DeadlineA)) OBLIGED(PC, paper_reviewed(P, Rev) BEFORE DeadlineR) OBLIGED (O, decision_on_paper(P, D, Rev1, Rev2) BEFORE DeadlineD)

Figure 3-14: Script for Review Process scene

Definition 3.3. Interaction pattern

Let ρ be a result of interaction scene s . An interaction pattern for ρ is a set $\Pi\rho = \{\rho_1, \dots, \rho_n\}$, where ρ_i are temporal achievement expressions such that:

$$\bigwedge_{i=1}^n \rho_i \rightarrow \rho, \text{ where } \rho_i \in \Pi\rho$$

Scene Norms. Scene norms describe the expected behavior of actors in a scene. The collaboration autonomy requirement (section 3.1) and the autonomy of agents are the reason to define scene norms. On the one hand, scene scripts do not uniquely define one single protocol for interaction, and on the other hand, actors are assumed to act independently from society expectations. Norms allow for deviations from expected behavior and provide the means to return to more acceptable states. Norms in OperA are further described in section 3.4.3.

As discussed above, landmarks provide a flexible way to describe interaction. In the scene script illustrated in Figure 3-14, organizational design makes explicit how reviews are supposed to be obtained: that is, first papers must be assigned to PC-members, and then reviews must be received from the PC-members. Actors are in this case still free to decide about how to assign papers (which reviewer gets which paper), how to deal with missing reviews or about the acceptance or not of papers (for instance, do reviewers get to vote about papers or does the program chair decide

alone). Such decisions are to be fixed in the contracts for scene play agreements in the Interaction Model. For example, in the conference society the interaction script for scene ‘Review process’ is partially described in Figure 3-14.

3.4.2.2 Scene transitions

Transitions between scene scripts specify which scenes can be reached from one given scene. Scene transitions describe the synchronization of interaction scenes. The definition of scene transitions is based on **scene connections**, that are 1:1 relations between a source and a target scene specifying the partial ordering of the scenes. Figure 3-15 summarizes the elements of the definition of a scene connection script.

Scene connector definition	
Source Scene:	A scene identifier
Target Scene:	A scene identifier
Departure Landmarks:	A list of landmarks describing departure constraints
Entry Landmarks:	A list of landmarks describing entry constraints

Figure 3-15: Elements of a script connection definition

For each connection, both the source as the target scenes must be existing scenes, specified in the interaction structure. Landmarks describe the conditions under which a scene can end or can start. A special case of such a constraint is the **evolution** of roles, which indicates how roles can move and evolve from scene to scene.

Definition 3.4. Scene connections

A connection between two scenes s_1 and s_2 is represented by the scene transition relation $st(s_1, s_2)$, $st \subseteq S \times S$. The transitive closure of st , $st^+ \subseteq S \times S$, is defined as: $\forall s_1, s_2, s_3 \in S$,

if $st(s_1, s_2)$ then $st^+(s_1, s_2)$

if $st(s_1, s_2)$ and $st^+(s_2, s_3)$ then $st^+(s_1, s_3)$

Transitions between scenes are 1:M or N:1 relations between scenes, that is, each source scene can be connected with several target scenes and each target scene may be reached from several source scenes. Scene transitions specify a network of scenes. In the case that a source scene is connected to many targets, or many sources are connected to the same target, it is necessary to describe what is the relationship between sources and targets. Furthermore, if several instances of a scene are possible, it must be indicated whether a new instance is to be created. That is, scene transitions synchronize sources and targets. We identify the following types of transitions, depicted in Figure 3-16:

1. **All-targets:** Specifies an AND relation between a set of source scenes and a target scene.
2. **Some-targets:** Specifies an OR relation between a set of source scenes and a target scene.
3. **One-target:** Specifies an exclusive OR relation between a set of source scenes and a target scene.

4. **New-target:** Indicates that a new instance of the target scene must be initiated
5. **All-sources:** Specifies an AND relation between a source scene and a set of target scenes.
6. **Some-sources:** Specifies an OR relation between a source scene and a set of target scenes.
7. **One-source:** Specifies an exclusive OR relation between a source scene and a set of target scenes.

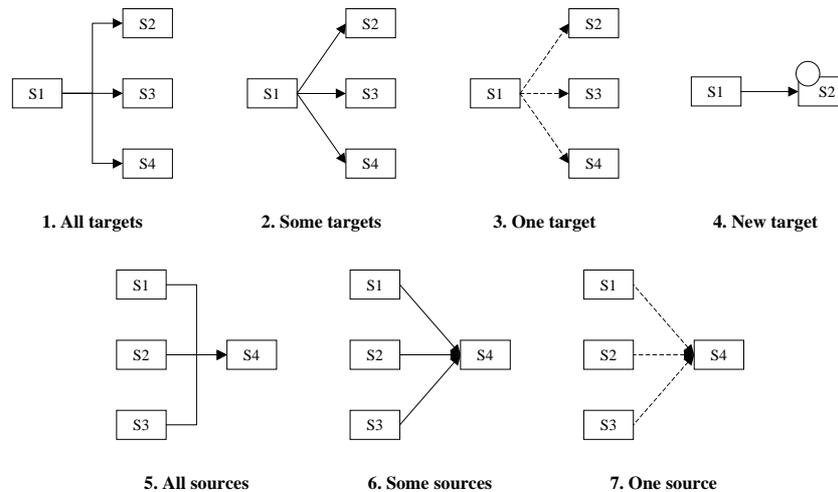


Figure 3-16: Scene transitions

The elements of the definition of a scene transition are summarized in Figure 3-17.

Scene transition definition	
Transition:	Transition identifier
Connectors:	A set of connector definitions
Type transition:	A list of landmarks describing entry constraints

Figure 3-17: Elements of a script connection definition

3.4.2.3 Role evolution relations

Society roles can evolve into other roles as a consequence of the activities of the enacting agents. That is, the role(s) an agent plays influence the possibilities for further action of that agent in the society. Roles change into other roles as a consequence of the results of an interaction scene. For instance, in the conference society, an actor that performs the role of ‘PC-member’ in the ‘Review Process’ scene can evolve into the role of ‘Session-chair’ in a ‘Conference Session’ scene.

Evolution relations between roles describe necessary, sufficient and conflict situations between roles. Informally, a **necessary** evolution relation specifies that the enactment of a role in a previous scenes implies the enactment of a given role in a following scene. A **sufficient** relation specifies which role must have been enacted in

a previous scene in order to be able to enact a given role. Finally, a **conflict** enactment relation indicates that two roles cannot be enacted simultaneously by the same actor. In the following we will describe and give examples of these different relations.

Figure 3-18 depicts a scene connector between the Registration and the Conference scenes. The label *organizer* in this connector represents the case of the trivial relation, that is, the rea of *Registration.organizer* will enact the role *Conference.organizer*. The other label indicates that only role-enacting agents (reas) of the role *Registration.applicant* are enabled to enact *Conference.participant*, given that expression *paid-fee* holds. This relation is of type *sufficient*, indicating that applicants do not have to enact the participant role (one can always decide not to attend after all)¹⁰.

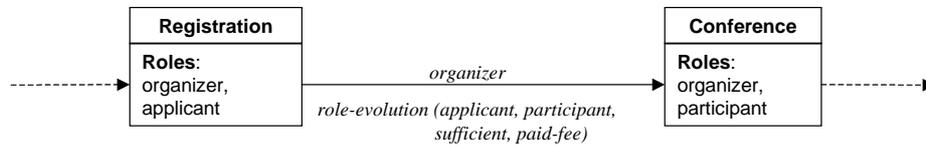


Figure 3-18: Scene connector example

A **conflict** between two roles in a scene means that an agent playing role r_1 cannot play role r_2 in the same scene. This definition of conflict is local to a scene. However, there may be situations in which global conflict must be specified, that is, if an agent enacts a given role in a scene, that same agent cannot enact another given role in any other scene.

Definition 3.5. Conflict relation

Given $s \in S$, and roles r_1, r_2 such that $s.r_1, s.r_2$:

1. A local conflict relation is represented by $r_1 \otimes_s r_2$
2. A global conflict between roles in different scenes is represented by $r_1 \otimes r_2$

The relation $\otimes_s \subseteq R \times R$ is symmetric, that is, $\forall r_1, r_2 \in R, r_1 \otimes_s r_2$ implies $r_2 \otimes_s r_1$. For example, the fact that the chair of a session cannot be a presenter in that same session is represented by the local conflict relation $\text{chairman} \otimes_{\text{session}} \text{presenter}$.

3.4.3 Normative structure

Agent societies need mechanisms to inspire trust into the agents that will join them. The field of ethics involves systematizing, defending, and recommending concepts of right and wrong behavior. A part of the design process of an agent society must therefore be the specification of the ethics of the society. One way to inspire trust in

¹⁰ An example of a necessary role evolution can be found in the Wedding society, where a rea of *Ceremony.bride* must enact the role of *Married-life.wife*, given that a condition *said-yes* holds.

the parties is by incorporating the regulations (**norms**) in the society architecture that indicate the type of behavior expected from role enactors. Deontic logic provides a formal basis to incorporate norms into agent theory [Meyer, Wieringa, 1993]. Using deontic logic, norms are translated into obligations, prohibitions and permissions. Furthermore, verification is possible using formal theorem provers.

According to sociology, a norm is a rule or standard of behavior shared by members of a social group (Encyclopedia Britannica). According to philosophy, a norm is an authoritative rule or standard by which something is judged and on that basis approved or disapproved (Columbia Encyclopedia). Examples of norms include standards of right and wrong, beauty and ugliness, and truth and falsehood. According to economics, a norm (from *norma*, Latin for carpenter's level) is a model of what should exist or be followed, or an average of what currently does exist in some context, such as an average salary among members of a large group [Dignum, 2002a].

Norms in OperA must capture the abstract norms and values that hold in the domain. The norms that govern an organization define the rights and obligations of the agents in the organization, related to the roles they play, or to a particular area of activity. In order to be incorporated in the model, domain norms are made concrete and explicit using the concepts defined in the model's communicative structure (ontology and communication framework, cf. section 3.4.4).

Many attempts have been made to formalize the specification of norms, of which deontic logic is one of the most successful [Meyer, Wieringa, 1993]. However, more work is needed on how to specify norms that hold in the context of a society, and at which level such specification should be done. In [Dignum, 2002a], F. Dignum argues that the level at which norms are specified is more abstract than the level at which the processes and structure of an organization are described and therefore there is a need to describe norms at a level that directly can be used in the institution. For example, a norm that says '*it is forbidden to discriminate on the basis of age*' can be straightforwardly represented in deontic logic as $F(\text{discriminate}(x, y, \text{age}))$, stating that the discriminate action is not allowed for agents. However, it is unlikely that agents operating in that organization will explicitly have such action available. It is therefore needed to indicate how the intended result (i.e. no discrimination on age) can be achieved in terms of the ontologies that hold in the organization. It is not our intention to further develop these aspects in OperA, but we assume that norms are already translated to the society environment.¹¹ For a discussion on the incorporation of abstract norms in electronic institutions, such as agent societies, we refer to [Dignum, 2002a].

In the OM, norms are specified as LCR expressions, and are part of the specification of roles, scene scripts and transaction scripts¹². Norms in OperA reflect a

¹¹ See chapter 6 for more on norm elicitation.

¹² A formal description of OperA norms is given in chapter 5.

translation of the abstract norms in the domain, and are therefore dependent on the choices made by society design.

Role norms: indicate the rules of behavior for actors performing a role irrespective of the interaction scenes it participates in. For example, a norm of the role of `PC-member` can state that in all cases, agents pretending to play that role must be honest. Another example is that an agent assuming the role of PC member is permitted to access the online conference management server.

Scene norms: describe the expected behavior of actors within an interaction scene. For example, in the ‘Review process’ script described above, an actor playing the `PC-member` role is obliged to provide the reviews of the papers assigned to him, to the program chair before a given deadline.

Transition norms: impose additional limitations to actors attempting to follow that arc between two scenes. For instance, in the example above, it can be imposed that the role transformation `PC-member` \Rightarrow `Session-Chair` can only occur when the actor playing `PC-member` in the ‘Review Process’ scene fulfilled all its obligations (that is, did provide the reviews before the deadline).

All norms are indexed with a role, but can be defined either in the definition of the role, in a scene script where the role participates, or in a scene transition that defines an evolution from the role. Norms identify classes of constraints over the subject of the norm. The specific way norms are made operational in a particular society instance, must finally be described in the social and interaction contracts between the agents that will enact the society roles. Norms are defined in OperA as follows:

Definition 3.6. Norm

Given a set of domain variables and functions defined in the communicative structure, the set of domain terms T_D is defined as usual. $Pred_N \subseteq T_D$ is the set of predicates that constitute normative expressions. A language for norms of role r is defined as

$$\varphi ::= O_r\varphi \mid P_r\varphi \mid F_r\varphi$$

where $O_r\varphi$, $P_r\varphi$ and $F_r\varphi$ indicate the obligation, permission and prohibition for role r to see to it that φ holds, respectively.

For example, an obligation for the `PC-member` role to provide a review R for paper P before deadline D in the Review process scene script, is represented in LCR as $O_{PC_member}(receive_review(PC_chair, P, R, PC_member) \leq D)$, and in the syntax of OperA as:

OBLIGED(PC-member, receive_review(PC-chair, P, R, PC-member) BEFORE D)

Norms that are defined for a role, will hold for all enactors of that role.

Norm inheritance: by defining groups of roles we can specify collective norms that apply to all members of a group. In any society, there is by default one group, that of all roles in the society. Society norms apply therefore to all roles, at all times. Moreover, norms specified for a group must be consistent with the norms of all roles in the group (cf. chapter 5, section 5.6).

3.4.4 Communicative structure

Interaction in OperA is represented as communication between the interacting actors¹³. The aim of the Communicative Structure component of the Organizational Model of OperA is therefore to describe the communication primitives. That is, to describe the set of performatives and the domain language specific to the society, to be used in the communication between role enacting agents¹⁴.

Any mechanism for communication must include both a **knowledge representation language** (to describe knowledge about the domain) and a **communication language** (to specify the interactions among agents). Knowledge representation models are based on *ontologies* that define the model and vocabulary for a particular domain of discourse. An Agent Communication Language (ACL) provides the language primitives that enable communication. ACLs are commonly thought of as *wrapper languages* in that they implement a communication protocol that is unaware of the choice of content language and ontology specification mechanism. One of the cornerstones of OperA is the openness of societies to heterogeneous actors, which use different knowledge representation languages. However, in order to allow agents to communicate successfully, a shared ACL (Agent Communication Language) is assumed. As postulated in speech act theory [Searle, 1969] and used in most agent communication languages, agent illocutions are not just propositions that may be true or false, but speech acts that may succeed or fail. A possible classification of speech acts, according to the philosophy of language, is:

- *Representatives of assertives*: represent a state of affairs
- *Directives*: order or ask the recipient to do something
- *Commissives*: represent a commitment of the speaker
- *Expressives*: express a certain mental (emotional) state
- *Declaratives*: Bring something about in the world
- *Permissives*: Allow for action
- *Prohibitives*: Ban some action

Many efforts have been done to specify and study communication between agents, an overview of which can be found in [Chaib-Draa, Dignum, 2002], and to develop content ontologies to support communication between agents [Cranefield et al., 2002], [Fensel, 2001]. Current ACLs, such as FIPA ACL [FIPA, 2002], or KQML [Finin et al., 1997], provide a catalogue of general purpose performatives with a, more or less, formal semantics. Recent work stresses furthermore the importance of relating these

¹³ In reality, interactions that involve the use of (scarce) resources are usually not of communicative nature. However, in OperA the communication of the fact that an actor is using a scarce resource, is used as necessary and sufficient representation of the act itself.

¹⁴ OperA assumes that the communicative framework will be used by all participating agents. How this can effectively be done is a complex issue that will not be pursued in this thesis.

communicative primitives to the organizational model of the agent society [Singh, 1998], [Serrano, Ossowski, 2002].

Besides communicative acts, OperA also provides means to represent concepts and relationships in the domain (a domain ontology). Ontologies must be rich enough to *cover* the domain of application in order for actors in the society to be able to interact in several contexts, but, on the other hand, keep in mind the *relevance* of the concepts, since a voluminous ontology can lead to inefficiency because agents will spend too much time finding the relevant concepts and meanings. Rather, ontologies should be *extensible*, that is, allow to add new elements. An example of an ontology used in an OperA model can be found in chapter 7, section 7.2.1.1.

In section 3.4.2, we described the relation between roles and interaction scenes in OperA. Setting out from the work of Serrano and Ossowski [Serrano, Ossowski, 2002], we will now relate these elements to the communicative actions in an agent society. The basic idea is that the objectives of a role are related to **communicative acts** (CA) that manipulate phrases of the domain language. Communicative acts are the basic building blocks of conversations and have a well-defined semantics in speech act theory, which is independent of the content of the action [Searle, 1969]. Both FIPA ACL as KQML are current efforts to define standard collections of CAs for agents. From a communicative perspective, scene scripts represent abstract conversations, composed by CAs involving the roles that participate in the scene. In the Interaction Model (cf. section 3.6), the issue of how communication between agents will fulfil this scheme and provide actual interaction will be further discussed.

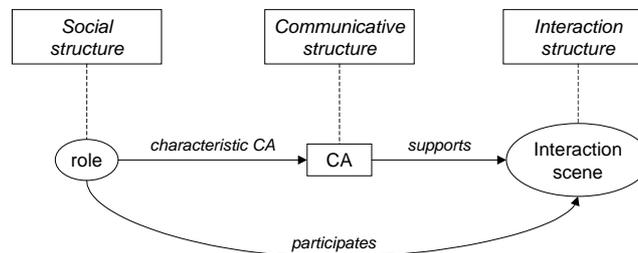


Figure 3-19: Communicative acts as links from roles to scenes

Figure 3-19 describes the relations between the social, interaction and communicative structures of the OM¹⁵. In the Social Structure, the characteristics of roles are described. The (sub-)objectives of a role, for the achievement of which the role depends on other roles, correspond to the content of the CAs that the role can perform. The ontological and performative characteristics of those CAs are specified in the Communicative Structure. Finally, in the Interaction Structure, the execution of interaction scenes is based on those communicative acts, in the sense that a scene describes a desired conversation, that is, a combination of CAs from different roles,

¹⁵ Adapted from the RICA model [Serrano, Ossowski, 2002].

between the participating roles. In other words, objectives of a role for which there is dependency relation to another role, are achieved through the execution of an interaction scene. Such interaction scenes represent an abstract conversation constructed from the CAs available to the participating roles.

Communicative acts that represent the different speech acts introduced above, are defined as follows:

Definition 3.7. Communicative act

Given roles r and s and expression φ , a communicative act is defined as $CA(s, r, \varphi)$. Role s is said to be the sender, role r is the receiver and φ is the content of the CA. Furthermore, s and r can also refer to role groups.

In OperA, the basic communicative acts *request*, *inform*, *commit* and *declare* are defined (cf. chapter 5, Definition 5.27). Note that not all objectives or sub-objectives of a role have to have a corresponding CA. However, (sub-)objectives which are part of interaction patterns of scene scripts must have corresponding CAs. Moreover, since role objectives and scene script results are specified as states of affairs to be achieved in the world, their communicative counterparts are of the declarative type. For example, in the conference society, the role of PC-member has the objective *paper-reviewed(Paper, Report)*. One of the sub-objectives of *paper-reviewed* is *review-received(PC-Chair, Paper, Report)* which corresponds to the an informative CA between the PC Member role and the PC-Chair role:

inform(PC-Chair, PC-member, review-received(PC-Chair, Paper, report)).

which means that role *PC-Chair* informs role *PC-member* that the review *Report* for paper *Paper* has been received by *PC-Chair*. The expression *review-received(PC-Chair, Paper, Report)* is defined in using the domain language and the meaning of its components (e.g. *Paper* or *Report*) is defined in the ontology.

Communicative Structure definition	
Ontology:	The content vocabulary used in the society
Content language:	The building rules for propositions over the content
Role illocutions:	A set of pairs (role-id, illocution), the illocutions of each role
ACL:	The agent communication language used

Figure 3-20: Elements of the definition of a communicative structure

Based on [Esteva et al., 2002a], we propose the following communicative structure, which defines the communication language, the domain representation language and the ontology used in the society. The formal specification of OperA will give a formal semantics to communication acts. The elements of the communicative structure are described in Figure 3-20.

3.5 Social Model

*'All the world's a stage,
And all the men and women merely players:
They all have their exits and entrances;
And one man in his time plays many parts.'*
– W. Shakespeare, *As You Like It*, Act II, Scene 7¹⁶.

The Organizational Model of OperA, introduced in section 3.4, describes the society design from the perspective of the organization. In the **Social Model (SM)**, the activity of independent agents in the society is specified. The central component of SM is the **agent**. In OperA, agent and role are fundamentally different concepts. Roles describe the organizational perspective on individuals, whereas agents represent the perspective and objectives of the individuals themselves. This difference is not always present in agent literature where often a role represents a kind of abstract agent or a class of agents. In our work, agent are executable entities, implemented in some language, and eventually have an operational semantics. Roles, on the other hand, are declarative entities meant to represent a part of the organization's design and can be taken up by the agents enacting the role. So, a role only gets an operational semantics indirectly through the agents that take up that role.

In OperA, an agent is assumed to be an autonomous, socio-cognitive entity capable of individual social behavior. As defined in chapter 2 (section 2.3.1.) *socio-cognitive* entities are not only endowed with mental attitudes towards the environment, but also assume other agents to have a similar attitude [Dennett, 1987]. Socio-cognitive entities are socially able, that is, have the ability to interact and cooperate with others. For an more extended discussion on agenthood, we refer to chapter 2. OperA makes no further assumptions on the internal architecture and design of individual agents. This enables the modeling of open agent societies, where often no knowledge is available about the specific architecture of intervening agents.

Following its own goals and interests, an external agent may seek to enact roles in a society. That is, in OperA, by joining a society, the agent will take up a role defined in the society's organizational model. A rational agent will want to enact a role whose objectives somehow contribute to its own goals. We further assume that a rational agent will try to achieve an as optimal as possible solution for its own goals, and act as much as possible according to its own characteristics. These may or may not, be different from the expectations the society has on the role. For example, in the conference society, a particular agent that will enact the PC member role may negotiate that she will only review two papers, instead of the 5 society design had in mind for PC members. Also, a group of enactors of the Participant role may ask the society to provide them a room to hold a meeting of a project they are all working on, during the course of the conference. This is a private objective of the agents which does not contribute to the society's objectives, but which may be agreed upon, if it

¹⁶ Quoted in [Odell et al., 2003]

causes no inconvenience to the society. Therefore there is a need to explicitly represent how the agent will enact the role and which (other) actions the agent is allowed or not in the society. In section 3.5.2 we discuss role enacting agents in more detail.

Participants are admitted to the society through a process of socialization, during which the participant negotiates with the society (represented by an institutional agent, for example, a gatekeeper) the terms and conditions of its participation. This process can be more or less free, depending on the type of society. On one extreme, open societies will allow most agents to join, whereas closed societies will not allow external agents to join, and therefore there will be no socialization process, as all agents will be a 'fixed' part of the society.

For each agent, the SM reflects the agent's own requirements and conditions concerning its participation in the society. Depending on the complexity of the implemented agents, the negotiation of such agreements can be more or less free. However, making these agreements explicit and formal, allows the verification of whether the animated society behaves according to the design specified in the OM. The SM specifies a population of agents in a society, and as such can be seen as an instantiation of the OM describing a society. When all roles specified in the OM are instantiated to agents in the SM, we say that the SM provides a **full** instantiation of the society, otherwise, it is a **partial** instantiation.

In principle, each external agent will negotiate its participation in the society, and agree on the duration of that participation, resources and communication capabilities, monitoring and sanctions, etc. Even when actual negotiation is not possible or necessary, the SM represents each agent by the (agreed) expectations of its behavior as enactor of a role. In the most simple case, this can be done by ignoring any abilities of the agent and imposing on it a rigid interface with the society that enforces desired behavior [Vasconcelos et al., 2002]. However, if the society wants to be able to profit from the individual capabilities of the participating agents, a more flexible mechanism is needed that, on the one hand, guides the agent on its activity within the society (it knows what is expected of it and what are its rights and norms), and on the other hand, enables the verification or prediction of the society behavior. Remember that, given the heterogeneity assumption, the internal architecture and capabilities of the agents are not externally known and therefore some explicit representation expectations on the behavior of individual participants is needed. In OperA, the concept of **social contract** is used to explicitly represent expectations on the behavior of agents.

In other words, the use of social contracts to describe agents' behavior in a society is motivated as follows. The central assumption of OperA is that society and agents have different interests and objectives and therefore must be specified independently from each other. The characteristics of the society as desired by the society designers are specified in the OM. Agents are assumed to be heterogeneous both in design and in objectives. Agents have their own plans on own to achieve their goals. Such plans are assumed to be private. According to such a plan, an agent may need to join an existing society (for example, an agent whose goal is to buy a computer, may join an auction house society in order to do so). However, the agent is not concerned with the

reason and requirements that lead to the specification of the society in the first place and whether the society has any objectives of its own.

The basic component of a social model is the **social contract**. Social contracts provide the link between agents and roles by describing explicitly the agreements for role enactment. The architecture of SM consists of a set of role-enacting agents, described by their social contracts, as described in Figure 3-21. Depending on the society design, contracts can give more or less freedom for the incorporation of agents' own goals and/or plans in the society activity. Social contracts are in principle negotiated for external roles as the enactment of facilitation roles is usually provided by agents controlled by the society, and follows a trivial contract¹⁷.

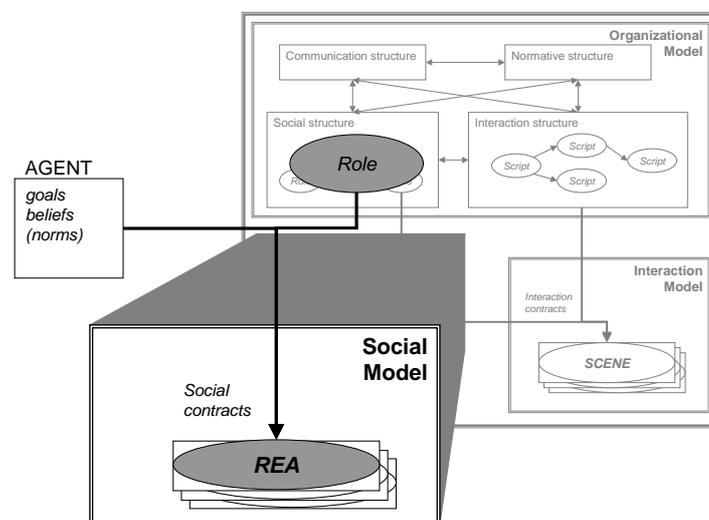


Figure 3-21: Architecture of SM

Social contracts enable to predict the behavior of the society, even when external agents exhibit autonomous behavior. The actual behavior of the society emerges from the goal-pursuing behavior of the individual agents within the constraints set by the OM. Therefore, social contracts allow to integrate the top-down specification of organizational structures, described in the OM with the autonomy of participating agents. On the one hand, social contracts allow for the verification of role enactment, that is, whether the agent did enact the role as agreed and enables actors to know how to proceed when a norm is violated. On the other hand, it makes explicit the expected behavior of an agent while enacting a society role, and therefore provides an 'interface' for other actors to know what to expect of this actor. That is, since in open

¹⁷ Trivial agreements are agreements that exactly match the role specification, and incorporate no own goals or constraints from the agent. In section 3.5.1, trivial contracts will be discussed in more detail.

societies there is no knowledge available about the internal structure and goals of participating agents, social contracts provide an uniform ‘interface’ to an agent. Thus, coordination and verification of society design is possible.

Finally, a formal language for specification of those contracts is needed in order to be able to formally evaluate and verify agent interaction. The Logic for Contract Representation (LCR) is a possible formalism for representation of contracts [Dignum et al, 2003]. LCR is introduced in chapter 4. Given an OM and a set A of agents, we define a social model as a set of social contracts mapping agents in A to roles in OM.

3.5.1 Social contracts

As discussed above, in OperA agents are considered to be black boxes, whose internal architecture and inner motivations are unknown. Social contracts are the means to specify the ‘coordination’ between agent and role. In a generic way, a contract is a statement of intent that regulates behavior among organizations and individuals. The use of contracts to describe activity of a system allows on the one hand, for flexibility in the balance between organizational aims and agent desires and on the other hand for verification of the outcome of the system. Because agents are viewed as black boxes, the contract specification must make no demands or assumptions on the architecture of individual agents participating in the society.

A **social contract** describes the conditions and rules applying to an agent enacting role(s) in the agent society. Informally, social contracts must specify the activity of agents as enactors of society roles, and include aspects such as the specification of the role(s), the time period the contract holds (either in absolute terms: from date to date, or in relative terms: until certain states hold), specific agreements and conditions governing the role enactment, and the sanctions to take when norms are violated (especially if specific sanctions are agreed upon).

Definition 3.8. Social Contract.

*Given a society S , a **social contract**, SC , is defined as a tuple $SC = (a, r, CC)$ where a is an agent, $r \in \text{roles}(S)$ is a role, and CC is a set of contract clauses. A social contract is denoted by $\text{social-contract}(a, r, CC)$.*

A contract clause is a deontic expression that describes conditions and deadlines for a specific obligation, prohibition or permission on the activity of the agent as enactor of the role. Contract clauses are formally specified as deontic expressions of LCR, the Logic for Contract Representation presented in chapter 4. A special kind of social contract is the **trivial social contract**, which does not specify any clauses. Such social contract indicates that role enactment follows exactly the description of the role in the OM, that is, the agent does not require any deviations to the expected behavior of the role. Furthermore, each agent can have simultaneously more than one social contract with the society, describing all the roles it enacts. For an example of a social contract, we refer once again to the Conference Society. Imagine that agent Anne will take the role of PC-member. The trivial contract for this enactment is:

social-contract(Anne, PC-member, {})

in which case agent Anne will assume the role of PC-member, following the description of the role exactly. A more complex social contract is:

*social-contract(Bob, participant, {IF not-used(room) THEN
PERMITTED(Bob, use-conference-room(project-meeting)})*

which describes the case that agent Bob will enact the role of participant and extends the role definition to describe that Bob is allowed to use the conference room for a project meeting when that room is not being used.

We say that a social contract defines **role-enacting agent** relations for all scenes where the role participates. Role enacting agents, or **reas**, are represented by $rea(a, r, s)$, where a , r and s are the agent, role and scene identifiers respectively.

Finally, we must remark that social contracts are abstract descriptions of the results and behavior expected from role enacting agents. Social contracts are not active entities. That is, the agents must incorporate the behavior described in the contract into their own activity in order for the society to be animated. OperA assumes that agents will assimilate social contracts into their architecture. How this is actually done is however not the concern of OperA. However, the animation of OperA societies requires that there must be ways to describe the process of mapping a role description r to an agent a . This meta description must be independent from agent architectures, languages and models. In section 3.5.2 we present some research towards the development of models to describe role enactment.

Definition 3.9. Role-enacting Agent

Given a society S and a social contract $C = (a, r, CC)$, then $\forall s \in \text{scenes}(S)$, such that $r \in \text{roles}(s)$, the role-enacting agent relation $rea(a, r, s)$ is defined, indicating that agent a enacts role r in scene s .

Social contracts identify uniquely role-enacting agents, which abide by the clauses expressed in the contract. The expression $rea(a, r, s)$ is a reference to such a role-enacting agent.

3.5.1.1 Setting up social contracts

The main objective of the SM is to establish social contracts describing role enacting agents in the society. The main concern of OperA is the specification of such contracts. However, a practical implementation of the model will require the description of how such contracts are created and managed. The basic idea is that agents that apply to be admitted in a society should be able to negotiate their participation in order to accommodate their own desires and requirements and, on the other hand, allow for the society to profit from their specific capabilities. In the same way, society design determines how much is fixed in the role definition, and how much freedom agents will have to decide on how to enact the role. Furthermore, a society must be able to describe its possibilities to potential applicants. Depending on

the degree of freedom that a society allows for its participants and on the reasoning capabilities of the agent, role negotiation can be a very complex process. In the following we present some guidelines to the functionality required to enable the setting up of social contracts.

Interaction Scene: PC-member role enactment	
Roles	Society keeper (SK), applicant (A), society register (R)
Results	$\rho = \text{contract}(A, \text{PCmember}, \text{SocialContract})$
Plans	$\Pi\rho = \{ \text{agreed}(\text{max-papers}(M)) \text{ AND } \text{agreed}(\text{review-deadline}(D)) \text{ BEFORE}$ $\text{contract-agreed}(\text{SK}, A, \text{social-contract}(A, \text{PCmember}, \text{CC})) \text{ BEFORE}$ $\text{contract-registered}(\text{R}, \text{social-contract}(A, \text{PCmember}, \text{CC}))$ $\}$
Norms	PERMITTED(SK, negotiate-social-contract(A, PCmember)). OBLIGED (SK, role-description-announced(role(PCmember))).

Figure 3-22: Negotiation of role enactment as interaction script

The negotiation of social contracts is specified in OperA as a special interaction scene script in the OM. This scene script describes the possible negotiable aspects (deadlines, results, capabilities) and generates a social contract describing the activities of a role enacting agent. For example, in the conference society, the informal description of the interaction script for the negotiation of the enactment of the PC member role by an applying agent is illustrated in Figure 3-22. As all other scripts, a social contract negotiation script can provide more or less interpretation freedom to the agents through the level of specification described by its landmarks.

Scripts for the negotiation of social contracts are not usually explicitly part of the society's interaction structure, but are part of the 'start' scene which is part of every interaction structure. The society can also specify roles whose objective is to control the execution of social contracts (such as the role 'society register' in the example in Figure 3-22). The actual verification of social contracts depends on the characteristics of such controlling roles.

3.5.1.2 Ending social contracts

Social contracts describe the results expected from role enacting agents and end when those results have been achieved. Role enacting agents are expected to remain in the society as long as their contracts hold. However, social contracts can describe conditions under which the agent can dissolve the contract and depart from the society. That is, when a role enacting agent wants, or has, to leave the society, it must be checked whether its holding contract(s) allow for premature departure.

The ending of social contracts is specified in OperA as a special interaction scene script in the 'end' scene described in the OM. This scene script describes the possible ways an agent can depart from the society. As all other scripts, a departure negotiation script can provide more or less interpretation freedom to the agents through the level of specification described by its landmarks. Scripts for the ending of social contracts are part of the 'end' scene of an interaction structure. As for the negotiation of social contracts, the society can also specify roles whose objective is to verify whether contracts are correctly ended and agents leave the society in an allowed fashion. Furthermore, contract-ending scripts should also describe the possibilities to dispose

of social contracts with agents that have not taken up their roles (e.g. a rea that never realized its tasks nor reacted to any message).

3.5.2 Role enacting agents

As stated before, in open agent societies one of the most important challenges is the specification of mechanisms through which prospective participants can evaluate the characteristics and objectives of society roles, in order to decide about participation. Currently, in most MAS agents are simply designed from scratch so that their behavior complies with the behavior described by the role(s) it will take up in the society. Comprehensive solutions for this point require complex agents that are able to reason about their own objectives and desires and thus decide and negotiate their participation in a society. A first step on the road to this solution (cf. [Dastani et al., 2003]) is to have a formalism to compare the specification of agents and roles and determine whether an agent is suitable to enact a role.

3.5.2.1 Relationship between role expectation and agent behavior

An important aspect concerning role enacting agents is that of modifying the agent to include the characteristics of the assumed role(s). A possible solution for this point has been proposed in [Vasconcelos et al., 2002], which extends agents with an interface to the society. This interface prevents any action not allowed by the role definition. However, it does not ensure the proactive behavior expected from the role and is not flexible enough to incorporate different enacting styles, capabilities and requirements of the agents. It actually makes the actual agent 'invisible' to the society and only its enactment of the role behavior apparent in the society. We think that the consequence of an agent adopting a role is more drastic than this. The actual agent behavior must often be modified according to the goals, norms and rights specified by the role.

In the following, we assume that agents have goals of their own, and are able to form (either by design or by deliberation) plans to achieve those roles¹⁸. These assumptions are consistent with the agent view described in the beginning of section 3.5 and can be seen as a 'minimal' agent definition. OperA describes agent societies from an global perspective, rather than from the perspective of the individual agents. Even though agents will take many roles simultaneously and along their life cycles, from the perspective of the society, each rea is a different individual. From the perspective of an OperA society it is furthermore up to the agent how to manage and prioritize its goals. That is, by assuming a role, the agent will receive the objectives from that role. How the agent will handle those objectives, whether it interprets them as goals or as norms, what priority it gives them, is up to the agent self. However, the society model is based on the assumption that agents that take up roles are expected to eventually realize the assumed objectives.

¹⁸ We are however not concerned with how agents get those goals. It may be by design, or by participation in other societies.

Nevertheless, societies are concerned with judging the attitudes of the different reas and how those will affect the performance of the role. That is, the society will not look at the agent as a whole but at how a certain rea is acting. Agent literature discusses extensively different types of social attitudes of agents: selfish, altruistic, honest, dishonest, etc [Castelfranchi, 1995], [Sichman, Conte, 1998], [Miceli et al., 1996], [Kalenka, 2001], [Huhns, Abdulla, 1999]. Different types of agents result in different role performances, because the way an agent will plan its goals, which is dependent on his social attitude, influences the realization of its role objectives and the fulfillment of the role norms. For instance, some types of agents will only attempt to achieve the goals of their adopted roles and forget their own private goals, while others will only attempt to achieve the goals from the role after all their own goals have been satisfied. Furthermore, the relations between agent plans and role objectives, and of agent goals and role sub-objectives must be considered, as well as the influence of the role norms on the behavior of agents. We apply these same concepts to the performance of reas in order to evaluate the different styles of role performance¹⁹.

Concerning the goals of the agent and the objectives of the role, the following basic types of role enactment by the agents can be distinguished [Dastani et al, 2003]:

1. **Social enactment:** The agent includes as many of its own goals as possible, but gives priority to the objectives of the role over its own.
2. **Maximally social enactment:** The agent only uses the objectives from the role and ignores its own goals, for the duration of the role enactment.
3. **Selfish enactment:** The agent includes as many of its own goals as possible and gives priority to its own goals over the objectives of the role.
4. **Maximally selfish enactment:** The agent only uses its own goals, and ignores any objectives of the role.

Concerning the effect of agent plans on role objectives, and of role sub-objectives on agent goals the following types of role enactment by the agents can be distinguished [Sichman, Conte, 1998]:

5. **Role enrichment by personal plans:** When the role has an objective for which no non trivial sub-objective set is specified, and the agent playing the role has a plan on how to achieve that role objective. This shows a good adequacy of the player to the role, from the society viewpoint (the actor really adds something to the society activity).
6. **Personal enrichment by role playing:** Dual to the previous. The agent did not have a plan on how to achieve one of its goals, which is provided to it by playing the role. In this case, it is the agent that profits from the society.

¹⁹ These are ideal characterizations, in reality role enactment will combine one or more of these types.

7. **Increasing power by role playing:** The role provides the actor with a ‘better’ way to realize its goals, for instance, the rights associated with the role enhance the activity of the agent.
8. **Diminishing power by role playing:** The society limits the possibilities of activity of the agent. Some plans the agent may have had are not applicable within the role description.

The relationship between agent plans and role sub-objectives is similar to that of agent goals and agent objectives:

9. **Social planning:** The agent gives priority to the sub-objectives of the role over its own plans, which are only used if role sub-objectives cannot be used.
10. **Maximally social planning:** The agent only uses the sub-objectives from the role and ignores its own plans, for the duration of the role enactment.
11. **Selfish planning:** The agent includes as many of its own plans as possible and gives priority to its own plans over the sub-objectives of the role.
12. **Maximally selfish planning:** The agent only uses its own plans, and ignores the sub-objectives of the role.

Finally, we must look to the ways role norms may affect the behavior of the agent playing the role. Since at this stage, we do not consider the case of agents that have norms of themselves, we do not evaluate how agent norms affect expectations on the role. The case of normative agents will be the subject of further research.

13. **Limitation of personal goals:** The role norms cause that some of the goals of the agent cannot be achieved within the society.
14. **Extension to personal goals:** The role norms create some extra goals for the agent, which it did not have previously.
15. **Alteration of personal plans:** The norms of the role cause an alteration on the plans of the agent; these must be extended with extra activity, some of the actions cannot be taken or the order in which actions were planned is not allowed.

Table 3-2: Relation between role and agent behavior

		Role		
		Objectives	Sub-objectives	Norms
Agent	Goals	objectives chosen over goals (1)	sub-objectives support goals (6)	norms limit goals (13)
		only objectives (2)	sub-objectives improve goals(7)	norms add goals (14)
	goals chosen over objectives (3)	sub-objectives limit goals (8)		
	only goals (4)			
Plans			sub-object. over plans (9)	
		plans help objectives (5)	only sub-objectives (10)	norms alter plans (15)
			plans over sub-object. (11)	
			only plans (12)	

The types of role enactment described above will result in different behaviors for the society since reas following different strategies will be more or less conforming to the expected behavior of the society. The society must therefore be able to describe its

expectations to the agents, during the negotiation phase. Table 3-2 gives an overview of the relationships between role expectations and agent behavior described above.

3.5.2.2 Consistency and compatibility of agents and roles

Having described the possible effects of the characteristics of agent and roles on each other, we must now discuss the actual enactment of a role by an agent. That is, can any agent enact any role? And if not, which are the conditions under which role enactment is possible for an agent? An appropriate enacting relation presupposes that both the agent and the role are internally coherent, that is, that there are no internal conflicts between its components and that the agent goals are achievable. In the following, we will consider only the basic types of role enactment, discussed in the previous section, that is, social enactment, maximal social enactment, selfish enactment and maximal selfish enactment. An extension to the other types of roles enactment described in section 3.5.2.1, is a subject for further research. The informal definitions of internal coherence of role and agent are as follows. The formal definitions of these concepts will be given in chapter 5 and are based on [Dastani et al., 2003].

Definition 3.10. Internally coherent agent

Given an agent described by its goals and plans, we say that the agent is internally coherent iff:

1. *The goals of the agent are not conflicting*
2. *The goals of the agent can be planned and reached using its plans*

Definition 3.11. Internally coherent role

Given a role described by its objectives, sub-objectives and norms, we say that the role is internally coherent iff:

1. *The objectives of the role are not conflicting*
2. *The objectives and norms of the role do not conflict*
3. *Sub-objectives in the same sub-objective set do not conflict*
4. *For each objective of the role for which a non trivial sub-objective set is specified, the objective does not conflict with the sub-objectives in the set.*
5. *For each objective, there is an interaction scene in the society which enables the realization of the objective.*

Given an internally coherent agent and an internally consistent role, we must now describe the conditions under which it is possible for the agent to fulfil the role. In the following, we consider two relationships between agent and role. The first is called **compatibility**, and is based on a subset relation between the agent and the role. Intuitively an agent is compatible with a role when the goals of the agent are a subset of the objectives of the role, that is, the agent naturally fulfils (part of) the objectives or sub-objectives of the role. For example, an agent that has as only goal the goal of reading a certain paper is compatible with the PC-member role that has the objective of reviewing that paper. In the same way, a role is compatible with an agent when the role objectives are a subset of the goals of the agent.

Compatibility indicates that the agent is suitable to fulfil the role. However, such ideal match is often not possible. We therefore introduce a weaker relation between agents and roles. This relation, **consistency**, indicates that the characteristics of the agent and the role do not oppose each other. Informally, an agent is consistent with a role if the goals and plans of the agent do not conflict with the objectives, sub-objectives and norms of the role. Similarly, a role is consistent with an agent if the objectives, sub-objectives and norms of the role do not conflict with the goals and plans of the agent. Consistency indicates that it is possible for the agent to fulfil the role.

Definition 3.12. Compatibility and consistency

Given an internally coherent agent a , and an internally coherent role r :

1. *Agent a is **compatible** with role r , if the goals of a are a subset of the objectives of r , and all plans of a can be formed using the sub-objectives of r .*
2. *Role r is **compatible** with agent a , if the objectives of r are a subset of the goals of a , and all sub-objectives of r can be achieved using the plans of a .*
3. *Agent a is **consistent** with role r if the goals and rules of the agent and the role do not conflict.*

Using the above definition 1 for compatibility of an agent with a role, it can be guaranteed that the agent will only achieve results that are in accordance with the role objectives (all what the agent aims for, is indeed also an objective of the role). In the converse case, that is, if the objectives of the role would be a subset of the goals of the agent, there would be valid and rational plans of the agent that would not guarantee the achievement of the objectives of the role.

An agent that is neither coherent nor consistent with a role has apparently goals and planning rules that, when enacting the role, may violate some norms that are associated with the role. A critical case is when an agent is consistent with a role but not compatible. In such a case, the agent has apparently some additional goals or planning rules that are not associated with the role. Although this agent will not violate any norms associated with the role, the agent may use its own preference ordering to achieve its own goal preferably and thus ignore the goals that are prescribed by the role. It is clear that in such a case, the agent enacts the role inappropriately. A similar situation can occur even when the agent and the role are compatible. In such cases, the appropriate enacting relation is not guaranteed since the enacting relation depends on the ordering on goals and roles as well. An agent can in fact use its own preference relation and thereby never achieve the goals associated to a role. The above definition enables us to define the concept of role enabled agents.

Definition 3.13. Enacting relations

Given an internally coherent agent a and an internally coherent role r :

- *a is strongly enabled to enact r iff a is compatible with r*
- *a is weakly enabled to enact r iff r is consistent with a .*

Given an internally coherent agent a and an internally coherent role r , we should consider what it means for agent a to enact role r appropriately, that is, in a way that meets the expectations of the society. The most simple case is that of total adoption, that is, when an agent a enacts role r , a will adopt all the goals and the norms associated with r . Agent a will also include the obligations of the role in its own model. These will trigger the agent to fulfill the obligations of the role. In addition, the agent can keep some or all of its own goals and rules, as long as it keeps its internal coherence. Furthermore, in order to achieve its goals, the agent must select and plan them. This selection mechanism is usually based on the ordering on goals and planning rules. When a adopts the goals and rules of a role r , a must also extend its orderings to include the goals and rules of the role. Of course, this can be done in many different ways which indicate how the agent assumes the role. For instance, this ordering can give preference to the agent's own goals (a selfish agent), or to the goals of the role (a social agent). In the most simple case, such ordering will be imposed on the agent by the society. In the future, agents should be able to reason and negotiate about combining these orderings. We shall also look at the case of sub-optimal enactment, that is deciding to accept enacting agents that only partially fulfil role expectation, e.g. when no better candidate is available.

The above classification is based on the assumption that the ordering on goals and rules of the enacting agent is in accordance with the ordering that is prescribed by the role. This assumption can be relaxed in which case the agent can either use its own ordering, the ordering of the role, or a combined ordering. Moreover, we may assume an ordering on the obligations and prohibitions associated to the role. In general, the possible choices to use these orderings result in a variety of agent types. In the case of conflicting orders, a social agent will adopt the order associated to the role and not its own. In contrast, a selfish agent will use its own order rather than the order associated with the role. We conclude this section with a proposition that indicates some relations between the different relations between agents and roles.

Proposition 3.1.

Let a be an internally coherent agent and r be an internally coherent role such that $rea(a, r, s)$, for a scene s . Then:

- *a is compatible with r , implies a is consistent with r .*
- *If a is consistent with r , then no violation of r can occur when a enacts r .*
- *Conversely, if a is not consistent with r , then violation of r can occur when a enacts r .*
- *If a is compatible with r , then a can only maximally socially enact r .*
- *If a is compatible with r and a is a social agent, then the enacting agent does not violate any norm that is associated with the role r .*
- *If a is consistent with r and a is a social agent, then the enacting agent does not violate any norm that is associated with the role r .*
- *If a is a maximally social agent, then a will never violate r , whether or not a and r are consistent and/or compatible.*

3.6 Interaction Model

The Organizational Model of OperA, introduced in section 3.4, describes the society design from the perspective of the organization. In section 3.5, we discussed how external agents join a society by specifying social contracts that describe which and how roles are to be enacted. Once ‘inside’ the society, agents will interact with others by participating in interaction scenes. The **Interaction Model (IM)**, specifies the activity of an Agent Society in terms of agreements between role enacting agents (specified in the SM) concerning the enactment of interaction scenes (specified in the OM). The scene scripts specified in the OM describe possible interactions as desired by organizational design. In fact, scripts are abstract, generic patterns for interaction which can be fulfilled in many ways.

In the same way that social contracts in the SM provide an instantiation of the roles specified in the OM to specific agents, the interaction contracts in the IM are the means to operationalize the interaction scripts specified in the OM. That is, social contracts describe the roles and norms applicable to an agent as enactor of a role in the society, and interaction contracts describe the operational results, interaction protocols and social norms applicable to the interaction between agents. The animation of the OM is therefore achieved in two steps: firstly a population of agents is described in the SM and secondly, concrete interactions are specified in the IM. Both instantiation steps are formalized as contracts. Such formalization allows for the verification of society behavior. The stepped approach provides the means to realize the requirements of internal autonomy and collaboration autonomy described in section 3.1.

When role enacting agents come together in an interaction scene, the actual interpretation of the scene script, that is, the interaction protocol to be used must be agreed upon. In OperA, role enacting agents will, for each scene, negotiate an **interaction contract** that defines their partnership, and fixes the way a specific interaction scene is to be played. Interaction contracts describe instances of scene scripts which inherit the organizational norms and objectives described in the interaction script and possibly extend or restrain it to accommodate the specific needs and desires of the participating agents.

OperA provides two levels of specification for interactions. The OM provides a script for interaction scenes according to the organizational aims and requirements and the IM, realized in the form of contracts, provides the interaction scenes such as agreed upon by the agents. It is the responsibility of the agents to ensure that their actual behavior is in accordance with the contracts (e.g. using a monitoring agent or notary services provided by the society for that). However, it is the responsibility of the society, possibly represented by some of its institutional roles, to check that the agents fulfill these responsibilities (this check is possible because of the existence of social contracts, cf. section 3.5.1).

The architecture of IM consists of a set of instances of scene scripts (called scenes), described by the interaction contracts between the role enacting agents for the roles in the scene script, as described in Figure 3-23. An interaction scene results from the instantiation of a scene script, described in the OM, to the reas actually enacting it

and might include specializations or restrictions of the script to the requirements of the reas.

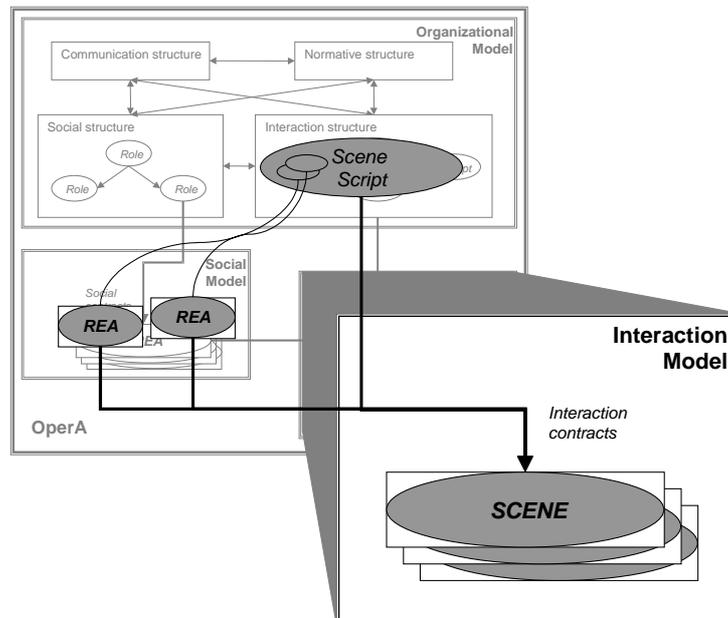


Figure 3-23: Architecture of IM

An interaction contract describes the actual interpretation of a scene script according to the enactors of the participating roles. Because interaction contracts are specified in the formal representation language LCR described in chapter 4, the verification of whether an actual interaction contract indeed fulfils the scene script can be achieved by logical reasoning in LCR. In the remainder of this section, we will introduce the definition of interaction contracts and discuss how interaction contracts can be negotiated and animated. Note however, that the negotiation and animation aspects of interaction contracts are not the focus of this work and are dependent on the specific agents that will participate in a society. We will therefore limit ourselves to present some considerations on this, as was the case as well with the setting up of social contracts.

3.6.1 Interaction contracts

Agreements between role enacting agents on how to realize an interaction script must be specified in order to be able to verify the actual behavior of a society. Moreover, as in OperA agents are viewed as black boxes, the specification of interaction contracts must make no demands or assumptions on the architecture of individual agents participating in the society. As with social contracts, such agreements are represented as contracts, formally specified using LCR. An **interaction contract** describes the conditions and rules applying to interaction between agents in the agent society. That

is, the **clauses** in an interaction contract specify actual instantiations of interaction scene scripts and must indicate the actors involved and the specific agreements and sanctions concerning the scene to be played. The contract must furthermore involve sufficient reas to cover all the needed roles in the scene.

Besides the refinement of the script to the desires and characteristics of the agents participating in the scene instance, interaction contracts must describe the **protocol** agreed by those agents to fulfil the script landmarks. Interaction protocols are the concrete representation of the refinement of scene script landmarks with the particularities imposed by the participants to the specific communicative capabilities of those participants.

Definition 3.14. Interaction Contract

Given a society S and a scene $s \in \text{scenes}(S)$, an interaction contract IC is defined as a tuple $IC = (A, s, CC, P)$, where the set of agents $A = \{a \in \text{Agents} : \exists \text{rea}(a, r, s) \mid r \in \text{roles}(s)\}$, CC is a set of contract clauses and P the protocol to be followed. An interaction contract is denoted by $\text{interaction-contract}(A, s, CC, P)$.

The set A in the definition above represents the set of all agents enacting reas participating in interaction scene s , and CC is a set of deontic expressions describing refinements to the script, that is, possible conditions and deadlines concerning the results and interaction patterns of scene s . Contract clauses are formally represented by LCR expressions. P is the protocol to be followed by the reas.

Protocols describe the actual interaction between reas. A rea interaction protocol describes a communication pattern for reas that is conform to the scene script and consists of CA characteristic of the reas. In principle, any protocol representation language can be used, such as (Colored) Petri Nets [Cost et al., 2000] or UML diagrams [Odell et al., 2001]. Standard protocols can be used for trivial contracts. An example of an interaction contract protocol for the conference society is given in section 3.6.2.

In the Conference Society, the following are examples of interaction contracts for the Review Process scene script²⁰, where P_1 and P_2 are the identifiers of the protocols used:

1. $\text{interaction-contract}(\{PC\text{-Chair}, pc_1, pc_2, pc_3, pc_4, pc_7\}, \text{review-process}, \{ \}, P_1)$
2. $\text{interaction-contract}(\{PC\text{-Chair}, pc_3, pc_5, pc_6\}, \text{review-process},$
 $\{ \text{IF NOT reviews-done}(P, Rev_a, Rev_b) \text{ BEFORE DeadlineR}$
 $\text{THEN OBLIGED}(PC\text{-Chair}, \text{paper-accepted}(P))\}, P_2)$

²⁰ For simplicity sake, reas are referred to in this example by an informal identifier, instead of using the formal format $\text{rea}(\text{agent-id}, \text{role-id}, \text{scene})$. E.g. $pc_1 = \text{rea}(\text{pc-member}, a_1, \text{review-process})$

The first example describes a trivial instantiation of the Review Process script with 5 enactors of the PC-member role. In the second example, the Review Process script is instantiated to 3 enactors of the role PC-member and is made more specific by indicating that in this case, the program chair is obliged to accept all papers for which the reviewers have not given a review by the deadline.

3.6.2 Setting up interaction contracts

The main objective of the IM is to establish interaction contracts describing concrete interaction agreements between agents in the society. A practical implementation of the model will require the description of how interaction contracts are created and managed. The basic idea is that agents participating in an interaction scene must be able to negotiate the specific protocol that will fulfil the interaction script that is specified in the OM. In OperA, such operationalization of an interaction script is achieved in two steps. Firstly, the role enacting agents negotiate the protocol to play the scene. During the negotiation part, reas will agree on the protocol to be used in the actual play of the script. That agreement is fixed in an interaction contract, which forms the basis for interaction. Secondly, the actual play of the scene according to that protocol will take place.

Protocols can be seen as conversations between the agents participating in the scene. In section 3.4.4 we described how scene scripts are related to the abstract communicative acts (CAs) associated with the participating roles. The communicative abilities of the role enacting agents are the active representation of those CAs. That is, messages which an agent can emit count as the abstract CAs described in the scene script. An agreed protocol is a meaningful combination of messages that achieves the results of the scene and satisfies its norms and constraints. The relation between the communicative capabilities of the agents and their corresponding organizational entities described in the OM is illustrated in Figure 3-24.

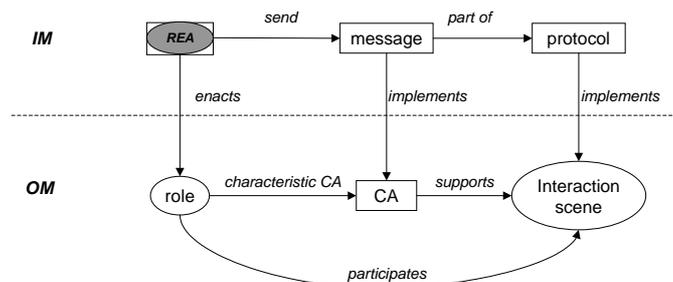


Figure 3-24: Scene enactment

In the following, we illustrate interaction protocols with an example from the Conference society. Suppose that the reas in a given instance of the Conference society use FIPA ACL, and therefore understand the messages *request*, *inform*, *agree*, *refuse*, and others described in the FIPA standard [FIPA, 2002]. Then the protocol for the Review Process scene, with only one rea for the role of PC-member, is depicted in Figure 3-25 using Petri Nets and in Figure 3-26 using UML sequence diagrams. In

both cases, the protocols give an informal representation of the trivial fulfillment of the interaction scene script. We assume that the reader is familiar with both representation techniques.

The Petri Net representation shows the landmarks of the scene script as places, and the communicative acts of the reas as transitions. Reas are represented by special places that must be input for all transitions representing a CA of that rea. Arcs between rea places and transitions conserve the rea tokens at the rea place. External events, such as deadlines are also represented as transitions, which are not triggered by reas²¹.

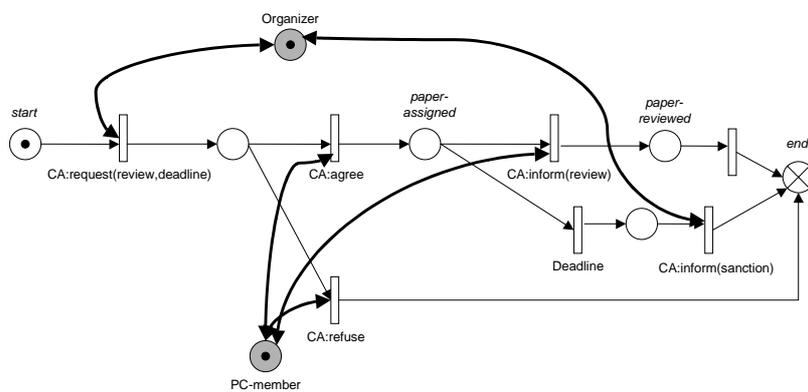


Figure 3-25: Protocol for the Review Process scene, using a Petri Net

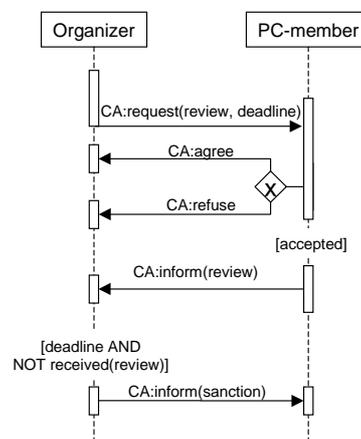


Figure 3-26: Protocol for the Review Process scene, using a UML diagram

²¹ For simplicity, only the relevant aspects of the protocol and the communicative acts are illustrated.

The UML sequence diagram in Figure 3-26 describes the communicative interaction between the reas of role PC-Chair and PC-member. Using this modeling technique, the landmarks of the script are not explicitly represented but are taken as the successful realization of the CAs of the reas.

In the same way as with social contracts, the verification of the correctness and completeness of a interaction contract and its corresponding interaction scene script is possible because both interaction contracts and scene scripts have a formal semantics specified in the logic language LCR. In chapter 5 we will discuss this verification in detail.

3.7 Conclusions

In this chapter, we have presented a model for multi-agent systems, the Agent Society Model (OperA). The model focuses on the organizational relations and requirements of multi-agent systems, and as such, takes a perspective on the system external to the agents themselves. In the current chapter we have given a detailed overview of the components and aims of the model and have made extensive use of examples to illustrate the capabilities and design choices of OperA. The formal semantics of OperA will be presented in chapter 5.

In the remainder of this section we will give a summary of the OperA framework and describe the contributions of OperA to multi-agent design and research.

3.7.1 OperA Summary

The OperA framework is based on research in organizational science and economics of how people create organizational structures to solve their coordination problems. The behavior of agents is regulated by both its social contracts, that describe its roles in the society, as well as by its interaction contracts, that describe actual interactions between agents. The OM describes the behavior of the organization by distributing the objectives of the organization between different roles. The OM also includes the description of the normative and communicative elements as well as the interactions in the domain. The SM describes the enactment of roles by (external) agents, by means of social contracts. Finally, in the IM the interactions between the agent populations are specified in terms of interaction contracts, that describe the explicit commitments between the partners on how to realize a certain interaction.

Figure 3-27 on page 97, depicts the conceptual model of OperA in terms of its core concepts and relationships between those concepts. In Figure 3-27, rounded boxes represent core entities of OperA, and rectangles represent the main components of those entities. The main models of the OperA framework (the Organizational, Social and Interaction models) are depicted with a gray background. Bold arrows identify the definition relationships between concepts, and other arrows depict extra relationships between concepts. Finally, broken arrows and hexagons represent external entities, not represented directly in OperA but that have a close relation to system. The figure does not aim at a complete specification of the model, but should be seen as an

illustration of the overall model. In the previous sections we have described all OperA concepts in detail. The formal semantics for OperA will be presented in Chapter 5.

3.7.2 Contribution to multi-agent systems

Agent-based models for organizations see agents as autonomous social entities (like employees in a company) that exhibit flexible, responsive and proactive behavior and are able to interact within a structure designed to realize organizational objectives and compliance to organizational norms and requirements. Such setting requires the integration of individual desires with organizational requirements. Furthermore, an open environment requires that agent societies are designed independently from the specific characteristics of the agents. Other current approaches, as the ones described in section 3.2, are often not well suitable for this job because they either use a centralist view on the development of agent societies or they are unable to allow for control or prediction of agent behavior.

OperA distinguishes between the mechanisms through which the structure and global behavior of the model is described and coordinated, and the aims and behavior of the service-providers (agents) that populate the model and is therefore able to represent interactions between agents in an open context. In the beginning of this chapter, we identified the requirements for models for open organizations: the **internal autonomy requirement**, and the **collaboration autonomy requirement**. The OperA framework fulfils both these requirements. The OM allows for the specification of pure organizational models of societies that do not require any knowledge of the agent architecture. Interactions and roles are described in the OM in an abstract fashion that allow for several possible concretizations. Furthermore, concretization of an organizational design is done in two steps: first the populations are chosen (SM) and only then agreements concerning interaction are decided (IM). This allows for further flexibility since any given population still can choose between several possible concrete interactions.

The combination of *refinement* and *agreement* in our proposal is rather unique. Refinement in itself is not new; for example, it is also used under the name of *leveling* in AUML [Odell et al., 2001], and is defined by the OMG as ‘a more detailed description that conforms to another (its abstraction)’²². However, in the view developed in this chapter, refinement is more than an abstraction device: it is a matter of the proper balancing of responsibilities. On the other hand, agreements on how specific agents are going to enact a given role or play a scene are fixed as contracts, which represent the responsibilities and possibilities of each party.

²² Refinement is proposed as a technique in UML’s Model Driven Architecture (<http://www.omg.org/mda/>), so that more detailed descriptions are built in a systematic way from abstract ones.

From the society perspective, it is important that these contracts adhere to the landmarks given by the Organization Model. Therefore, the specification language for OperA must be formalized syntactically and semantically so that formal verification is possible. We have developed a branching time deontic logic called LCR (described in chapter 4) that can provide the logical semantics of both the Organization Model and the contracts. Using the models presented in this chapter, and the LCR logic, it becomes possible to give a precise and implementable specification of agent societies. One important topic for future research is precisely to demonstrate this, that is, to develop specification architectures and design patterns for various types of agent societies.

Finally, one apparent omission of our work is the lack of an implementation of OperA. A working system, and its application to one or more case studies would demonstrate the practical possibilities of the model and prove the conceptual choices of OperA. However, we feel that, on the one hand, the development of a complete implemented tool to specify OperA models would be in itself enough for another Ph.D. thesis, and on the other hand, work on a prototype would use up quite some time which we have rather used for the development of the semantics of OperA, presented in chapter 5.

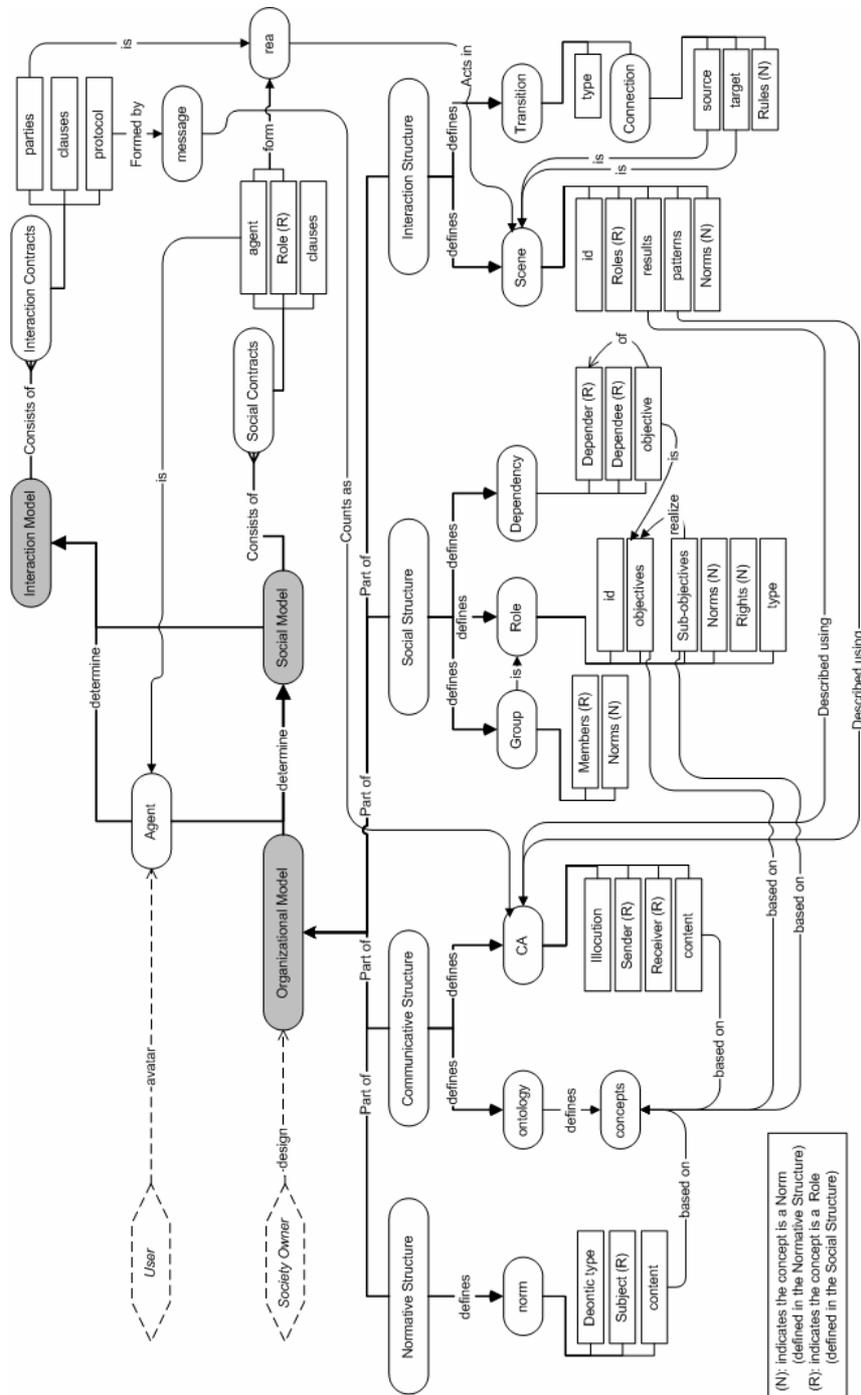


Figure 3-27: Conceptual model of Opera

Chapter 4

Logic for Contract Representation

'Logic has a wider reach than truth'
- G.H. von Wright, Is there a logic of norms?
Logic and Norms 4(3):265-283, 1991

*'Norms can neither be true nor false,
Thus there cannot be a logic of norms'*
- J. Jørgensen, Imperatives and Logic,
Erkenntnis, 7: 288-296, 1938.

The Agent Society Model presented in chapter 3 distinguishes between the mechanisms through which the structure and global behavior of the model are described and coordinated, and the aims and behavior of the service-providers (agents) that populate the model. In the OperA framework, contracts are used to integrate the top-down specification of organizational structures with the autonomy of participating agents. In this chapter we introduce the **Logic for Contract Representation** (LCR), a very expressive logic for describing interaction in multi-agent systems. LCR makes it possible to check whether agents in an agent society follow some desired interaction patterns and whether desired social states are preserved by agent activity. LCR is used as a formal basis for the OperA framework. In order to be able to compare LCR with other deontic logics, we also will describe how LCR behaves in common contrary-to-duty situations.

This chapter is organized as follows. Section 4.1 provides our motivation for the formalization of norms in agent systems, and the use of contracts for the representation of interaction. In section 4.2, we discuss related work on the formalization of organizational behavior in multi-agents systems. Section 4.3 introduces LCR. In section 4.4 we show how several contrary-to-duty situations are handled in LCR. Finally, in section 4.5 we present our conclusions and indicate directions for further research. In chapter 5 we will describe how LCR can be used to formalize the OperA model and represent interaction contracts between agents.

A previous version of this chapter was published as [Dignum et al, 2003].

4.1 Introduction

Norms have been identified in social sciences as crucial tools for important social issues such as coordination, cooperation, trust and reputation. A formalism for agent societies must furthermore be able to uniformly describe and reason about social structure (landmarks and roles) and interaction (social and interaction contracts). Such formalism facilitates the analysis of societies and verification through logical reasoning, that is, verification of society design gets down to prove inconsistencies in the logical description.

In systems where agents are assumed to be autonomous and rational, agents can, involuntarily or by deliberate choice, violate social norms and regulations and therefore one must be able to deal with and reason about such violations. The use of deontic logic as a formalism for multi-agent systems has been advocated by several researchers (cf. [van der Torre, 2003]). Deontic logic provides mechanisms to reason about violability of norms, that is, about how to proceed when norms are violated. In practice, logical formalisms for agents have been used to (1) specify agents in an abstract manner and to (2) verify and reason about agent behavior, independently of the implementation language used to represent the agent.

A more advanced form of agenthood, normative agents (that is, agents that can reason about norms and obligations) can bridge the gap between individual autonomous agents and the agent society, in the sense that the cognitive concept of obligation is the building block of complex social notions like coordination, cooperation, trust and reputation.

Furthermore, verification of the behavior of an open society, where the design of participating agents cannot be controllable, must be based on the externally observable effects of agent actions. That is, from the society perspective, different actions that bring about the same state of affairs in the world cannot be distinguished. From the above considerations, it follows that a logical formalism for the OperA model must be able to represent:

- Deontic relations (obligations, prohibitions, permissions)
- Externally observable results of agent actions (changes in state caused through influence of agents)
- Temporal relationships (effect of actions and agreements is not instantaneous and not deterministic, several futures are possible at each moment depending on agent decisions and environment changes)
- Violations and reasoning about effects and recovery from violated states

In this chapter we present a logical formalism for describing interaction in an agent society. The formalism enables the specification of social norms and interaction contracts. The logical formalism combines elements from deontic and branching time logic. In the remainder of this paper we will introduce the main features of this logic.

4.2 Related Work

In situations where several agents cooperate within an organizational framework, designed to realize global society objectives, contracts are the means to specify expectations on the behavior of other participants. In this chapter we are concerned with the formalization of such contracts. Due to the characteristics of contracts, the best way to formalize contracts is, in our opinion, to make use of deontic and temporal logics.

Although a great deal of work has been done in the specification of contracts, to our knowledge, there is not, as yet, any complete logical formalism for contract specification. In the area of Business Organization, work concentrates on the development of standards for contracts. However, such standards mostly provide a purely syntactic formalization of contracts²³. As a result, most contract-drafting systems are in fact advanced word processors, that use templates for contract clauses.

A more interesting proposal is the DocLog model proposed by Tan and Thoen that provides three layers for contract specification, including a logical layer [Tan, Thoen, 2000]. The formalization of this layer is based on a previous version of the same deontic logic that is the basis of the definition of LCR, namely the BTLcont logic introduced in [Dignum, Kuiper, 1999]. Moreover, the main objective of DocLog is the semi-formal specification of contracts based in XML, which reflects its aim of representing contracts in a web-based environment to be used by people, and not as representation of agent interaction.

Deontic logic for agent organizations has also been recently used in a proposal by Pacheco and Carmo [Pacheco, Carmo, 2003]. They introduce a role based model for *organized collective agency*, based on the legal concept of *artificial person* and on a normative perspective on organizations. Their logic attempts to capture the concept of taking up a role. However, the logic does not include any temporal concepts, which makes it not suitable to represent real life organizations. Moreover, they lack a formal definition of roles (viewed as identifiers) and assume that roles are generated from the contracts between agents.

Finally, we should mention the formalization work of the Alfebiite consortium [Artikis, Pitt, 2003]. Although the aim of this work is not the logical formalization of contracts, they propose a formal model for open agent societies that uses deontic logic for the representation of interaction protocols between agents. However, the model does not provide mechanisms to explicitly describe temporal aspects and therefore is not suitable to specify and reason with deadline or other violation issues.

²³ A good example is the work of the International Chamber of Commerce described at <http://paction.modelcontracts.com/aclhome/>

4.3 Logic for Contract Representation

The Logic for Contract Representation (LCR) that we propose is based on a branching-time logic. This means that formulae are interpreted over tree-type branching structures that represent all conceivable ways the system can evolve. Nodes represent *states* and arcs correspond to the occurrence of *events*. A *path* represents a course of events and links states in the time structure according to the choices and possibilities available to agents at each moment. Our proposal extends the formalism based on Temporal and Deontic Logic, BTLcont, proposed by Dignum and Kuiper [Dignum, Kuiper, 1999]. BTLcont is in itself an extension to the well known branching-time temporal logic CTL*, proposed by Emerson and Halpern [Emerson, 1990] and [Halpern, Moses, 1992]. While Emerson and Halpern provide a sound and complete axiomatization for CTL*, we do not address the issue of completeness in this chapter. Our main aim is to present an expressive semantics for contracts, that represent interaction between agents in an abstract way, that is, independent from the internal architecture of the agents.

We further extend branching time logic with a *stit* operator, E_a ('agent a sees to it that') based on Pörn [Pörn, 1974]. This allows us to refer to the externally 'observable' consequences of an action instead of the action itself. Remember that agent internals are not visible from the organizational perspective, and therefore it is not possible to refer to specific actions of an agent. In our use of E_a , we draw from the logic proposed by Wooldridge for the combination of a *stit* operator with a temporal logic [Wooldridge, 1996].

Moreover, clauses in a contract, deontic expressions in LCR, indicate that something must happen (ideally something happens) but in fact, it may never happen at all! A logic for contract representation must therefore be able to reason about states in which an obligation has been violated. Obligations have to do with the preference of individuals (or societies) to be in a certain state. $O_a\varphi$ (the obligation for agent a to see to it that φ holds) indicates that, in the society, it is ideal for a to be in a state where φ holds. This does not mean that agent a cannot be in other states either by choice or necessity. A violation, $\text{viol}(a,\varphi,\delta)$, is interpreted as 'agent a is in a violation situation concerning the obligation to be in a state where φ holds before deadline δ '²⁴. The basic idea is that worlds in which a violation proposition holds are less preferred by the agent concerned, in that society. Sanctions are defined in order to make it possible for violations to be redeemed.

²⁴ When no deadline is specified for an obligation, the violation is simplified to $\text{viol}(a, \varphi)$.

4.3.1 Syntax of LCR

LCR is an extension of CTL*, which in turn is an extension of classical propositional logic²⁵. Well-formed formulae of LCR are built from a set Φ of atomic propositions that may be combined using the classical proposition connectives \vee ('or') and \neg ('not'). Other propositional connectives such as \wedge ('and'), \rightarrow (logical implication) and \leftrightarrow (logical equivalence) can be introduced as abbreviations. The language also contains the constants *true*, *false*, and the CTL* operators:

- A (always in the future),
- S (since),
- X (in the next state, on all paths),
- Y (yesterday, or in the previous state),
- U (until),
- \leq (before), and,
- the *stit* operator E.

The E operator is labeled with agents and/or group identifiers. Elements a, b, \dots of a set Ags of agent identifiers are used as labels for E. For example E_a is read as 'agent a sees to it that'. Furthermore, we introduce a predicate $\text{viol}(a, \varphi, \delta)$, which holds in states where an obligation to do φ before δ has been violated by agent a .

Definition 4.1. Syntax of LCR

The set of well-formed formulae of LCR is introduced inductively, given a set Φ of atomic propositions (including true and false) and a set Ags of agents. As in CTL, LCR distinguishes between state formulas (evaluated in a state) and path formulas (evaluated in a path).*

1. Every member of Φ is a state formula
2. If φ, φ_1 and φ_2 are state formulas, then so are $\neg\varphi, \varphi_1 \vee \varphi_2, Y\varphi$, and $\varphi_1 S \varphi_2$
3. If φ is a state formula, then so is $E_a\varphi$, for all $a \in \text{Ags}$
4. If φ_1 and φ_2 are state formulas, then so is $\text{viol}(a, \varphi_1, \varphi_2)$, for all $a \in \text{Ags}$
5. Each state formula is also a path formula
6. If ψ is a path formula, then $A\psi$ is a state formula
7. If ψ, ψ_1 and ψ_2 are path formulas, then so are $\neg\psi, \psi_1 \vee \psi_2, \psi_1 U \psi_2, \psi_1 \leq \psi_2$, and $X\psi$

²⁵ In finite domains, the existential quantifier can be introduced as a finite disjunction and the universal quantifier as finite conjunction.

4.3.2 Semantics of LCR

Usually different events are possible at any moment. That is, at each moment different futures are possible depending on the events in the world. We therefore have defined the semantics for LCR using branching time structures.

Definition 4.2. Branching Time Structure

A branching time structure is a tuple (W, R) where:

- W is a set of worlds (*states*) and
- $R \subseteq W \times W$ is the successor relation on states, such that the reflexive, transitive closure of R , R^* , is a total tree relation.

R^* represents all possible courses of system history. A **path** (or trace) through R is a sequence (s_i, s_{i+1}, \dots) , such that $\forall i \in \mathbb{N}$ we have $(s_i, s_{i+1}) \in R$. If t is a path then state $t(i)$ is the i -th element of t . We assume that there is a state s_0 , which is the root of (W, R) . Furthermore, we represent the tail of the path starting with state $t(i)$ by $t[i]$.

Definition 4.3. Semantic model

A semantic model M for LCR is a structure $M = (W, R, \pi)$ where (W, R) is a branching time structure and π is a valuation function, which associates each $s \in W$ with the set of atomic propositions from Φ that are true in that world.

A path is a full and infinite sequence of states. Paths do not have to start from the root, but once started, there is always a following state in the path. By acting, agents can influence the next state in a path. The actions of agents are some of the possible events in the tree. In order to be able to represent the influence of an agent on changes in the world, we introduce the notion of controllable and uncontrollable expressions.

4.3.2.1 Controllable and uncontrollable propositions

Intuitively it only makes sense to specify $E_a\phi$ for a formula ϕ if agent a can indeed ‘see to it’ that ϕ holds, that is, if the agent can control or influence the truth value of ϕ . For instance, it does not make sense to express $E_a \text{rains}$ because the fact whether it rains or not is not something that usually an agent can control. Inspired by the work of Boutelier [Boutelier, 1994] and Cholvy and Garion [Cholvy, Garion, 2001], we partition for each agent a the set of atomic propositions Φ in any world w of M in two classes: C_a and \bar{C}_a in which C_a is the set of atomic propositions that agent a can control and \bar{C}_a the set of atomic propositions that a cannot control.

Definition 4.4. Valuation function

1. Let π be the valuation function of a semantic model $M = (W, R, \pi)$, which associates each $s \in W$ with the set of atomic propositions from Φ that are true in that world. For a set P of atomic propositions, $\pi(P)$ indicates the restriction of π

to the propositions in P (that is, the subset of true propositions of P). For every agent a , π can thus be written as $\langle \pi(C_a), \pi(\bar{C}_a) \rangle$ the composition of the restriction of π to the controllable atomic propositions of a and the non-controllable atomic propositions of a .

2. For a set P of atomic propositions, $\mathbf{II}(P)$ is the set of all valuations of atoms of P .
3. Given two valuation functions u and v such that $\text{dom}(u) \cap \text{dom}(v) = \emptyset$, the concatenation of two valuation functions, $u.v(p)$, is defined as:

$$u.v(p) = \begin{cases} u(p), p \in \text{dom}(u) \\ v(p), p \in \text{dom}(v) \end{cases}$$

Definition 4.5. Controllable and uncontrollable propositions

Given classes C_a and \bar{C}_a defined as above,

1. a proposition φ is **a-controllable** in a semantic model $M = (W, R, \pi)$, iff $\forall u \in \mathbf{II}(\bar{C}_a), \exists v_1, v_2 \in \mathbf{II}(C_a)$ and $\exists s_1 \in W, \pi(s_1) = u.v_1, \exists s_2 \in W, \pi(s_2) = u.v_2$, such that $(M, s_1) \models \varphi$ and $(M, s_2) \not\models \varphi$.
2. An expression is **a-uncontrollable** iff it is not a-controllable.

For example consider model $M = (W, R, \pi)$ and the atomic propositions p and q ,

such that $p \in C_a$ and $q \in \bar{C}_a$. In this case, proposition $p \wedge q$ is not a-controllable, because q is not a-controllable, and if q is false, agent a cannot make $p \wedge q$ to be true. Moreover, tautologies are never a-controllable, that is, if something is always true, no agent can claim to see to it that it can see to it that a tautology will become true. The formal proof for this is as follows:

Theorem: $\forall \varphi$, if $\models \varphi$ then φ is a-uncontrollable for agent a . That is, agent a cannot control tautologies.

Proof. Suppose $\exists \varphi$, such that $\models \varphi$ and φ is a-controllable. $\models \varphi$ implies $\forall s, (M, s) \models \varphi$. However from the definition of a-controllable there must be a $s \in W$ such that $(M, s) \models \neg \varphi$. Therefore φ cannot be a tautology.

4.3.2.2 Path and State Semantics

As in CTL*, we define the semantics for state and path formulae separately. A path formula is a formula that is interpreted with respect to a path through a branching time structure. Paths correspond to histories of the system. In contrast, a state formula is interpreted with respect to a system state. The semantics of path formulae are given via the path formula satisfaction relation represented by ' \models ' that relates tuples of the form (M, t) , where M is a LCR-model, $M = (W, R, \pi)$, and t a path (or trace) in M , to path formulae of LCR. This relation is defined by the following rules:

- (P1) $(M, t) \models \varphi$ iff $(M, t(0)) \models \varphi$, where φ is a state formula
(P2) $(M, t) \models \neg\psi$ iff not $(M, t) \models \psi$
(P3) $(M, t) \models \psi_1 \vee \psi_2$ iff $(M, t) \models \psi_1$ or $(M, t) \models \psi_2$
(P4) $(M, t) \models X\psi$ iff $\forall t': (t, t') \in R, (M, t') \models \psi$
(P5) $(M, t) \models \psi_1 U \psi_2$ iff $\exists i \in \mathbb{N}$ such that $(M, t[i]) \models \psi_2$ and $\forall k \leq i, (M, t[k]) \models \psi_1$
(P6) $(M, t) \models \psi_1 \leq \psi_2$ iff $\forall i \in \mathbb{N}$ such that $(M, t[i]) \models \psi_2, \exists j \leq i, (M, t[j]) \models \psi_1$

The semantics of state formulae are given via the state formula satisfaction relation, also represented by ' \models ' that relates tuples of the form (M, s) , where M is a LCR-model, $M = (W, R, \pi)$, and s a world in W , to state formulae of LCR. This relation is defined by the following rules:

- (S1) $(M, s) \models p$ iff $p \in \pi(s)$, where $p \in \Phi$
(S2) $(M, s) \models \neg\varphi$ iff not $(M, s) \models \varphi$
(S3) $(M, s) \models \varphi_1 \vee \varphi_2$ iff $(M, s) \models \varphi_1$ or $(M, s) \models \varphi_2$
(S4) $(M, s) \models A\psi$ iff $\forall t \in \text{paths}(W, R)$, if $t(0) = s$ then $(M, t) \models \psi$
(S5) $(M, t(i)) \models Y\varphi$ iff $(M, t(i-1)) \models \varphi$
(S6) $(M, t(i)) \models \varphi_1 S \varphi_2$ iff $\exists k \leq i$ such that $(M, t(k)) \models \varphi_2$ and $\forall j, k < j \leq i, (M, t(j)) \models \varphi_1$
(S7) $(M, s) \models E_a\varphi$ iff 1) for φ is *a-controllable*: $\forall s' \in W$, if $(s, s') \in R, (M, s') \models \varphi$
2) for φ is *a-uncontrollable*: *false*

The semantics of LCR are standard branching time semantics with the exception of $E_a\varphi$. $E_a\varphi$ is intended to represent the fact that agent a sees to it that φ is satisfied. The semantic rule for $E_a\varphi$ can be described informally as: agent a acts in world w in such a way that the truth of the a -controllable expression φ is guaranteed. The *stit* operator E_a ignores the means by which agent a will bring about a state of affairs. We furthermore introduce the operator $D_a\varphi$ that represents the fact that a specific state of affairs has indeed been brought about by agent a in the previous world. $D_a\varphi$, meaning ' φ has been done by a ' is defined as:

$$(M, t(i)) \models D_a\varphi \quad \text{iff} \quad (M, t(i-1)) \models \neg\varphi \wedge E_a\varphi$$

The following property holds for D_a :

$$\models D_a\varphi \rightarrow \varphi \quad (1)$$

This is a direct consequence of the definition of E . That is, if in a state $t-1$, $E_a\varphi$ holds, from the definition of E it follows that in state t , φ holds (for a -controllable φ). And, for uncontrollable φ , $D_a\varphi$ never holds, so the implication is true as well.

4.3.3 Representing deontic modalities in LCR

In a logic for the representation of contracts it must be possible to specify a time limit for realizing a certain state of affairs. Contracts express commitments that agents make to each other, that is an obligation for an agent to bring about a certain state of affairs (that is of interest to another agent). A deadline for the fulfillment of such obligations is usually indicated by the contract. A possible way to express deadlines is to indicate that an event should take place before a certain condition becomes true.

Moreover, clauses in a contract indicate that something must happen (it is desirable that something happens) but in fact it may never happen at all! A logic for contract representation must therefore be able to reason about situations (worlds in the semantics above) in which an obligation has been violated. Obligations have to do with the preference of individuals (or societies) to be in a certain state. $O_a\varphi$ indicates that, in the current society, it is ideal for a to be in a state where φ . This does not mean that agent a cannot be in other states either by choice or necessity. Worlds where a violation proposition holds are less ideal for the agent concerned.

4.3.3.1 Obligations with deadlines

We introduce obligation as a derived operator in LCR. Obligations in LCR express the fact that agent a is expected to bring about a certain result (or state of affairs) ρ before a certain condition (deadline) δ has become valid.

Definition 4.6. Obligation with deadline

The obligation of agent a to see to it that result ρ is achieved before an event δ happens, is defined in LCR as:

$$O_a(\rho \leq \delta) =_{def} A((\neg\delta \wedge \neg viol(a, \rho, \delta)) \cup ((E_a \rho \wedge X(A \Box \neg viol(a, \rho, \delta))) \vee X(\delta \wedge viol(a, \rho, \delta))))$$

Formally, the operator \Box (always in the future), is defined as follows:

$$\Box \varphi =_{def} \neg(\text{true} \cup \neg\varphi).$$

This definition expresses the fact that, considering an obligation with a deadline, two moments are relevant: the moment when the agent sees to it that ρ holds, and the moment when the deadline occurs. In all states until either of those moments, neither the deadline nor the violation hold. In the case the agent sees to it that ρ , then the violation will not occur anymore in the future, expressed by $X(A \Box \neg viol(a, \rho, \delta))$. In the case the deadline holds, the violation also will hold. Note that, whereas temporal expressions are persistent in LCR, this is not necessarily the case for non-temporal expressions. This implies that the achievement of ρ when an obligation $O_a(\rho \leq \delta)$ holds, does not guarantee that it will always be the case that ρ .

Moreover, the operator O_a represents an obligation for agent a , without reference to the recipient of the obligation. In some cases it is relevant to refer to an obligation from agent a towards agent b . This case is described by O_{ab} . The obligation without deadline is a special case of the definition above and is defined as²⁶:

$$O_a \rho =_{def} O_a(\rho \leq true)$$

The definition of obligation expresses the fact that in all worlds reachable from a world where $O_a(\rho \leq \delta)$ holds, either the agent has seen to it that result ρ has been achieved or a violation of the obligation holds in those worlds. Intuitively, the idea seems to be that an obligation will ‘disappear’ once the result is achieved within the deadline. However, this is not the case. Fulfilling an obligation does not mean that the obligation disappears but, once the result is achieved within the deadline, the obligation can never result in a violation anymore. Formally this is represented as:

$$\models O_a(\rho \leq \delta) \rightarrow \neg viol(a, \rho, \delta) \ S D_a(\rho \leq \delta) \quad (2)$$

It is interesting to note that in LCR the proposition $O_a(\rho \leq \delta) \wedge O_a(\neg\rho \leq \delta)$ is consistent. This can be proven by the possible sequence of states in M , given in Figure 4-1, and considering the property above, that says that the obligation expression does not disappear after the result is achieved. The truth value of the obliged expression φ can however change in time. In Figure 4-1, at point t_k before δ , both obligations hold and φ does not hold. In the same way, it can be proven that $O_a \rho \wedge O_a \neg\rho$ holds.

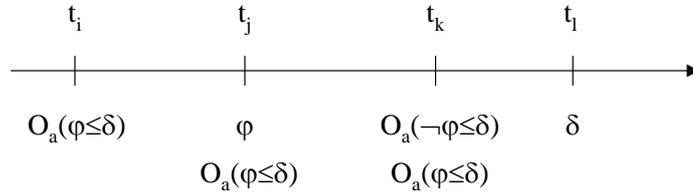


Figure 4-1: Possible path for $O_a(\rho \leq \delta) \wedge O_a(\neg\rho \leq \delta)$

4.3.3.2 Conditional Obligations

Conditional obligations are obligations that only become active if the precondition becomes valid. Unlike regular obligations, that only hold once, a conditional obligation will come in force every time the condition holds.

²⁶ Violations related to an obligation without deadline are often also specified without the deadline parameter, that is, $viol(a, \varphi)$ is an abbreviation for $viol(a, \varphi, true)$.

Definition 4.7. Conditional Obligation with deadline

The obligation of agent a to see to it that result ρ is achieved before an event ρ happens given that precondition σ holds, is defined in LCR as:

$$O_a(\rho \leq \delta \mid \sigma) =_{def} A((\sigma \rightarrow O_a(\rho \leq \delta)) U(D_a \rho \vee \delta))$$

In this definition, the sub-expression $U(D_a \rho \vee \delta)$ is necessary in order for the conditional obligation to be removed once it has been realized (or it cannot be done anymore because the deadline has passed). Otherwise, whenever π becomes true the obligation will arise. Because π can still be true after the obligation is fulfilled, the obligation will arise again and again.

Note that the special case of a conditional obligation, $O_a(\rho \leq \delta \mid true)$ is not the same as the regular obligation $O_a(\rho \leq \delta)$, but expresses an obligation that always holds. Another property of conditional obligations, is that they become ‘normal’ obligations whenever the precondition holds. Formally, this is expressed as:

$$\models O_a(\rho \leq \delta \mid \pi) \wedge \pi \wedge \neg \rho \wedge H \neg \delta \rightarrow O_a(\rho \leq \delta) \quad (3)$$

In the above expression, $H \neg \delta$ expresses the fact that the deadline δ has never happened in the past. Formally, the operator H (always in the past), is defined as follows:

$$H\varphi =_{def} \neg(\text{true } S \neg\varphi).$$

Intuitively, one expects that once a deadline has passed, its violation will always hold, or at least until a sanction is performed by the agent. However this is not yet the case, if we consider the definitions above. We need thus to introduce the following axiom:

$$\models \text{viol}(a, \rho, \delta) \rightarrow A(\text{viol}(a, \rho, \delta) U D_a(\sigma \leq \delta'))$$

where σ is the sanction expression that, if achieved before a deadline δ' , will remove $\text{viol}(a, \rho, \delta)$, which can also be represented by $\text{sanction}(\sigma, \delta', a, \rho, \delta)$.

Sanctions are defined as obligations conditional on the occurrence of a violation, which leads to the following observation concerning sanctions. Given a sanction $\text{sanction}(\sigma, \delta', a, \rho, \delta)$, that is, the sanction that will remove $\text{viol}(a, \rho, \delta)$, then:

$$O_a(\sigma \leq \delta' \mid \text{viol}(a, \rho, \delta)) \wedge \text{viol}(a, \rho, \delta) \rightarrow ((\text{viol}(a, \rho, \delta) \leftrightarrow \text{viol}(a, \sigma, \delta')) S \delta') \quad (4)$$

Informally, given a sanction related to a violation $\text{viol}(a, \rho, \delta)$, in states where the violation holds, the realization of the sanction will result in states where the violation is removed (does not hold anymore). So, either all the related violations disappear through performing the sanction or additional violations arise when it is not performed.

Obligations, such as sanctions, that hold in cases of violation of another obligation, are known as **contrary-to-duty** obligations. Contrary-to-duty situations lead to some well-known paradoxes in standard deontic logic (SDL). In the next

section we discuss some of these and describe how our formalism behaves in contrary-to-duty situations.

4.4 Contrary-to-duty Imperatives

A contrary-to-duty obligation is an obligation that is only in force in a sub-ideal situation. This is often necessary to represent some aspects of legal systems. Unfortunately, contrary-to-duty reasoning leads to notorious paradoxes of deontic logic. Paradoxes of deontic logic are logical expressions (in some logical language) that are valid in (many) well-known logical systems for deontic reasoning, but which are counterintuitive in a common sense reading [Meyer et al., 1998]. The problem with most contrary-to-duty situations is that obligations referring to the most ideal situation conflict with obligations referring to less ideal cases. In contrast to many deontic logics, LCR explicitly represents the notion of violation, and it is a temporal logic, which makes it possible to refer to different moments. Intuitively, violation changes the context of (normative) reasoning. Violation contexts distinguish between ideal and sub-ideal contexts, varying in degree of ‘ideality’. Therefore, the representation of contrary-to-duty imperatives in LCR is in most cases straightforward.

It is not our intention to show here how LCR behaves for all the many contrary-to-duty situations that have been described for deontic logic, but we will take three versions of the Chisholm paradox [Chisholm, 1963], the forward, the parallel and the backward versions, as representative. Moreover, because our research is applied in the area of Knowledge Management and the support of Communities of Practice, we have taken examples from these areas for the informal description of the paradoxes, instead of using well known descriptions, such as the ‘gentle murder’ situation.

Note that, from the definitions of obligation and conditional obligation in LCR, it can be proven that $O_a(\varphi) \wedge O_a(\psi|\varphi)$ does not imply $O_a(\psi)$. This is essential for the faithful representation of contrary-to-duty situations.

4.4.1 The forward version of the Chisholm paradox

In this contrary-to-duty situation extra activities are obliged after a certain obligation holds, which do not hold otherwise. In our example, the rules of a knowledge sharing community are as follows: Meeting chairs must publish notes of the meeting (F1). When meeting notes are published, this must be announced to group members (after publishing) (F2). If not published, then it must not be announced (F3).

In Standard Deontic Logic (SDL), a paradox follows from the case that notes are not published (F4). The formal specification of the above rules, in the generic case is:

- (F1) $O_a(\varphi)$ (a is obliged to publish meeting notes)
- (F2) $O_a(\psi | \varphi)$ (given that notes are published, a is obliged to announce it)
- (F3) $O_a(\neg\psi | \neg\varphi)$ (if not published, a is obliged not to announce publication)
- (F4) $\neg\varphi$

From the way obligations are defined in LCR, F1 expresses that φ still has to be made true by a , that is $E_a(\varphi)$, and in F2 the obligation $O_a\psi$ only holds in states where φ already holds. This implies a time difference between when φ should be true and ψ should be true. This is why this represents the forward version of the Chisholm paradox. Because, in our formalism, violation of norms is explicitly represented, this paradox does not result in states where a contradiction holds. Originating states can, however, be associated with a preference. The following table portrays the different states originating in this situation, where state S_0 represents a state where all obligations hold and no action has been done and in each state only the relevant propositions are specified.

Table 4-1: Forward version of Chisholm paradox in LCR

S_0	S_1	Possible next states
$O_a(\varphi)$ (F1)	$\neg\varphi$ (F4)	$\neg\varphi$ (F4)
$O_a(\psi \mid \varphi)$ (F2)	$\text{viol}(a, \varphi)$ (from F1)	$\text{viol}(a, \varphi)$ (from F1)
$O_a(\neg\psi \mid \neg\varphi)$ (F3)	$O_a(\neg\psi)$ (from F3)	ψ (from F3)
		$\text{viol}(a, \psi)$ (from F3)
		$\neg\varphi$ (F4)
		$\neg\psi$
		$\text{viol}(a, \varphi)$ (from F1)

4.4.2 The parallel version of the Chisholm paradox

This version of the Chisholm paradox is better known as the Forrester, or ‘gentle murder’ paradox. An example of this contrary-to-duty situation, that results in a paradox in SDL, is when the following agreements hold in a community: Members of the community are obliged never to publish internal department reports (P1). But, if a member does publish an internal report, then it must be published in the discussion area (P2). Because publishing in the discussion area is a special case of publishing, this means that both activities are simultaneous (P3). The paradox arises when a report is published (P4). The generic formal specification of this situation is:

- (P1) $O_a(\neg\varphi)$ (a is obliged not to publish internal reports)
- (P2) $(O_a(\varphi \rightarrow \psi))$ (if published, a must publish it in the discussion area)
- (P3) $\psi \rightarrow \varphi$ (publishing follows from publishing in discussion area)
- (P4) φ (a report is published)

Because, in our formalism, violation of norms is explicitly represented, this situation does not result in states where a contradiction holds. From P1 to P3, it is not possible in LCR to derive $O_a(\psi)$, which is usually the cause of the ‘gentle murder’ paradox in SDL. The following table portrays the different states originating in this situation, where state S_0 represents a state where the obligations hold and no action has been done and in each state only the relevant propositions are specified. Because ψ can only be done at the same moment as φ , in states where φ holds either ψ or $\neg\psi$ must hold (ψ cannot happen at a latter state, because φ is persistent).

Table 4-2: Parallel version of Chisholm paradox in LCR

S_0		Possible next states	
		φ	(P4)
$O_a(\neg\varphi)$	(P1)	ψ	
$O_a(\varphi \rightarrow \psi)$	(P2)	$\text{viol}(a, \neg\varphi)$	(from P1)
$\psi \rightarrow \varphi$	(P3)	φ	(P4)
		$\neg\psi$	
		$\text{viol}(a, \neg\varphi)$	(from P1)
		$\text{viol}(a, \varphi \rightarrow \psi)$	(from P2)

4.4.3 The backward version of the Chisholm paradox

In this contrary-to-duty situation extra activities are obliged before a certain obligation holds, which are not obliged otherwise (and the negation is then also obliged). In our example, this can be described by the situation in which the following community rules hold: Members must attend group meetings (B1). If one attends a meeting, then one must tell that one is coming (before) (B2). If one does not attend a meeting, then one must not tell that one is coming (B3). In SDL a paradox will occur when one does not attend a meeting (B4). The generic formal specification of this situation is:

- (B1) $O_a(\varphi)$ (a is obliged to attend group meetings)
- (B2) $O_a(\psi \leq \varphi)$ (If a attends, a must tell a is coming, before the meeting)
- (B3) $O_a(\neg\psi \leq \neg\varphi)$ (If a doesn't attend, a must not tell a is coming)
- (B4) $\neg\varphi$

From this specification, one can see that LCR is highly suitable to represent backward contrary-to-duty obligations due to the fact that in LCR deadlines are used explicitly. Because, in LCR, violation of norms is explicitly represented and there is a clear notion of time, the above situation does not result in states where a contradiction holds.

Table 4-3: Backward version of Chisholm paradox in LCR

S_0		Possible next states	
		$\neg\varphi$	(B4)
$O_a(\varphi)$	(B1)	$\neg\psi$	
$O_a(\psi \leq \varphi)$	(B2)	$\text{viol}(a, \varphi)$	(from B1)
$O_a(\neg\psi \leq \neg\varphi)$	(B3)	$\neg\varphi$	(B4)
		ψ	
		$\text{viol}(a, \varphi)$	(from B1)
		$\text{viol}(a, \neg\psi, \neg\varphi)$	(from B3)

The following table portrays the different states originating in this situation, where state S_0 represents a state where the obligations hold and no action has been done and in each state only the relevant propositions are specified. Because ψ must be done before φ , in states where φ holds (B4) either ψ or $\neg\psi$ must already hold.

4.5 Conclusions

Contracts are used in agent societies to indicate agent conformance to some desired interaction patterns and to verify whether desired social states are preserved by agent activity. In this chapter we have introduced LCR, a very expressive logic for describing interaction in multi-agent systems. This logic makes it possible to describe and verify contracts that specify interaction between agents. So far, we have concentrated on the logical foundation for the representation of norms. In chapter 5 we will describe how LCR is used to specify contracts in OperA. Furthermore, in chapter 5 we will extend LCR to represent other OperA society components.

Chapter 5

Formal Model for OperA

*'I see well that never is our intellect satisfied,
unless that truth illumines it
beyond which no truth may soar.'*
- Dante Alighieri, *The Divine Comedy*, 1319.

The role of formal methods is to provide a clear and precise description of what a system is supposed to achieve, rather than a formulation of how it operates. The presence of a formal model supports the use of structured design techniques and formal analysis, facilitating development, composition and reuse [Esteva et al., 2001]. This chapter presents the formal semantics underlying the OperA framework. These semantics is based on the deontic and temporal logic LCR introduced in chapter 4 to specify contracts. In this chapter, LCR is extended to represent roles, scenes, inter-scene relationships and the other society elements.

The work presented here is not a specification language but a semantic theory for OperA. However, we are certain that this theory is expressive enough to enable the construction of a specification language for OperA. Later in this chapter we provide a formal syntax for OperA, as a first step towards the definition of the specification language for OperA. The operationalization of the language is however object of future research.

This chapter is organized as follows. Section 5.1 introduces the basis for the semantics in terms of domain and communication representation. Section 5.2 deals with the logical interpretation of OperA Organizational Models. In section 5.3 we discuss the concept of contract in detail. The logical interpretation of Social Models is presented in section 5.4 and that of Interaction Models in section 5.5. In section 5.6 we discuss verification of OperA models. The formal syntax of OperA is presented in section 5.7. Finally, in section 5.8, we present our conclusions and indicate areas for further research.

5.1 Building Blocks of OperA

This section introduces the basic elements of the formal language, by providing a precise semantics of the basic elements of OperA. These elements enable the further formalization of the Organizational Model in the following section. The starting point for the semantics is the formal logic LCR presented in chapter 4. LCR provides a semantics for achievement and commitment expressions that are used to describe the actions and agreed interaction between role enacting agents. In the following, we extend LCR with primitives needed to represent:

- the domain language (ontology) of an OperA framework,
- the identifiers of society components,
- the application of achievement expressions to role enacting agents,
- the beliefs of role enacting agents.

5.1.1 Application Domain

The first step to give a semantics to the OperA model is to formalize the way domain concepts are described. In principle, domain concepts are formulas in any knowledge representation language and designers are free to choose the representation which best fits their purposes. Such representation is fixed by the communication structure in the OM of the agent society. Model ontologies are formalized in this chapter as a first order logic to represent domain concepts.

Definition 5.1. Domain language

A domain language describing a domain D , is a set of first order formulas L_D , built from a signature $\Sigma = \langle \text{Pred}_D, \text{Func}_D, \text{Id}_D \rangle$ of predicate symbols Pred_D , function symbols Func_D , identifier symbols Id_D (also called constants), and a countably infinite set of variables Var_D , $x, y, z, \dots, x_1, \dots$. Term_D denotes the set of terms built from Σ . Terms in the language are defined inductively on the functions, identifiers and variables, as: $\forall i \in \text{Id}_D, i \in \text{Term}_D, \forall x \in \text{Var}_D, x \in \text{Term}_D. \forall t_1, \dots, t_n \in \text{Term}_D, \forall f \in \text{Func}_D, f(t_1, \dots, t_n) \in \text{Term}_D. L_D$ is defined as follows²⁷

- If $p \in \text{Pred}_D$, of arity n , and $t_1, \dots, t_n \in \text{Term}_D$, then $p(t_1, \dots, t_n) \in L_D$
- If $t_1, t_2 \in \text{Term}_D$, then $t_1 = t_2 \in L$
- If $\varphi \in L$, then $\neg\varphi \in L$
- If $\varphi, \psi \in L$, then $\varphi \wedge \psi \in L$
- If $\varphi \in L$, then $\forall x(\varphi) \in L$

²⁷ In the remainder of this chapter, we will omit the subscript D , whenever the reference to the domain is clear and not ambiguous.

We further define L_+ , as being L without the negation and equality expressions. The domain language L_D provides a formal representation of the domain ontology specified in an OperA model. Specific OperA concepts, described in the society ontology are introduced in the remainder of this chapter as special operators and expressions of the language L_D .

A few extra definitions are due. First, a formula of type $p(t_1, \dots, t_n)$ is called an atom; the set of atoms is denoted by Atom_D . The notions of free and bound variable are as usual. A *ground atom* or *ground term* is an atom or term, respectively, without occurrences of free variables. A formula without free variables is also called a *sentence*. The usual entailment relation for first order logic is denoted by \models . Informally, $\Gamma \models \varphi$ if φ is implied by the set of sentences Γ . Finally, identifiers are actually functions of 0 arity. We choose to separate both sets for the sake of readability.

5.1.1.1 Identifiers in OperA

Having defined a language to talk about the domain of an agent society, we can now go on and define a language for agent societies operating over that domain. In order to be able to refer to the different OperA entities, the following identifiers are defined:

Definition 5.2. OperA Identifiers

Let Id_D denote the set of identifier symbols of a domain language L_D . The sets of agent identifiers, role identifiers and scene identifiers are represented respectively by $Agents_D = \{a_1, \dots, a_n\}$, $Roles_D = \{r_1, \dots, r_m\}$ and $Scenes_D = \{s_1, \dots, s_k\}$, such that

- $Agents_D \subseteq Id_D$,
- $Roles_D \subseteq Id_D$
- $Scenes_D \subseteq Id_D$
- Identifier sets are pair wise disjoint:
 $Agents_D \cap Roles_D = \emptyset$ and $Agents_D \cap Scenes_D = \emptyset$ and $Roles_D \cap Scenes_D = \emptyset$

Role enacting agents (reas) play an important part in OperA. Formally, a role enacting agent, *rea*, is a predicate $rea(a, r, s)$, where $a \in Agents_D$, $r \in Roles_D$ and $s \in Scenes_D$. In the following, we will use the identifiers i, j, k, \dots , to refer to reas. Formally, we define the set of role enacting agents, $Reas_D$, as:

$$Reas_D = \{i \in Id_D: i = rea(a, r, s), \text{ for some } a \in Agents_D, r \in Roles_D \text{ and } s \in Scenes_D\}$$

Furthermore, *rea* identifiers are also pair-wise disjoint with identifiers for roles, scenes and agents.

5.1.1.2 Achievement

The activity of reas brings about particular states of affairs in the world. The dynamic nature of interaction can best be analyzed using a branching time framework, in which states are ordered into a treelike structure, with forward branching representing the openness of the future and the determinacy of the past. The *stit* theory provides a

precise and intuitive semantics for the concept that ‘agent a sees to it that φ ’ [Horty, Belnap, 1995]. We have used similar concepts in the definition of the LCR operator $E_a\varphi$ in chapter 4. A tree structure for the representation of time reflects the alternatives, and therefore the possibility for choice, available for agents. That is, it is possible to model the influence that an agent can exercise upon the course of history. It has been argued that *stit* statements are a most suitable construct to describe the alternatives and choices that assign an action to an agent [Belnap, Perloff, 1988].

Achievement expressions can be seen as the result of abstract actions. If actions are interpreted as functions that map some state of affairs into another one [Dignum, Linder, 1997], *stit* statements describe the intended resulting state of affairs without considering the actual action. Formally, achievement expressions are defined as follows.

Definition 5.3. Achievement expressions

Given expression $\varphi \in L_D+$ and $i \subseteq Reas_D$, the expressions $E_i\varphi \in L_D$ and $D_i\varphi \in L_D$. Furthermore, the expressions $E_r\varphi$ and $D_r\varphi$, where $r \in Roles_D$ indicate an achievement for any *rea* of the role. That is,

1. $E_r\varphi \rightarrow \exists i \in ReasD: rea(i,r,s) \wedge E_i\varphi$
2. $D_r\varphi \rightarrow \exists i \in ReasD: rea(i,r,s) \wedge D_i\varphi$

Informally, $E_i\varphi$ represents the fact that i sees to it that φ , and $D_i\varphi$ the fact that i has done (saw to it that) φ . The semantics of $E_i\varphi$ and $D_i\varphi$ are defined in chapter 4. Note that the evaluation of both $E_i\varphi$ and $D_i\varphi$ is always done in one state, and the fact that in a certain state $E_i\varphi$ (or $D_i\varphi$) holds, does not guarantee that φ will still hold in later states. I.e., the fact that someone saw to it that there is milk in the in the fridge at a certain moment, does not mean that the milk will still be there later. The properties of E and D are formally specified as follows:

Definition 5.4. Achievement axioms

1. $\models E_i\varphi \rightarrow XD_i\varphi$
2. $\models D_i\varphi \rightarrow \varphi$
3. $\models (D_i\varphi \leq \delta) \wedge (D_i\psi \leq \delta) \leftrightarrow (D_i\varphi \wedge D_i\psi) \leq \delta$
4. $\models (D_i\varphi \leq \delta) \vee (D_i\psi \leq \delta) \leftrightarrow (D_i\varphi \vee D_i\psi) \leq \delta$
5. $\models \neg (D_i\varphi \leq \delta) \rightarrow \neg D_i\varphi \leq \delta$

In the remainder of this chapter, is often useful to be able to identify the set of achievement expressions in the language. We therefore define the set Act_D as:

Definition 5.5. Act_D

Given a domain language L_D , the set of all achievement expressions Act_D , is defined by the smallest set such that $\forall \varphi \in L_D, E_i\varphi \in Act_D$.

5.1.1.3 Beliefs

OperA specifies agents societies from a perspective external to the agents and therefore does not provides the means to refer to internal aspects of agents. This entails that the concept of agent's belief does not play a very important role in the semantics of OperA. However, in order to be able to describe the meaning of communicative acts (cf. section 5.2.3), we need to be able to make some minimal assumptions on the architecture of agents, an in particular, we need to specify the meaning of agent beliefs. This because, the semantics of communication is defined based on the epistemic state of the communicating partners²⁸. We are aware that other researchers have proposed different semantics to communication not based on epistemic logic (e.g. [Nickles, Weiss, 2003]) . However, to our knowledge, epistemic logic is still the most common and proven basis for the semantics of communication. Because beliefs do not play a very important role in OperA, we will limit ourselves to the use of the standard modal operator and axioms for belief from epistemic logic [Halpern, Moses, 1992], [Meyer, van der Hoek, 1995].

In the following, we introduce the belief operator $B_i\varphi$ which means that agent i believes φ .

Definition 5.6. Belief

Given an expression $\varphi \in L_D$ and a rea identifier $i \in Reas_D$, the expression $B_i\varphi \in L_D$.

Definition 5.7. Belief axioms

1. $\models B_i(\varphi \rightarrow \psi) \rightarrow (B_i\varphi \rightarrow B_i\psi)$
2. $\models \neg(B_i\varphi \wedge B_i(\neg\varphi))$
3. $\models B_i\varphi \rightarrow B_i(B_i\varphi)$
4. $\models \neg B_i\varphi \rightarrow B_i(\neg B_i\varphi)$

5.2 Logical Interpretation of the Organizational Model

The Organizational Model (OM) of an OperA model consists of the communicative, normative, social and interaction structures of the society. It describes the society structure as desired by society design. In the following, we will provide a formalization of the different components of the OM.

²⁸ This is similar to the way FIPA refers to agents in their Agent Communication Specification. FIPA also does not enforce any specific agent architecture but makes a (minimal) set of assumptions about agents, in order to be able to describe their communication.

5.2.1 Social Structure

The social structure of a society specifies the roles, groups and role dependencies in the society. In the following, we give the formal specification of these elements.

5.2.1.1 Roles

In OperA, roles describe an organizationally-sanctioned structured bundle of activity types. Role descriptions identify the necessary activities and services to achieve society objectives and enable to abstract from the individuals that will eventually perform the role.

Definition 5.8. Role

A role is a tuple $\mathbf{role}(r, \mathbf{Obj}, \mathbf{Sbj}, \mathbf{Rgt}, \mathbf{Nor}, \mathbf{tp})$ where $r \in \mathit{Roles}_D$ is the identifier of role, $\mathbf{Obj} \subseteq \mathit{Act}_D$ is the set of objectives of role, $\mathbf{Sbj} \subseteq \mathit{Act}_D$ is the set of sub-objectives sets of role, $\mathbf{Rgt} \subseteq \mathit{Deon}_D$ are the rights of role, $\mathbf{Nor} \subseteq \mathit{Deon}_D$ are the norms of role, and $\mathbf{tp} \in \{\mathit{external}, \mathit{institutional}\}$ is the type of role.

Given a society S , the set of all roles in that society is identified by R_S . We define the following functions on a role:

- | | |
|---|---|
| - $\mathit{id}: \mathit{R}_S \rightarrow \mathit{Roles}_D$ | $\mathit{id}(\mathbf{role}(r, \mathbf{Obj}, \mathbf{Sbj}, \mathbf{Rgt}, \mathbf{Nor}, \mathbf{tp})) = r$ |
| - $\mathit{objectives}: \mathit{R}_S \rightarrow 2^{\mathit{Act}_D}$ | $\mathit{objectives}(\mathbf{role}(r, \mathbf{Obj}, \mathbf{Sbj}, \mathbf{Rgt}, \mathbf{Nor}, \mathbf{tp})) = \mathbf{Obj}$ |
| - $\mathit{sub-objectives}: \mathit{R}_S \rightarrow 2^{\mathit{Act}_D}$ | $\mathit{sub-objectives}(\mathbf{role}(r, \mathbf{Obj}, \mathbf{Sbj}, \mathbf{Rgt}, \mathbf{Nor}, \mathbf{tp})) = \mathbf{Sbj}$ |
| - $\mathit{rights}: \mathit{R}_S \rightarrow 2^{\mathit{Deon}_D}$ | $\mathit{rights}(\mathbf{role}(r, \mathbf{Obj}, \mathbf{Sbj}, \mathbf{Rgt}, \mathbf{Nor}, \mathbf{tp})) = \mathbf{Rgt}$ |
| - $\mathit{norms}: \mathit{R}_S \rightarrow 2^{\mathit{Deon}_D}$ | $\mathit{norms}(\mathbf{role}(r, \mathbf{Obj}, \mathbf{Sbj}, \mathbf{Rgt}, \mathbf{Nor}, \mathbf{tp})) = \mathbf{Nor}$ |
| - $\mathit{type}: \mathit{R}_S \rightarrow \{\mathit{external}, \mathit{institutional}\}$ | $\mathit{type}(\mathbf{role}(r, \mathbf{Obj}, \mathbf{Sbj}, \mathbf{Rgt}, \mathbf{Nor}, \mathbf{tp})) = \mathbf{tp}$ |

On the properties of roles, OperA assumes that different roles have different objectives and all roles have at least one objective. Formally, this is represented as follows:

Proposition 5.1. Properties of roles

1. $\forall R_1, R_2 \in \mathit{R}_S: \mathit{id}(R_1) = \mathit{id}(R_2) \leftrightarrow \mathit{objectives}(R_1) = \mathit{objectives}(R_2)$
2. $\forall R \in \mathit{R}_S: \mathit{objectives}(R) \neq \emptyset$

Furthermore, sub-objectives for an objective are so defined that the achievement of all the sub-objectives in a set entails the achievement of the objective:

Definition 5.9. Sub-objective set

Let γ be an objective of role r . The set $\Pi\gamma = \{\gamma_1, \dots, \gamma_n\}$ such that $\bigwedge_{i=1}^n \gamma_i \rightarrow \gamma$, where $\gamma_i \in \Pi\gamma$ is a set of sub-objectives of γ . Furthermore, for each role objective γ , finitely

many sets of sub-objectives can be defined: $\{\Pi_1\gamma, \dots, \Pi_m\gamma\}$ corresponding to alternative ways to achieve the objective.

A few observations are due here, concerning the above logical interpretation of an OperA role. With respect to role objectives, it may, in the first instance, seem strange that those are not interpreted as a logical modality in LCR, but as a (generic) achievement expression. In the line of the intuitive relation between role objectives and agent goals, a possibility would be to give objectives a modal interpretation, to represent ideal or most desired states in the society world. In such a model, one could, for example, say that a state where role objectives are achieved is more preferred than a state where they are not achieved. However, since the operational semantics of objectives depend on the way objectives are treated and assumed by the agent acting the role and on the semantics of that agent, such a preference relation at society level does not necessarily have any meaning at the agent level. That is, the properties of a modal operator for objectives are too weak to require a logical interpretation other than that of generic expressions. Furthermore, as the interpretation of the society structure now stands, no reasoning on role objectives is needed, as we do not consider logical relations between objectives of different roles, or do any reasoning on the consequences of objectives. In [Dastani et al., 2003] we present initial research concerning this issue, where a modal interpretation is indeed chosen. In our opinion, the choice for a modal interpretation of objectives, and the consequent preference relations, cannot be taken lightly and its implications for the mental attitudes of the individual agents must be thoroughly studied. This is an issue for further research.

5.2.1.2 Groups

As described in chapter 3, roles can be organized into groups. Groups provide the means to refer collectively to a set of roles and to specify shared norms for the roles in the group.

Definition 5.10. Group

Given a society S and a set of roles R_S in that society, a group is a tuple $\mathbf{group}(g, Rls, Nor)$, where $g \in Roles_D$ is the identifier of the group, $Rls \subseteq \{\rho \in Roles_D: \exists r \in R_S, id(r) = \rho\}$ is the set identifiers of roles in the group and $Nor \subseteq Deon_D$ are the norms for the group.

Given a society S , the set of all groups in that society is identified by G_S .

Remark: We have chosen to use elements of the set of role identifiers, $Roles_D$, as identifiers of groups. This is due to the fact that groups are mainly used to refer to any of the roles in the group, e.g. in a scene script. It makes definitions simpler if one set of identifiers is used. Nevertheless, the concepts of role and group remain quite different, in particular on the fact that groups do not have objectives and roles do. For example, in a University society, the roles of *professor*, *lecturer* and *Ph.D.-student* are defined which are grouped in group *teacher*. A scene script to describe a course requires an enactor of *teacher* to participate. In this society $\{professor, lecturer, Ph.D.-student, teacher\} \subseteq Roles_D$, $\{professor, lecturer, Ph.D.-student\} \subseteq R_{University}$ and

$teacher \in G_{University}$. The definition of scene Course requires a set of role identifiers to indicate the participants in the scene (cf. definition 5.21). In this example, *teacher* is an element of that set.

5.2.1.3 Dependency relations between roles

One of the main issues in OperA is the specification of coordination between role enacting agents in a regulated society environment. Therefore, the representation of relationships between roles is one of crucial importance. Role dependencies indicate the relations between roles, through which objectives can be passed. That is, the dependency relation $r_1 \underline{\phi}_\gamma r_2$ between roles r_1 and r_2 indicates that a rea of role r_1 can pass their objective γ to a rea of role r_2 . Dependency relations identify a pre-order of roles in the society, represented as $\mathfrak{R} = (R_S, \underline{\phi})$, where R_S is the set of roles in a society S . More on dependency relations is discussed in chapter 3. Basically, we identify three types of role dependencies (cf. chapter 3):

1. $r_1 \underline{\phi}_\gamma^H r_2$, representing the hierarchical relation, where rea r_1 delegates ϕ to r_2
2. $r_1 \underline{\phi}_\gamma^M r_2$, representing the market relation, where rea r_2 bids for ϕ to r_1
3. $r_1 \underline{\phi}_\gamma^N r_2$, representing the network relation, where both rea r_1 and r_2 can request the other for ϕ

In organizational systems, it is usual to organize roles in an inheritance hierarchy, or *is-a* hierarchy. In such hierarchies, child roles inherit the characteristics (attributes, rights, norms) of their parent roles. It is important to note that dependency relations as defined in OperA are not inheritance relations, but define the links through which objectives can be delegated to other roles. Coordination of behavior is relatively easy in a hierarchical society, in which case when an agent i enacts a role that is superior to the role that agent j enacts, a request from i will result in an obligation for j . In networks and markets, however, coordination requires some more effort. In general, one can identify three different reasons for an agent to commit itself to a request from another agent [Dignum, Weigand, 1995]:

- **Power:** rea j accepts a request from rea i because of some domination relationship between i and j . This type of relation is standard in hierarchical societies, but can also be explicitly defined between two specific roles, in other types of societies (cf. chapter 3, section 3.3.1.3). Power relations are represented by $power(i, j, \phi)$, indicating that rea i has power over j for ϕ .
- **Authorization:** when rea j has committed itself to i for a certain service, a request from i leads to an obligation when the conditions are met. This relation is established by mutual agreement, e.g. in a (previous) interaction scene, for a certain time and under certain conditions. Although authorization relations can happen in any type of society, they are typical of markets (e.g. an order leads to the obligation to pay when payment is requested). Authorization relations are represented by $auth(i, j, \phi)$, meaning that rea i is authorized by j to do ϕ .

- **Charity:** agent j will answer a request from agent i without being obliged to do so. This is typical in network societies. An obligation arises when the agent answers with a positive commitment.

We will now define the semantics of power and authorization relations. Charity relations do not have a specific operator or semantics, since such relations are completely dependent on the commitment relation (see section 5.2.3 for more about commitments).

5.2.1.3.1 Power relations

Power relations are a special case of the role dependency relations described in chapter 3. Dependency relationships describe how actors can interact and contribute to the realization of the objectives of each other. In other words, a dependency relation, $r_1 \underline{\phi}_\gamma r_2$, between roles r_1 and r_2 for objective γ of r_1 indicates that objective γ can be passed to r_2 , that is, that r_2 can realize objective γ for r_1 .

Definition 5.11. Power relation (cf. definition 3, in chapter 3)

Given expression $\varphi \in L_D$ and reas $i, j \in Reas_D$, the expression $power(i, j, \varphi) \in L_D$. The hierarchical dependency relation $r_1 \underline{\phi}_\varphi^H r_2$ between roles $r_1, r_2 \in Roles_D$ gives rise to a power relation $power(i, j, \varphi)$ between $i, j \in Reas_D$, in all $s \in Scenes_D$, such that $rea(i, r_1, s)$ and $rea(j, r_2, s)$.

The expression $power(i, j, \varphi)$ means informally that i has the power to get j to achieve φ . Power relations are *reflexive*, i.e. each reas has power over itself, and often, but not always, also *transitive*, if reas i has power over reas j and j has power over k , then i has power over k . Moreover, power to demand φ implies power to demand all what can be derived from φ . For example, if i has power to request j to lay the table, i has also the power to request i to lay the plates on the table. Formally, the following axioms hold for the power relation:

Definition 5.12. Reflexivity of power relations

The power relation $power \subseteq Reas_D \times Reas_D \times L_D$ is reflexive. That is, for all $i \in Reas_D$ and $\varphi \in L_D$: $power(i, i, \varphi)$.

5.2.1.3.2 Authorization relations

Authorization relations are in fact dynamic power relations. That is, while a power relation is determined by the structure of the society, authorization relations may be dynamically created by the reas to describe situations when power can be (temporarily) effective. Informally, an authorization, $auth(i, j, \varphi)$ means that i is authorized by j to achieve φ .

Definition 5.13. Authorization relation

Given expression $\varphi \in L_D$ and reas $i, j \in Reas_D$, the expression $auth(i, j, \varphi) \in L_D$.

Furthermore, authorization relations always hold in the case of a power relation. That is, if an agent a has the power to request φ from agent b , then a is also authorized to request b to achieve φ . Formally:

Definition 5.14. Authorization relation axiom

$\neq power(i, j, \varphi) \rightarrow auth(i, j, \varphi)$ holds for all $\varphi \in L_D$ and $i, j \in Reas_D$.

Although it may seem that an authorization does not necessarily establish a power relation of i over j , it does so when φ involves a state to be achieved by j , which is often the case with communicative acts, as we will discuss in section 5.2.3. An example is the authorization relation $auth(Anne, Bob, request(Anne, Bob, pay(Bob, Anne, goods)))$ which means that Anne is authorized by Bob to eventually request Bob to pay her some goods. That is, the authorization creates a power relation of Anne over Bob for that payment. In reality, achieving the requested state will often remove the authorization, that is, once Bob has paid, Anne no longer has the authorization to request that payment.

5.2.1.3.3 Role dependency relations

Role dependency relations depend on the power relations between roles. The way the objective γ in a dependency relation $r_1 \Phi_\gamma r_2$ is actually passed between r_1 and r_2 depends on the coordination type of the society. In hierarchies, the parent role will delegate some of its sub-objectives to its children. In this case, the enactor of a children role can not decide which objectives it will get. In markets, children roles can request the assignment of objectives from the parent role. In this case, the enactors of a children role can choose to take the objectives of their parent such that it best fit their own private goals. In a network relation, both roles are authorized to request the objective from the other. That is, actually the network relation defines a symmetric relation between the two roles, or equivalence. Actors in a network identify with the global goals of the society and therefore will have a personal aim to achieve its objectives.

Definition 5.15. Axioms on dependency relations

Given $r_1, r_2 \in RolesD$, the following axioms hold:

1. $r_1 \Phi_\gamma^H r_2 \rightarrow power(r_1, r_2, \gamma)$
2. $r_1 \Phi_\gamma^M r_2 \rightarrow auth(r_2, r_1, request(r_2, r_1, \gamma))$
3. $r_1 \Phi_\gamma^N r_2 \rightarrow auth(r_2, r_1, request(r_2, r_1, \gamma)) \wedge auth(r_1, r_2, request(r_1, r_2, \gamma))$

Note that axiom 3 above, on the network relation, defines an equivalence relation between r_1 and r_2 , that is $r_1 \Phi_\gamma^N r_2 \leftrightarrow r_2 \Phi_\gamma^N r_1$. This is in accordance with the concept

of network in OperA, which assumes peer relationships between roles. The relation ϕ_γ^N is therefore symmetric. Taking as example the Conference society described in chapter 3, consider the role relationship, *PC-Chair* $\phi_{paper-reviewed}$ *PC-member*. Depending on the coordination type, three cases are possible:

6. PC-Chair $\phi_{paper-reviewed}^H$ PC-member, in which case a power relation holds:
power(PC-Chair, PC-member, paper-reviewed)
7. PC-Chair $\phi_{paper-reviewed}^M$ PC-member, in which case authorization relation holds:
auth(PC-member, PC-Chair, request(PC-member, PC-Chair, paper-reviewed))
8. PC-Chair $\phi_{paper-reviewed}^N$ PC-member, in which case two authorization relations:
auth(PC-Chair, PC-member, request(PC-Chair, PC-member, paper-reviewed))
and auth(PC-member, PC-Chair, request(PC-member, PC-Chair, paper-reviewed)) hold.

In this example, the network dependency relation is probably not adequate, as it indicates an equivalence relation between program chair and PC members, which is not intuitive. The hierarchical relation is the most common, and the market relation can be used to model situations when PC members can apply to review, or indicate their preferences for, certain papers.

5.2.1.4 Social structure definition

Using the definitions above, we are now able to provide the formal definition of a social structure. Social structures include the roles, groups and dependency relations of a society.

Definition 5.16. Social Structure

The social structure of a society S is a tuple $social-structure(R_S, G_S, D_S)$ where R_S is the set of roles in S , G_S is the set of groups in S and D_S the set of dependency relations between roles of S .

5.2.2 Interaction structure

Activity in OperA is articulated through **scenes**, that follow pre-defined abstract **scripts**, and can be organized into an **interaction structure**, that enables the representation of more complex interactions. Transition scripts describe the synchronization and partial ordering of scene scripts and identify the constraints for role evolution along the interaction structure. We use the concept of landmark and landmark pattern introduced in [Kumar et al., 2002], to describe scenes and extend this concept for the representation of scene transitions.

In the following, we first introduce the basic concepts of landmark and landmark pattern and their application to scene scripts and follow with the specification of scene transitions and role evolution relations.

5.2.2.1 Landmarks

Interaction between agents is traditionally specified by means of protocols that fix the communicative acts to be used. However, there may be several communicative actions that result in the same state. Based on the work of [Kumar et al., 2002], OperA uses a **landmark** based approach to represent interaction. As described in chapter 3, interaction scripts provide the minimum requirements and constraints for interaction that are necessary to achieve the interaction results sought by the society design, according to the view of the society. Such approach allows agents to choose the best applicable action, from their own perspective, to achieve those landmarks.²⁹

The concept of landmark was first introduced in [Smith et al., 1998], as defining a set of specifiable semantic properties that must hold of the agents involved (e.g. an offer has been made; an offer has been accepted). A landmark is identified by the set of propositions that are true in the state represented by the landmark. Our formal definition of landmarks is based on Kripke models. This is consistent with the semantics of LCR as described in chapter 4 (cf. definition 3 in chapter 4)³⁰. In the following, landmarks are described as subsets of worlds, and relations between landmarks are specified using LCR operators.

Definition 5.17. Landmark

A landmark $\lambda \in \mathcal{L}_D$ is a conjunction of atomic expressions $\lambda = \{\wedge s : s \in 2^{AtomD} - \emptyset\}$. Given a semantic model $M = (W, R, \pi)$ for \mathcal{L}_D , λ identifies a subset $A \subseteq W$ such that $\forall w \in A: (M, w) \models \lambda$.

Using LCR semantics, landmarks can be partially ordered using the operators \leq (before) and \vee (or) defined in chapter 4. From the semantics of LCR follows that the partial order relation between landmarks is transitive, that is, if λ_1 comes before λ_2 and λ_2 comes before λ_3 , then λ_1 must necessarily come before λ_3 . A partially ordered set of landmarks is called a **landmark pattern**. Landmark patterns are visualized as directed graphs whose nodes represent the landmarks and whose directed arcs

²⁹ In the case of non-intelligent agents, such choices can be fixed in the agent architecture by the agent designer, but the approach allows for the maximal use of the agent's capabilities and autonomy within the constraints and requirements imposed by society design.

³⁰ [Kumar et al., 2002] present an alternative semantics for landmarks. For the sake of consistency with the overall framework, we choose to use Kripke-model semantics as in chapter 4.

represent the partial ordering.³¹ For example, Figure 5-1 shows the graphical representation of the following partial order expressions:

- $\lambda_1 \leq \lambda_2 \leq (\lambda_3 \vee \lambda_4)$
- $\lambda_3 \leq (\lambda_5 \vee \lambda_6)$
- $\lambda_4 \leq \lambda_6$

which can be compacted into $\lambda_1 \leq \lambda_2 \leq ((\lambda_3 \leq (\lambda_5 \vee \lambda_6)) \vee (\lambda_4 \leq \lambda_6))$

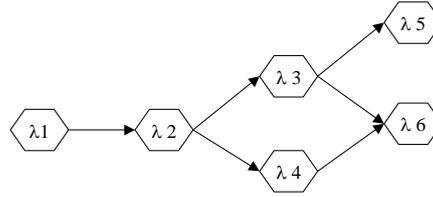


Figure 5-1: Landmark pattern example

5.2.2.2 Scene scripts

In the terms of [Smith et al., 1998], interaction scripts can be seen as conversation patterns, and describe a relative sequence or pattern of landmarks in an interaction. Formally scene scripts are defined as follows:

Definition 5.18. Scene script

A scene script is a tuple $scene(s, Rls, Res, Ptn, Nor)$ where $s \in Scenes_D$ is the identifier of the scene, $Rls \subseteq \{\rho \in Roles_D : \exists r \in R_S, id(r) = \rho\}$ is the set of identifiers of roles in the scene, $Res \subseteq Act_D$ is the set of results of the scene, $Ptn \subseteq Act_D$ is the set of interaction patterns of the scene, and $Nor \subseteq Deon_D$ are the norms of the scene.

Given a society S , the set of all scenes in the society is identified by S_S . Scenes are identified by their name³². We define the following functions on a scene:

- $id: S_S \rightarrow Scenes_D$ $id(scene(s, Rls, Res, Ptn, Nor)) = s$
- $roles: S_S \rightarrow 2^{R_S}$ $roles(scene(s, Rls, Res, Ptn, Nor)) = Rls$
- $results: S_S \rightarrow 2^{Act_D}$ $results(scene(s, Rls, Res, Ptn, Nor)) = Res$
- $patterns: S_S \rightarrow 2^{Act_D}$ $patterns(scene(s, Rls, Res, Ptn, Nor)) = Ptn$

³¹ Note that, in opposition to for instance finite state machines, transition arcs in landmark patterns represent only the ordering of landmarks and not the actions required for state transitions.

³² A note on notation: we use lowercase s, s_1, s_2, \dots to denote scene identifiers (i.e. elements of $Scenes_D$) and uppercase S, S_1, S_2, \dots to denote role tuples (i.e. elements of S_S)

- norms: $S_S \rightarrow 2^{DeonD}$ $norms(scene(s, Rls, Res, Ptn, Nor)) = Nor$

Furthermore, OperA assumes that different scenes have different results. Formally, this is represented as follows:

Proposition 5.2. Properties of scenes

- $\forall S_1, S_2 \in S_S: id(S_1) = id(S_2) \leftrightarrow results(S_1) = results(S_2)$
- $\forall S \in S_S: results(S) \neq \emptyset$

In the interaction model, the pattern of landmarks associated with an interaction script is realized by specifying the protocol consisting of agent actions (actual conversation) for each landmark transition, such that performing those actions, from the source landmark, provably results in the target landmark. The landmark pattern representing an interaction pattern is defined as follows:

Definition 5.19. Results as landmark patterns

Let ρ_1, \dots, ρ_n be the results of a scene script. The landmark pattern for this set of objectives is

$$\lambda_S \leq (\rho_1 \vee \dots \vee \rho_n) \leq \lambda_E$$

where λ_S and λ_E are landmarks, such that $\lambda_S \rightarrow \neg \rho_1 \wedge \dots \wedge \neg \rho_n$ and $\lambda_E \rightarrow \rho_1 \wedge \dots \wedge \rho_n$, denoting the initial and the end state of the landmark pattern, respectively.

The graphic representation of a generic landmark pattern for scene objectives is depicted in Figure 5-2.

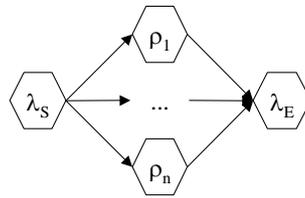


Figure 5-2: Landmark pattern for objectives

Depending on the interaction patterns specified in the scene script for each result, the landmark pattern above can be further refined, taking these interaction patterns as extra constraints on the partial order of scene results. We will illustrate this by the way of an example, the interaction script Review Process described in chapter 3, section 3.4.2.1, and illustrated in Figure 5-3. The results of this script are represented in landmark patterns.

The landmark for result r_1 , Lr_1 , is refined using the landmark pattern describing the interaction pattern for result r_1 as is illustrated in Figure 5-4. The landmark pattern for the interaction script above is therefore formally represented by:

$$\lambda_S \leq ((\lambda_{Sr1} \leq ((\lambda I \leq (\lambda 3 \vee \lambda 4)) \vee (\lambda 2 \leq (\lambda 3 \vee \lambda 5)) \leq \lambda 6 \leq \lambda_{Er1})) \vee Lr2) \leq \lambda_E$$

where the landmarks $\lambda_1, \dots, \lambda_6$ are as follows:

λ_1 : $DONE(O, assign_paper(P, PC1, DeadlineR) \wedge \neg DeadlineA \wedge \neg DeadlineR$

λ_2 : $DONE(O, assign_paper(P, PC2, DeadlineR) \wedge \neg DeadlineA \wedge \neg DeadlineR$

λ_3 : $DeadlineA \wedge \neg DeadlineR$

λ_4 : $DONE(PC1, receive_review(O, P, Rev1) \wedge \neg DeadlineR$

λ_5 : $DONE(PC2, receive_review(O, P, Rev2) \wedge \neg DeadlineR$

λ_6 : $DeadlineR$

Interaction Scene: Review Process	
Roles	PC-Chair (1), PC-member (2..Max)
Results	$r_1 = DONE \forall P \in Papers, paper_reviewed(P, PC1, review1) \text{ AND } paper_reviewed(P, PC2, review2)$ $r_2 = DONE \forall P \in Papers, decision_on_paper(P, decision, review1, review2)$
Interaction Patterns	Pattern(r_1) = { $DONE(O, paper_assigned(P, PC1, DeadlineR) \text{ BEFORE } DeadlineA$), $DONE(O, paper_assigned(P, PC2, DeadlineR), \text{ BEFORE } DeadlineA$), $DeadlineA \text{ BEFORE } DeadlineR$, $DONE(PC1, paper_reviewed(P, PC1, review1) \text{ BEFORE } DeadlineR$), $DONE(PC2, paper_reviewed(P, PC2, review2) \text{ BEFORE } DeadlineR)$ }
Norms	PERMITTED($O, paper_assigned(P, PC, DeadlineA)$) OBLIGED($PC, paper_reviewed(P, Rev) \text{ BEFORE } DeadlineR$) OBLIGED ($O, decision_on_paper(P, D, Rev1, Rev2) \text{ BEFORE } DeadlineD$)

Figure 5-3: Script for Review Process scene

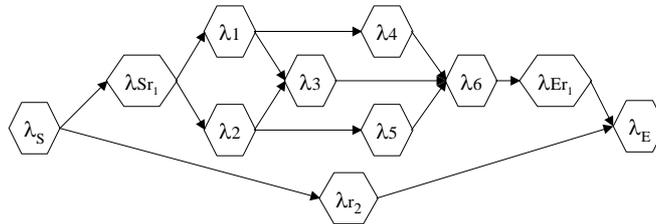


Figure 5-4: Landmark pattern for 'Review Process'

Landmark patterns represent a partial order of states. Of course, not all possible states are represented in a pattern but only those that are relevant for the scene. That is, landmark patterns represent a subset of world states that describe a situation related to the scene. Transitions between states in a landmark pattern occur through the activity of the reas, or because of changes in the world (e.g. time).

5.2.2.3 Scene Transitions

Scene transitions describe the synchronization of scenes and are represented by arcs in an interaction structure model (cf. chapter 3, section 3.3.2.2). In order to define scene transitions, we first introduce the concept of scene connectors, which are labeled arcs between two scenes. Scene connectors describe a partial ordering of scenes, which can be enriched with synchronization mechanisms involving several

connectors. Scene connectors are defined as follows.

Definition 5.20. Scene connector

A scene connector is a tuple **connector**($s, t, \lambda_s, \lambda_t$) where $s, t \in \text{Scenes}_D$ are the identifiers of respectively the source and the target scenes, and $\lambda_s, \lambda_t \subseteq L_D$ are the landmarks describing constraints on respectively the source and target scenes.

The set of all connectors in a society S is denoted by C_S . For simplicity, a scene connector $(s, t, \lambda_s, \lambda_t)$ can be also represented by $s \rightarrow t$. Connections are labeled with identifiers of the roles that can traverse that connection, and possibly, constraints describing the conditions under which such traversing is allowed. The migration of role enacting agents from scene to scene, is dependent on the one hand, on the connections between scenes, and, on the other hand, on the results achieved by the agent in the source scene(s).

Role evolution relations indicate that agents that have successfully enacted a given role in the source scene, are permitted (in the case of a sufficient relation), or obliged (in the case of a necessary relation) to fulfil another role in the target scene. As in chapter 3, we use in the definition above the expression $s.r$ to denote the fact that role r is a role of scene s . Formally, given a scene connection, we define role evolution as:

Definition 5.21. Role evolution relation

Given roles $r_1, r_2 \in \text{Roles}_D$, such that $r_1 \in \text{roles}(s), r_2 \in \text{roles}(t), s, t \in \text{Scenes}_D, s \rightarrow t$, a role evolution relation **role-evolution**($s.r_1, t.r_2, SN, \lambda$), where $SN \in \{\text{necessary}, \text{sufficient}, \text{conflict}\}$ and landmark $\lambda \subseteq L_D$ represents the conditions, indicates that successful enactment of role r_1 in s is necessary or sufficient, respectively, for the enactment of role r_2 in t .

In the trivial case, role enacting agents traversing a connection will keep the same role, that is a role evolution where $r_1 = r_2$ in the definition above. A special case of role evolution is the **conflict** of enactment, that is, the case that one same enactor cannot play two roles. The notion of conflict, already discussed in chapter 3, is formalized as follows:

Definition 5.22. Role conflict

Given scenes $s_1, s_2 \in \text{Scenes}_D$, roles $r_1, r_2 \in \text{Roles}_D$, such that $r_1 \in \text{roles}(s_1), r_2 \in \text{roles}(s_1)$, and $a \in \text{Agents}_D$, the conflict role evolution, **role-evolution**($s_1.r_1, s_2.r_2, \text{conflict}, \lambda$), represented as $r_1.s_1 \otimes r_2.s_2$ represents a conflict between roles r_1 and r_2 . That is, $\forall i \in \text{Reas}_D, \text{rea}(a, r_1, s_1) \rightarrow \neg \text{rea}(a, r_2, s_2)$.

The relation \otimes is symmetric, that is, $\forall r_1, r_2 \in \text{Roles}_D, s_1, s_2 \in \text{Scenes}_D, r_1.s_1 \otimes r_2.s_2 \leftrightarrow r_2.s_2 \otimes r_1.s_1$. A usual use of the conflict relation specifies conflicts of roles in one scene. That is, the case where $s_1 = s_2$ in the definition above. Also, it may be necessary to indicate a global conflict, that is a conflict between roles in all scenes in

which the roles participate. We therefore distinguish the following types of role conflict:

1. A **local conflict** within a scene s is represented by $r_1 \otimes_s r_2$
2. A **global conflict** between roles in different scenes is represented by $r_1 \otimes r_2$

Scene connectors can further be combined into scene transitions, that describe the synchronization between scenes. Scene transitions have different types, modeling the synchronization needs. The graphic description of scene synchronization was described in chapter 3, section 3.3.2.2. In the following, we provide the formal definition of scene transitions.

Definition 5.23. Scene transitions

In a society S , a transition between scenes is a tuple **transition**(*Conn*, *type*), where $Conn \subseteq C_S$ and $type \in \{\text{all-targets, some-targets, one-target, new-target, all-sources, some-sources, one-source}\}$ describes the synchronization, such that one and only one of the following rules holds:

1. $\forall c \in Conn, \exists s \in S_S: \text{source}(c) = s$, or
2. $\forall c \in Conn, \exists s \in S_S: \text{target}(c) = s$

The set of all transitions in a society S is represented by T_S . Informally, each transition relates a set of source scenes with one target scene, or one source scene with a set of target scenes. This is expressed in the following proposition. Depending on the path followed by role enacting agents when traversing an interaction structure, they may either start new scene instances or join existing scenes.

For the completeness of the definition of scene relations, we also need to define the concept of connection path between two scenes to represent the fact that there is an ordered set of connectors that link the two scenes. Formally a connection path is defined as:

Definition 5.24. Connection path

Given two scenes $s_1, s_n \in S_S$, we say that there is a connection path between s_1 and s_n , represented by $s_1 \Rightarrow s_n$, iff

1. $s_1 = s_n$, or
2. $\exists s_i \in S_S: s_1 \rightarrow s_i$, and $s_i \Rightarrow s_n$

5.2.2.4 Interaction structure definition

The formal definition of an interaction structure is derived from the definitions above.

Definition 5.25. Interaction Structure

The interaction structure of a society S is a tuple **interaction-structure**($S_S, s_0, s_\Omega, C_S, T_S, E_S$) where S_S is the set of scene scripts of S , $s_0 \in S_S$ is the initial scene, $s_\Omega \in S_S$ is the end scene, C_S is the set of scene connectors of S , such $\forall s \in S_S$, there is a

connection path from s_0 to s_Ω that passes by s , T_S is the set of scene transitions of S , and E_S the set of role evolutions of S .

We refer to the interaction structure of a society S by I_S . As was discussed in chapter 3, interaction structures must describe partial orderings of scenes starting with the initial scene and ending in the end scene. Therefore the need to impose the restriction that all scenes must be connected to the initial and final scenes. This is expressed in the following property of interaction structures:

Definition 5.26. Interaction Structure

Given an interaction structure $I_S = \text{interaction-structure}(S_S, s_0, s_\Omega, C_S, T_S, E_S)$, the following requirement must hold for all scenes in S_S :

1. $\forall s \in S_S, s_0 \Rightarrow s$, and
2. $\forall s \in S_S, s \Rightarrow s_\Omega$

5.2.3 Communicative structure

Besides a domain ontology, which is formally represented by a domain language as described in definition 5.1, the communicative structure of a society includes the specification of the communicative acts to be used in the society. In chapter 3, we discussed how role performing agents interact in a scene by exchanging messages. We assume that agents are able to exchange messages, which result in the landmarks specified in the scene script, as fixed in their interaction contracts. That is, messages which an agent can emit, count as the abstract CAs as described in chapter 3. An agreed protocol is a meaningful combination of messages that achieves the results of the scene and satisfies its norms and constraints. The basic units of communication between role enacting agents are seen as speech acts³³. Protocols are fixed in interaction contracts.

Illocutionary logic is used to formally describe speech acts [Searle, Vanderveken, 1985]. The illocution of a speech act is the content of the message that the speaker intends to be recognized by the hearer as what the speaker intends to be doing (informing, requesting, agreeing, etc.). The basic concept of illocutionary logic is the **illocutionary act**, which consists of three parts: **propositional contents**, that describe what the speech act is about, **illocutionary context**, that indicates relevant knowledge about the situation in which the speech act is made (who is the speaker, who is the hearer, and at what time and location and other circumstances) and **illocutionary force**, that determines the reasons, and purpose for the communication. That is, the illocutionary force describes what the speaker means by the utterance. E.g. the point of a request is to get others to do things, and the point of promises is to commit the speaker to do something.

³³ In chapter 2, a short introduction to speech act theory is given.

Searle defines the illocutionary force as the combination of the illocutionary point of an utterance, and the particular presuppositions and attitudes that must accompany that point, including the degree of strength of the illocutionary point, mode of achievement, conditions on the propositional content, preparatory conditions, and sincerity conditions. Searle distinguishes five basic illocutionary points [Searle, 1969]:

- **Assertive**: the purpose is to say how things are (according to the speaker)
- **Directive**: the purpose is to get the hearer to do bring about a state of affairs
- **Commissive**: the purpose is to commit the speaker to bring about a state of affairs
- **Declarative**: the purpose is to bring about a state of affairs by saying it (depends of course on the authority of the speaker)
- **Expressive**: the purpose is to express the feelings or attitudes of the speaker.

An important issue is the success of the performance of an illocutionary act. Recall that communication is the way that role enacting agents have to realize the objectives of an interaction scene. The successful realization of an interaction scene is therefore dependent on the success of the performance of speech acts between reas. Obviously, the success of an utterance depends on the (physical) conditions of speaking and understanding. That is, is there a medium for communication (e.g. a telephone), or is the hearer paying attention. These conditions are however of little theoretical interest. When referring to the success of an utterance we are mainly concerned with illocutionary force [Verharen, 1997].

In order to formalize interaction protocols described in OperA and verify their fitness to the formal specification of the interaction scene, it is necessary to give a formal semantics to the CAs used to compose those protocols. As in chapter 3, we assume that role enacting agents use FIPA ACL as communication language. FIPA ACL has a formal semantics which is based on speech acts [FIPA, 2002]. The basic illocutions in FIPA ACL, from which a whole library of CAs has been developed, are **request**, **commit**, **inform** and **declare**.

Definition 5.27. Basic communicative acts

Given $i, j \in Reas_D$ and an expression $\varphi \in L_D$, a communicative act $ILL(i, j, \varphi) \in L_D$, where $ILL \in \{request, commit, inform, declare\}$ is the illocutionary point, i is the speaker (or sender), j is the hearer (or receiver) and φ is the content. The following expressions are basic communicative acts:

- $request(i, j, \varphi)$
- $commit(i, j, \varphi)$
- $inform(i, j, \varphi)$
- $declare(i, j, \varphi)$

Usually, semantics of communicative acts are specified in terms of action logics [Dignum, Greaves, 2000]. This way of specification of semantics takes an agent perspective, in the sense that it describes the effects of communication in terms of the

actions available to the agent. In OperA we take a more abstract view on communication, that is, we do not look at communication as an action, but merely consider the (externally observable) effects of the communication. We are well aware that this choice does not allow for a very rich formalization of communication, but it results in a simpler logic which is enough to describe the intended effects for our purposes. Further research is needed to combine OperA with formal agent descriptions, which will result in a more complex description of communication.

In OperA, communication between agents is seen as expressing or requesting achievement statements. That is, agents can request or commit to ‘see to it that a given state of affairs holds’ (cf. section 5.1.1.2). The semantics of communicative acts in OperA is therefore specified in terms of achievement expressions. As in chapter 4, we choose not to represent communicative acts as actions but as the description of the resulting state³⁴. The formal definitions of the above communicative acts are:

- $\text{request}(i, j, \varphi) \quad =_{\text{def}} \quad E_i \text{ requested}(i, j, \varphi)$
- $\text{commit}(i, j, \varphi) \quad =_{\text{def}} \quad E_i \text{ committed}(i, j, \varphi)$
- $\text{inform}(i, j, \varphi) \quad =_{\text{def}} \quad E_i \text{ informed}(i, j, \varphi)$
- $\text{declare}(i, j, \varphi) \quad =_{\text{def}} \quad E_i \text{ declared}(i, j, \varphi)$

In the remainder of this chapter, it is often useful to be able to identify the set of communicative expressions in the language. We therefore define the set $\text{Comm}_{\mathcal{D}}$ as:

Definition 5.28. *Comm_D*

Given a domain language $L_{\mathcal{D}}$, the set of all communicative acts $\text{Comm}_{\mathcal{D}}$, is defined as:

1. All basic speech acts are elements of $\text{Comm}_{\mathcal{D}}$
2. If $t \in \text{Comm}_{\mathcal{D}}$ then also $\text{ILL}(i, j, t) \in \text{Comm}_{\mathcal{D}}$, $\text{ILL}(i, j, \neg t) \in \text{Comm}_{\mathcal{D}}$,
where $\text{ILL} \in \{\text{request}, \text{commit}, \text{inform}, \text{declare}\}$

The intended effects of communicative acts are described by means of deontic and epistemic operators, and using the dependency relations between agents. For instance, a request uttered in the case that there is a power or authorization relation, will have a different meaning than in case such relation does not exist. In the following, we formally define the intended effects of communicative acts:

Definition 5.29. *Axioms for communicative acts*

1. $\models \text{commit}(i, j, \varphi) \rightarrow XO_{ij}\varphi$
2. $\models \text{request}(i, j, \varphi) \wedge \text{power}(i, j, \varphi) \rightarrow \text{commit}(j, i, \varphi)$
3. $\models \text{request}(i, j, \varphi) \wedge \text{auth}(i, j, \text{request}(i, j, \varphi)) \rightarrow \text{commit}(i, j, \varphi)$
4. $\models \text{request}(i, j, \varphi) \wedge \neg(\text{power}(i, j, \varphi) \vee \text{auth}(i, j, \text{request}(i, j, \varphi))) \rightarrow$

³⁴ To be completely in line with the semantic meaning, we should talk of *states achieved by communication*. However, for readability and to keep to standard notions, we choose to use the most common term of communicative acts.

$$XO_j(\text{accept}(j, i, \varphi) \vee \text{refuse}(j, i, \varphi))$$

5. $\models \text{inform}(i, j, \varphi) \rightarrow XB_j(B_i\varphi)$
6. $\models \text{declare}(i, j, \varphi) \wedge \text{power}(i, j, \varphi) \rightarrow X\varphi$
7. $\models \text{declare}(i, j, \varphi) \wedge \text{auth}(i, j, \text{declare}(i, j, \varphi)) \rightarrow X\varphi$
8. $\models \text{power}(i, j, \varphi) \rightarrow \text{auth}(i, j, \text{request}(i, j, \varphi))$
9. $\models \text{power}(i, j, \varphi) \rightarrow \text{auth}(i, j, \text{declare}(i, j, \varphi))$

These axioms describe how obligations can arise for an agent: by means of a request based on a power or authorization relation, or by committing itself. Furthermore, in the axioms above, it is assumed that agents are sincere, and their intentions are reflected in the communicative act used (e.g. case 5, on beliefs). However, we will not pursue these issues further.

To illustrate the effect of communication between roles, we will use the example introduced in section 5.2.1.3.3 concerning a dependency between the program chair and PC member, *PC-Chair* $\underset{\text{paper-reviewed}}{\Phi}$ *PC-member*:

1. In the case of a hierarchical relation, $\underset{\text{paper-reviewed}}{\Phi}^H$, the power relation $\text{power}(\text{PC-Chair}, \text{PC-member}, \text{paper-reviewed})$ holds. Therefore, according to axiom 2 above, a request from PC-Chair to PC-member for paper-reviewed results in a commitment for PC-member: $\text{commit}(\text{PC-member}, \text{PC-Chair}, \text{paper-reviewed})$.
2. In the case of a market relation, $\underset{\text{paper-reviewed}}{\Phi}^M$, the authorization relation $\text{auth}(\text{PC-member}, \text{PC-Chair}, \text{request}(\text{PC-member}, \text{PC-Chair}, \text{paper-reviewed}))$ holds. Therefore, according to axiom 3 above, a request from PC-member to PC-Chair for paper-reviewed results in a commitment for PC-Chair: $\text{commit}(\text{PC-Chair}, \text{PC-member}, \text{paper-reviewed})$, which means that the PC-Chair commits to have PC-member, achieve that objective.
3. In the case of a network relation, $\underset{\text{paper-reviewed}}{\Phi}^N$, both authorization relations $\text{auth}(\text{PC-Chair}, \text{PC-member}, \text{request}(\text{PC-Chair}, \text{PC-member}, \text{paper-reviewed}))$ and $\text{auth}(\text{PC-member}, \text{PC-Chair}, \text{request}(\text{PC-member}, \text{PC-Chair}, \text{paper-reviewed}))$ hold. Therefore, according to axiom 3 above, a request from PC-Chair to PC-member for paper-reviewed results in a commitment for PC-member: $\text{commit}(\text{PC-member}, \text{PC-Chair}, \text{paper-reviewed})$, and a request from PC-member to PC-Chair for paper-reviewed result in a commitment for PC-Chair: $\text{commit}(\text{PC-Chair}, \text{PC-member}, \text{paper-reviewed})$.

Besides the basic communicative acts, other relevant messages are **accept**, **refuse**, **cancel**, **failure** and **authorize**, which are defined as abbreviations based on the basic communicative acts. Furthermore, communicative acts cause effects on the current world, by creating or removing obligations for reas:

- $\text{accept}(i, j, \varphi) \quad =_{\text{def}} \quad \text{inform}(i, j, \text{commit}(i, j, \varphi))$
- $\text{refuse}(i, j, \varphi) \quad =_{\text{def}} \quad \text{inform}(i, j, \neg\text{commit}(i, j, \varphi))$

- $\text{cancel}(i, j, \varphi) \quad =_{def} \quad O_{ji}\varphi \rightarrow \text{inform}(i, j, \neg O_{ji}\varphi)$
- $\text{failure}(i, j, \varphi) \quad =_{def} \quad O_{ij}\varphi \rightarrow (\text{inform}(i, j, \neg\varphi \wedge \neg O_{ij}\varphi))$
- $\text{authorize}(i, j, \varphi) \quad =_{def} \quad \text{declare}(i, j, \text{auth}(j, i, \varphi))$

Landmark patterns, as described in section 5.1.1.2, are used to describe abstract conversation protocols, that is, interaction patterns in a scene script. In an interaction contract, those interaction patterns are ‘personalized’ to the specific reas involved, by describing the communicative acts that will achieve the interaction results. Formally, concrete conversation protocols are *realized* from a landmark pattern by specifying communicative acts for each landmark transition, such that the achievement of those communications provably results in the target landmark.

5.2.4 A language for communication and contracts: Illocutionary LCR

The LCR language introduced in chapter 4, is meant for the representation of commitments between (groups of) agents. Commitments are specified as clauses in a contract and function as constraints on the role enactment relationship between agent and roles. The language as described in chapter 4 is however not rich enough to represent communication between agents, which as mentioned before is the way interaction is interpreted in OperA. The previous sections enable us to define an extension to the language LCR that incorporates illocutionary acts. This language, Illocutionary LCR (ILCR) has the same form as LCR, as described in chapter 4, but it includes communicative acts as specified in section 5.2.3.

Definition 5.30. Illocutionary LCR (ILCR)

Given the language LCR, the communication deontic temporal language ILCR is defined as follows:

1. *Every expression φ of LCR is an expression of ILCR*
2. *If φ is a state formula in LCR and $i, j \in \text{Ags}$, then $\text{inform}(i, j, \varphi)$, $\text{commit}(i, j, \varphi)$, $\text{declare}(i, j, \varphi)$ and $\text{request}(i, j, \varphi)$ are formulas of ILCR.*
3. *If φ, ψ are formulas of ILCR and $i, j \in \text{Ags}$, then so are $\neg\varphi$, $\varphi \wedge \psi$, $E_i\varphi$, $A\varphi$, $\varphi U \psi$, $\varphi \leq \psi$, $X\varphi$, $\text{viol}(i, \varphi, \psi)$, $\text{inform}(i, j, \varphi)$, $\text{commit}(i, j, \varphi)$, $\text{declare}(i, j, \varphi)$ and $\text{request}(i, j, \varphi)$*

The semantics of ILCR are the extension of the semantics of LCR with the formalization of the communicative acts as described in section 5.2.3.

5.2.5 Normative expressions

Norms regulate the internal and external behavior of agents in a society. In OperA we focus on norms concerning the external behavior of agents, which we categorize into three types: role, scene and transition norms. In chapter 4 we have introduced a deontic logic specific to model interaction contracts consisting of deontic expressions with deadlines. In the following we describe how LCR is used to model normative expressions in OperA.

We assume that norms can be categorized in three types: obligations, prohibitions and permissions. Obligations and prohibitions restrict the possible actions (are derived situations) while permissions are generally used to indicate the conditions on which an action can be performed. Permissions are also closely related to the rights of an role or agent.

Definition 5.31. Normative expressions

Given an expression $\varphi \in L_D$ and a rea $i \subseteq Reas_D$, the expressions $O_i\varphi$, $P_i\varphi$, $F_i\varphi \in L_D$.

Furthermore, the expressions $O_r\varphi$, $P_r\varphi$ and $F_r\varphi$, where $r \in Roles_D$ are defined using the definition of normative expression for reas above as $\forall i: rea(i,r) \rightarrow O_i\varphi$ (respectively $P_i\varphi$ and $F_i\varphi$).

Informally, $O_i\varphi$ represents an obligation for rea i to achieve φ , $P_i\varphi$ represents the permission for rea i to achieve φ , and $F_i\varphi$ represents the prohibition for rea i to achieve φ . The semantics of $O_i\varphi$ are as defined LCR (cf. chapter 4). The expressions $P_i\varphi$ and $F_i\varphi$ are defined as abbreviations:

Definition 5.32. Prohibitions and permissions

1. $P_i\varphi =_{def} \neg O_i\neg\varphi$
2. $F_i\varphi =_{def} O_i\neg\varphi$

The first axiom above, stipulates that everything that is not forbidden is permitted. Note that permission is here not interpreted as an *authorization*, but we take the weak definition of permission (as in Dynamic Deontic Logic) in which permission to do φ means that there is no obligation to do $\neg\varphi$. A special case is the operator $auth(i,j, \varphi)$ introduced in section 5.1.1.2. In this case, we do need to assure that the rea that receives such request is permitted to achieve the requested state, that is

$$|= auth(i,j, \varphi) \rightarrow P_i\varphi$$

In the remainder of this chapter, it is often useful to be able to identify the set $Deon_D$ of deontic expressions in the language.

Definition 5.33. $Deon_D$

Given a domain language L_D , the set of all deontic expressions $Deon_D$, is defined as:

1. $\forall\varphi \in L_D, OPF_i\varphi \in Deon_D$
 2. If $\alpha \in Deon_D$ then also $OPF_i\alpha \in Deon_D$
- where $OPF \in \{O, P, F\}$

5.2.6 Formal representation of Organizational Model

We are now able to give a formal specification of the Organizational Model of an OperA society. This is based on the definitions of domain language (definition 5.1),

social structure (definition 5.15), interaction structure (definition 5.25) and communicative acts (definition 5.28).

Definition 5.34. Organizational Model

Given a domain language L_D , an Organizational Model is defined as a tuple $OM(L_D, Illoc, SocStruct, IntStruct)$, where $Illoc \subseteq Comm_D$, is the set of illocutions defined for the society, $SocStruct = social-structure(R_S, G_S, D_S)$, is the social structure of the society, and $IntStruct = interaction-structure(S_S, s_0, s_\Omega, C_S, T_S, E_S)$ is the interaction structure of the society. We denote the an organizational model by OM_D .

5.3 Using Contracts to Model Interaction in OperA

In this section we introduce the formal concept of contract. Contracts are dynamic entities that can be negotiated, accepted, executed and fulfilled. We analyze the contract lifecycle in the light of OperA societies. The section ends with a formal definition for contracts in OperA that is based on the LCR language.

5.3.1 Contracts and normative systems

A contract is a statement of intent that regulates behavior among organizations and individuals [Morciniec et al., 2001]. Contracts formally specify the behavior that each contractual party is expected to follow in an ideal world. But contracts usually also describe what should happen in sub-ideal situations that occur when one or more parties do not fulfil their contractual commitments. Contracts are composed of a **informative** section, describing the parties, the contract validity and often the normative system of reference (i.e. the legal or institutional system underlying the contract), and a **behavioral** section, consisting of a set of normative statements describing the expected behavior of the parties. Contracts are a way to describe the norms that regulate the behavior of agents in a system. Norms that are the result of contractual commitments are also called **endogenous**. Norms defined by a normative system of reference are called **exogenous**. Exogenous norms can be seen as authoritative determinations of ideal behavior by the underlying institution [Sallé, 2002]. Furthermore, contractual relationships exhibit the following characteristics:

1. **Conflict avoidance:** Sanctions and the desire to avoid them are a practical reason for parties to comply with the commitments fixed in a contract. Rational agents are assumed to consider both the commitments and the sanctions when deciding their actions.
2. **Highly distributed.** This introduces differences in the view that each contractual agent has on the contractual commitments.

The use of languages based on deontic logic to specify the behavioral components of a contract has its roots on the work of von Wright [von Wright, 1950]. In our work, a formal representation of normative statements is given using LCR (cf. chapter 4). In OperA, norms are first class entities which can be followed or violated by the agents. Although we consider the possibility of normative agents [Castelfranchi et al. 2000],

which consciously are able to deliberate about norms, this is not a requirement on agents in OperA systems: norm compliance or violation can as well be the result of ignorance or inability of the agent to understand or adapt to the norm.

It is important to note that, in this view, norms are quite different from constraints, because we assume that agents follow a deliberative approach and can therefore reason about the norms and choose to violate them when that better fits their goals. This is fundamentally different from, e.g. the work of Boman, which considers norms as constraints on the behavior [Boman, 1999]. In his work norms are used to ensure that the agent takes no actions that violate the norms, whereas in our work norms define boundaries for agent action.

Current models and implementations of multi-agent systems often make use of implicit (hard-coded) information to represent shared context which makes interoperability of heterogeneous agents difficult [Dellarocas, 2000]. Contracts have been proposed as means to make explicit the way agents interact with and within the society [Pacheco, Carmo, 2003]. Contracts provide a means to specify and eventually verify that the required global properties of the society will indeed emerge from the interactions between agents. This view is similar to the architecture proposed by [Andrade et al., 2001] for the coordination of software components.

5.3.2 Contract lifecycle

Contracts are dynamic entities that follow a well defined lifecycle. The lifecycle consists of different phases during which contracting parties are found, the contract terms are negotiated, the contract is executed and finally discharged, successfully or otherwise. Figure 5-5 depicts the different stages of a contract.

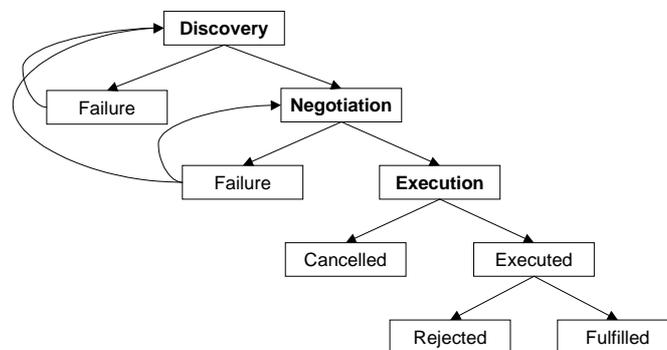


Figure 5-5: Contract lifecycle phases

The main phases of a contract can be described as follows [Reeves et al., 2002], [Kollingbaum, Norman, 2002]:

- **Contract discovery phase:** Agents identify their interaction needs and engage in the search and evaluation of eventual partners. In this phase the contractual roles, abstract business interactions and contractual situations are specified. The

template typically has a number of free variables that are agreed upon in the next phase.

- **Contract negotiation phase:** Participants assume contract roles and negotiate the details of their responsibilities. The negotiable variables of the contract (deadlines, order of actions) become fixed and concrete business interactions are bound to the abstract ones defined in the template. The relationships between contract parties are created and captured in contract statements. The statements contain policy expressions that imply obligations and rights of parties.
- **Contract execution phase:** Actual delivery of contract consideration takes place. Typically, this phase constitutes service or goods delivery, invoicing, bill calculation, presentment and payment. The interactions between the parties are monitored for their compliance to the terms agreed on in the contract.

The formal language LCR is suitable to represent and reason about contracts and their execution but is not enough to represent the initial states of the contract lifecycle. LCR was therefore extended in section 5.2.4, with communication formalisms that enable the representation of contract negotiation. In this section, we assume contracts to be in the execution stage and do not look at the process through which a contract has come to exist.

5.3.3 Contract Specification

The interaction structure as specified in the OM, gives a partial ordering of interaction scenes and the consequences of transitions from one to the other scene to the roles (creation, destruction, etc). That is, the interaction structure indicates the possibilities of an agent at each stage and the consequences of its choices. Explicit representation of commitments is necessary in order to know how to proceed if the norm is violated but first of all to inform the agents about the behavior they can expect from the other agents. In this way, coordination can become possible. A formal language for contract specification, such as LCR, allows the evaluation and verification of the interaction between role enacting agents and their ‘compliance’ to the characteristics specified in the scene script.

5.3.3.1 Generic contracts

Contracts in OperA are first class objects used to specify both the interaction of members with the rest of the society – the *social contracts*, that describe rights and obligations of role enacting agents, reas; and the specific interactions between members – the *interaction contracts* that describe a the way specific participants have agreed to enact a scene. Nevertheless, both types of contracts are basically the same, from a deontic point of view. An important attribute of a contract is its stage. From a social perspective, some stages are desirable and others not. Sanctions are mechanisms of social control used to influence contract parties to maintain the contract in a desirable stage, or to recover from an undesirable stage. In the following we give a formal definition of generic contracts in LCR.

Definition 5.35. Contract.

A **contract**, C , is a tuple $C = (A, CC, S, T)$ where A is a set of agents, CC is a set of contract clauses (expressed as LCR formulae), S is the set of possible **stages** of the contract, and T gives the **transition rules** between states.

Transition rules are events in the domain that alter the status of the contract. Alternatively, a **stage graph** can be used to describe allowed states of the contract and the transition rules between states. The state graph represents the possible evolution(s) of the contract and the consequences of the changes in state to the different parties. Contract clauses are deontic expressions of LCR, that is, obligations, permissions or prohibitions, and as such may indicate deadlines and/or conditions.

5.3.3.2 Example

The following example is intended to illustrate the use of LCR to represent the interactions between two agents in an agent society. This contract expresses an exchange commitment agreed between agents S (a seller) and B (a buyer): S has agreed to sell B a bicycle for €500. S has 2 days to give the bicycle to B after which B must pay S within 1 day. If S does not provide the bicycle on time, then the exchange will not go through. If B does not pay on time then an extra €10 is due within 2 days. Formally, this contract is specified as $C = (\{S, B\}, \{cc1, cc2, cc3, cc4\}, \{s0, s1, \dots, s8\}, T)$, where the contract clauses are defined as³⁵:

cc1: $O_S(\text{get-goods}(B, \text{bicycle}) \leq \text{day2})$

cc2: $O_B(\text{get-money}(S, \text{€}500) \leq \text{day1} \mid D_S(\text{get-goods}(B, \text{bicycle}) \leq 2 \text{ days}))$

cc3: $O_B(\text{cancel-deal}(S, B, \text{bicycle}, \text{€}500) \leq \text{day1} \mid \text{viol}(S, \text{get-goods}(B, \text{bicycle}), \text{day2}))$

cc4: $O_B(\text{get-money}(S, \text{€}10) \leq \text{day2} \mid \text{viol}(B, \text{get-money}(S, \text{€}500), \text{day1}))$

Contract states $\{s0, s1, \dots, s8\}$ and transition graph T are depicted in Figure 5-6.

³⁵ Note that, for example, ‘day2’ abbreviates the expression that denotes a time 2 days from now (the moment at which the contract is agreed), and is true only at that point in time. A more detailed treatment of real time deadlines can be found in [Dignum et al., 1996].

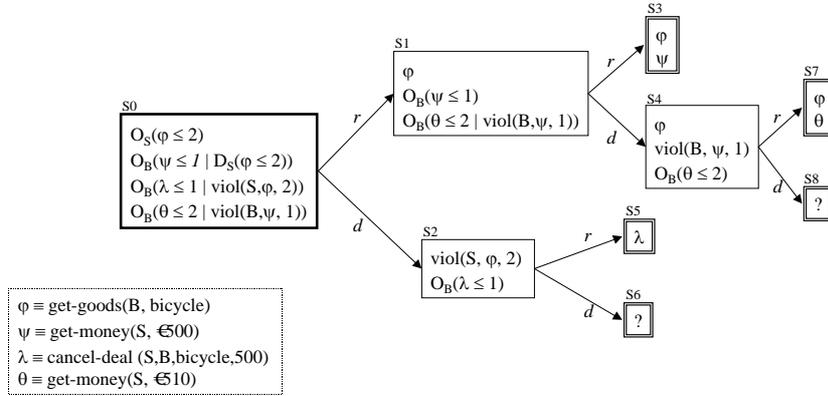


Figure 5-6: Example of a contract

This contract transition graph represents the possible evolution of the contract and the consequences of the changes to the different parties. Transitions between contract stages are expressions representing events in the agent society. In the figure, arrows labeled r represent the achievement of the result and arrows labeled d represent that the deadline has passed without result. In each box only the relevant propositions are displayed. A bold box is the initial stage and double lined boxes are final stages. Stages S6 and S8 are not specified in the contract. Since most contracts are not exhaustive, such stages will probably appear in every contract graph. Consequences of reaching such stage can be determined by society norms (for example, guilty agent is expelled from society).

5.3.4 Contract Negotiation and Execution

In the previous section, a complete formalization for the Organizational Model of an OperA society was presented. We now go on with the formalization of the Social and Interaction Models. Both the Social Model and the Interaction Model of OperA societies are based on contracts. Social contracts in the Social Model describe the relationship between agents and the society, as job contracts in the human society, and interaction contracts in the Interaction Model describe relations between role enacting agents within the society. In this section, we give general considerations on contracts as preparation for the following sections, where the semantics of the Social Model and Interaction Model will be presented.

The LCR language introduced in chapter 4 is suitable to represent contracts and their execution. Both social contracts, describing role enactment, as well as interaction contracts, describing agent interaction, can be specified in LCR. Contracts originate from a process of negotiation that requires communication between the parties. In our work, we are mainly concerned with the negotiation and execution stages. In the following, we will describe the specific aspects of negotiation and execution of social and interaction contracts in OperA.

In general, the main problem with contract negotiation is to determine what is to be negotiated. Two basic components are distinguished when considering negotiation:

the **negotiation mechanism** or protocol and the actual **negotiation strategies** to be employed by the participating agents [Lomuscio et al., 2001]. The negotiation mechanism specifies the ‘rules of encounter’ between the participants, that is, what deals can be made and what sequences of offers are allowed. The negotiation strategies used by the participating agents are part of the agent’s architecture and therefore outside the scope of our work.

In OperA, the role and scene descriptions in the OM provide the scope and content for the negotiation of respectively, social and interaction contracts. That is, the aim of the negotiation of a social contract is to reach an agreement concerning the enactment of a role, according to the role specification, and negotiation of interaction contracts will result in an agreement concerning the playing of a scene, following the landmarks specified in the scene script. A negotiation language must therefore enable the definition of the content and scope of the negotiation and the rules and implications of negotiation actions. Furthermore, the negotiation language must enable the specification of conversation protocols through which agents can agree, propose, commit and reject each other’s suggestions. Using the ILCR language introduced in section 5.2.4, contracts can be negotiated.

Using the formal definitions of the communicative acts, contract clauses are negotiated as follows. Given two negotiating parties $i, j \in \text{Reas}_D$, and φ the issue being negotiated, a contract $C = (\{i, j\}, \{O_{ij}(\varphi < d)\})$ is generated as follows from the dialogue in Figure 5-7. In this conversation, the inform illocutions indicate that both parties agree and are aware of the obligation of the first party. The effect of the conversation on party i is the addition of the obligation, and on party j , the belief that the obligation holds for i . The resulting contract represents the common knowledge of the parties concerning the obligation. Note that this is a mere example of how contract clauses are negotiated. More complex dialogues will be needed to negotiate conditional obligations and sanctions.

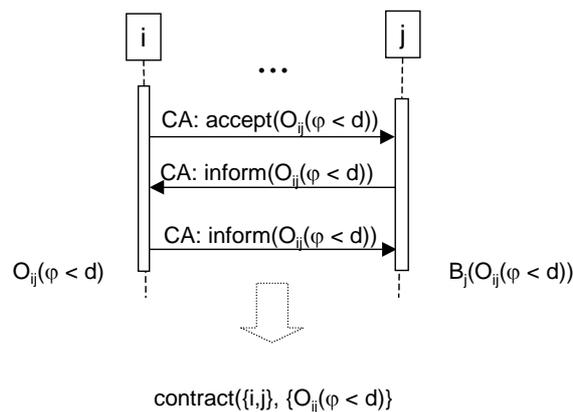


Figure 5-7: Dialogue for contract negotiation

ILCR incorporates communication acts into the LCR language and therefore enables the representation of a contract negotiations. In the next sections, the generic

contract definition given in definition 5.35 will be refined for social and interaction contracts, respectively.

5.4 Logical Interpretation of the Social Model

Social contracts describe the agreed behavior of agents as enactors of society roles. The semantics of social contracts are the same as those of regular contracts (described in chapter 4), but in the case of a social contract one party is the society itself. In practice this means that only one party, the agent, will be actively involved in the execution of the contracts, whereas the other party, the society, is responsible to provide the necessary ‘background’ (scenes, facilitation roles, etc.).

Negotiation mechanisms for social contracts are described in OperA in the *role negotiation scenes*, as described in chapter 3, section 3.4.1.1. The negotiation mechanisms of interaction contracts are specified in these scene scripts. The following definition of social contract is taken from chapter 3 (definition 3.10):

Definition 5.36. Social Contract

Given a society S in a domain D , and an agent $a \in \text{Agents}_D$, a **social contract**, SC , is defined as a tuple $SC(a, r, CC)$ where $r \in R_S$ is a role and $CC \subseteq \text{Deon}_D$ is a set of contract clauses. A social contract is denoted by **social-contract**(a, r, CC).

We are now able to give the formal definition of a Social Model. This definition is relative to an existing Organizational Model and an existing set of agents.

Definition 5.37. Social Model

Given a domain language L_D and a Organizational Model, OM_D , defined on that domain, a Social Model is a tuple $SM(OM_D, Agts, SCs)$ where $Agts \subseteq \text{Agents}_D$ and $SCs = \{\text{social-contract}(a, r, CC) : a \in Agts, r \in \text{roles}(OM_D)\}$ is a set of social contracts between elements of $Agts$ and roles in OM_D .

A Social Model, SM_D , is complete if there is a social contract for all roles, and is incomplete otherwise. Formally:

Definition 5.38. Complete SM_D

A social model $SM(OM_D, Agts, SCs)$ is complete iff: $\forall r \in \text{roles}(OM_D), \exists c \in SCs : c = \text{social-contract}(a, r, cc)$.

5.5 Logical Interpretation of the Interaction Model

The behavior of a society is manifest through the (communicative) actions of the agents enacting its roles. The behavior of an organization is therefore associated with:

- The objectives of the society (represented by role objectives)

- The current role enacting agents, and the commitments fixed in their social contracts

In terms of the landmark framework, interaction contracts describe how an agent must perform the transitions in the landmark pattern associated with an interaction scene. That is, an interaction contract describes a particular realization of the landmark pattern. Usually, patterns are only fully executable when all the roles involved in the interaction are instantiated to actual agents. In the interaction script, results and interaction patterns are described in terms of the desired situations to be achieved. However, agents interact through communicative acts which are not directly or explicitly related to the landmarks specified in the scene script. The first step to formalize the interaction model is therefore to relate the communicative acts of the agents, to the landmarks of the scene, that is to describe agent action in the context of the society. Another approach, *action logic*, uses the *count-as* operator, to formalize this idea [Jones, Sergot, 1996].

5.5.1 Interaction contracts

In the same way that social contracts in the SM provide an instantiation of the roles specified in the OM to specific agents, the interaction contracts in the IM are the means to operationalize the interaction scripts specified in the OM. The following definition of interaction contract is taken from chapter 3 (definition 3.14):

Definition 5.39. Interaction Contract

Given a society S in a domain D and a scene $s \in \text{Scenes}_D$, an **interaction contract**, IC , is defined as a tuple $IC(A, s, CC, P)$, where A is a set of agents, such that $A = \{a \mid \text{rea}(a, r, s) \text{ and } r \in \text{roles}(s)\}$, CC is a set of contract clauses and P the protocol to be followed. An interaction contract is denoted by **interaction-contract** (A, s, CC, P) .

We are now able to give a formal definition of an Interaction Model. This definition is relative to an existing Social Model, containing the social contracts for an agent population of an Organizational Model.

Definition 5.40. Interaction Model

Given a domain language L_D and a Social Model, SM_D , defined on that domain, an **Interaction Model** is a tuple $IM(SM_D, ICs)$ where $ICs = \{\text{interaction-contract}(\text{Parties}, \text{scene}, CC) \mid \text{Parties} \subseteq \text{Reas}_D, \text{scene} \in \text{Scene}_D\}$ is a set of interaction contracts between reas defined in the SM_D .

An Interaction Model, IM_D , is complete if there is an interaction contract for every scene script, and is incomplete otherwise. Formally:

Definition 5.41. Complete IM_D

An interaction model $IM(SM_D, ICs)$ is complete iff $\forall s \in \text{scenes}(OM_D) \exists c \in ICs: c = \text{interaction-contract}(\text{Parties}_s, s, cc)$, where $\text{Parties}_s = \{\text{rea}(a, r, s) \mid r \in \text{roles}(s)\}$.

5.6 On the Verification of OperA Models

Open systems allow agents to enter and exit the system dynamically and unpredictably, while closed systems employ a fixed set of agents that are known a priori and thus agent interactions can be statically predefined [Sycara et al, 2003]. In open systems, neither the agents' behavior nor their construction is fixed and under the control of a single authority. An OperA model can be thought of as a kind of abstract protocol that governs how member agents should act. However, this is not sufficient to ensure the correct functioning of the system. It must also be ensured that member agents indeed comply with the protocol, that is, behave according to the norms and act towards the results described in the OperA model.

The most challenging aspect of the design and analysis of OperA models is to ensure and verify that global objectives of the society are met and no undesirable interactions between agents occur. This is especially so due to the openness criterion over which OperA is built. Because of the open nature of OperA models, compliance can only be expressed in terms of observable behavior and not in terms of rational behavior of agents (as for example expressed in BDI architectures) [Venkatraman, Singh, 1999]. We identify three levels of verification:

1. Does the society design comply with its requirements?
2. Does a society instantiation (social contracts) comply with the society design and it is sufficient to guarantee society activity as specified in the society design? That is, are there enough social contracts to enable society activity?
3. Does the society activity comply with society design? That is, are interaction contracts compliant with scene descriptions?

In OperA all interaction is specified in contracts, which can be translated into formal expressions in LCR. Since OperA models are also expressible in LCR, verification of compliance is reduced to a logical formal exercise. Actual compliance, that is, answering the question '*Do the agents behave as expressed in the contracts?*', is verifiable in 'run time', through the monitoring and sanction mechanism expressed in society design.

Another advantage of using contracts to express interaction, is that interaction is not a consequence of rational behavior of the agents. Rather, rational behavior may result as a consequence of obeying the contract. This also allows non-rational agents to participate as long as their architecture is able to realize the activities expressed in the contracts [Venkatraman, Singh, 1999]. In the following, we describe verification rules for organization models. We do not pretend to provide a complete verification, but to indicate the kind of issues that must be considered when verifying OperA organizational models.

At the OM level, it is necessary to verify the society structure and its components. One of the main issues of such verification is to assert the completeness of the society description with respect to the designer's objectives. The fact that OperA has a formal semantics, permits to give models a precise semantics, and can be used to guide and support the designer while building and refining a conceptual model. On the down

side, writing formal specification languages usually requires strong skills, and formal specifications are often very ineffective for discussing with the stakeholders.

The first activities related with model verification take place at the early stages of the development methodology³⁶, and concern the assessment of the model against possible inadequacies, incompleteness or inconsistencies, the validation of the initial specification with the stakeholders in order to end up with an agreed set of requirements, and, the management of model refinement and evolution, in order to maintain its consistency, propagate changes, merge redundant information [Perini et al, 2003]. For example, if a scene script specifies a set of participating roles, those roles must be defined.

Once the Organizational Model for a society is specified, verification will determine whether the model provides enough elements to make the achievement of the society objectives possible. At this level, this is all what can be verified. The actual realization of society objectives can only be checked when actual agents are animating the society. In the following we specify and motivate some of the properties that can be verified for an Organizational Model. Note that this is a minimal set of properties, meant as example of the kind of properties that need to be verified. In order to develop an automated model checker, more research is due to reach a complete set of properties.

Formally, given a society S , the following properties must be verified³⁷:

1. $\forall s \in S_S \forall r \in \text{roles}(s): \exists R \in R_S: \text{id}(R) = r$
2. $\forall s \in S_S \forall \gamma \in \text{results}(s) \exists R \in R_S: \text{id}(R) \in \text{roles}(s) \wedge \gamma \in (\text{objectives}(R) \cup \text{sub-objectives}(R))$
3. $\forall R \in R_S \forall \rho \in \text{objectives}(R) \exists s \in S_S: \text{id}(R) \in \text{roles}(s) \wedge \rho \in \text{results}(s)$
4. $\forall G \in G_S \forall r \in \text{roles}(G) \forall \phi \in \text{norms}(G) \forall \psi \in \text{norms}(R): \phi \wedge \psi$ is consistent, where $R \in R_S: \text{id}(R) = r$.

Property 1 above describes the fact, for each interaction scene in the society, it must be verified that the participating roles have been specified in the model. For example, for the scene ‘Review Process’ both roles PC-Chair and PC-member must be specified. This property is an example of properties on the existence of components. The same type of property must be defined, for instance, to verify the existence of all scenes referred to by a scene connector or transition.

Properties 2 and 3 refer to the verification of the feasibility of society objectives. That is, for all role objectives, there must be a scene where that objective is part of the scene results, and for all scenes, every result of the scene must be part of the objectives or sub-objectives of at least one of the participating roles. For example, in

³⁶ Cf. chapter 6 about the design methodology

³⁷ A note on notation: we use lowercase r, r_1, r_2, \dots to denote role identifiers (i.e. elements of Roles_D) and uppercase R, R_1, R_2, \dots to denote roles (i.e. elements of R_S)

the conference society, a result of the scene ‘Review Process’ is (cf. Figure 5-3 in page 129):

DONE $\forall P \in \text{Papers, paper-reviewed}(P, \text{PC1, review1}) \text{ AND } \text{paper-reviewed}(P, \text{PC2, review2})$

Participants in this scene are the roles ‘PC-Chair’ and ‘PC-member’. According to property 2, $\text{paper-reviewed}(P, \text{PC, R})$ must be an objective of at least one of these roles. This is indeed the case, as $\text{paper-reviewed}(P, \text{PC, R})$ is the objective of the role PC-member (cf. the definition of PC-member in figure 3-7, in chapter 3).

The last property indicates that norms defined for a group must be consistent with the norms of each role in that group. For example, in the conference society, a norm was defined for the group Organizers, formed by roles PC-Chair, Local-Organizer, etc., specifying that organizers are forbidden to submit papers to the conference (cf. figure 3-9 in chapter 3). In this situation, of course it should not be the case that, for example, the role of PC-Chair has a role norm specifying that a program chair is obliged to submit a paper to the conference.

Furthermore, model verification must also support the management of model evolution, catching the creation of inconsistencies due to a modification of a (previously consistent) model. For example consider that in an evolution of an OperA model a role is removed. In this case, verification should be able to determine critical problems of other roles and/or scenes. This corresponds to general question of whether a change in the model makes it impossible to achieve a goal that was previously possible to achieve, or if it introduces new critical goals or dependencies.

The complete specification of verification rules for OperA will be part of future work, when we will implement an automated tool for the specification of OperA models.

5.7 Syntax of OperA

We complete this chapter with a description of the basic elements of the OperA syntax. The operationalization of a specification language for OperA, based on the syntax below, is however part of our future research.

```

<!-- OM definition -->
<OM> ::= OM(<Domain-Language>, <SS>, <IS>)
<!-- Language definition -->
<Domain-Language> ::= (<pred>+, <func>+, <id>+, <var>+, <Term>+)
<Form> ::= <PosForm> |  $\neg$ <Form>
<PosForm> ::= <pred>(<Object> [, <Object>]*) |
               <PosForm> AND <PosForm> |
               FORALL <var> <PosForm>
<Object> ::= <func> |
              <id> |

```

```

        <func>(<Object> [, <Object>]*)
<!-- Social structure definition -->
<SS> ::= Social-structure(
    Roles: <Role>+,
    Groups: <Group>*,
    Role-dependencies: <Role-dependency>*)
<!-- Role definition -->
<Role> ::= Role( <Role-Id>,
    Objectives: <Role-objective>+,
    Sub-objectives: <Role-sub-objective-sets>*,
    Rights: <Role-right>*,
    Norms: <Role-norm>*,
    Type: <Role-type> )
<Role-id> ::= <id>
<Role-objective> ::= <PosForm> | <id> = <PosForm>
<Role-sub-objective-sets> ::= {<PosForm>} |
     $\prod(\langle id \rangle) = \{\langle PosForm \rangle\}$ 
<Role-right> ::= <PosForm>
<Role-norm> ::= <Norm>
<Role-type> ::= external | institutional
<!-- Group definition -->
<Group> ::= Group(<Group-Id>,
    Roles: <Role-Id>+,
    Norms: <Norm>+ )
<Group-Id> ::= <Role-Id>
<!-- Role dependency definition -->
<Role-dependency> ::= Dependency(<Role-Id>, <Role-Id>, <Dep-type>)
<Dep-type> ::= hierarchy | market | network
<!-- Norm definition -->
<Norm> ::= OBLIGED(<id>, <Norm-Form>) |
    PERMITTED(<id>, <Norm-Form>) |
    FORBIDDEN(<id>, <Norm-Form>) |
    IF <Achieved-Form> THEN <Norm>
<Norm-Form> ::= <Form> | <Form> BEFORE <Form>
<Achieved-Form> ::= DONE(<id>, <Form>)
<!-- Interaction structure definition -->
<IS> ::= Interaction-structure(
    Scenes: <Scene>+,
    Start-scene: <Scene-Id>,
    End-scene: <Scene-Id>,

```

```

    Connectors: <Connector>*,
    Transitions: <Transition>*,
    Role-evolutions: <Role-evolution>* )
<!-- Scene definition -->
<Scene> ::= Scene( <Scene-Id>,
    Roles: <Scene-role>+,
    Results: <Scene-result>+,
    Patterns: <Scene-pattern>*,
    Norms: <Scene-norm>*)
<Scene-Id> ::= <id>
<Scene-Role> ::= [<Participant-id>:] <Role-Id> [( <Arity>)]
<Scene-result> ::= <Achieved-Form> | <id> = <Achieved-Form>
<Scene-pattern> ::= {<Landmark>} | Pattern(<id>) = {<Landmark>}

<Landmark> ::= <Form> |
    <Achieved-Form> |
    <Landmark> BEFORE <Landmark>
<Scene-norm> ::= <Norm>
<Participant-id> ::= <id>
<Arity> ::= <integer>
<!-- Connector definition -->
<Connector> ::= Connector(
    Source: <Scene-Id>,
    Target: <Scene-Id>,
    Source-constraints: <Landmark>,
    Target-constraints: <Landmark> )
<!-- Transition definition -->
<Transition> ::= Transition(<Connector>+, <Transition-type>)
<Transition-type> ::= all-targets |
    some-targets |
    one-target |
    new-target |
    all-sources |
    some-sources |
    one-source
<!-- Role evolution definition -->
<Role-evolution> ::= Role-evolution(
    <Scene-role>,
    <Scene-role>,
    <Evolution-type>,
    <Landmark> )
<Scene-role> ::= <Scene-Id>.<Role-Id>

```

```

<Evolution-type> ::= necessary | sufficient | conflict
<!-- SM definition -->
<SM> ::= SM(<OM>, <Agent>+, <Social-contract>+)
<Agent> ::= <id>
<Social-contract> ::= Social-contract(
    Agent: <Agent-Id>,
    Role: <Role-Id>
    Clauses: <Contract-clause>*)
<Contract-clause> ::= <Norm>
<!-- IM definition -->
<IM> ::= IM(<SM>, <Interaction-contract>+)

<Interaction-contract> ::= Interaction-contract(
    Parties: <Rea>+,
    Scene: <Scene-Id>
    Clauses: <Contract-clause>*,
    Protocol: <Protocol>)

<!-- partial definition of protocol -->
<Protocol> ::= includes the definition of <Illocution>
<Illocution> ::= <Comm-act>(<Sender>, <Receiver>, <Form>)
<Comm-act> ::= request | commit | inform | declare
<Sender> ::= <id>
<Receiver> ::= <id>

```

5.8 Conclusions

The extension of the temporal deontic logic LCR with illocution and epistemic elements gives a formal framework and integrated semantics that provides a precise interpretation of OperA concepts, at all three levels of society specification (organizational, social and interaction). The formalism provides a rather realistic representation of a domain, in the sense that it treats temporal aspects and furthermore is able to represent deadlines and their influence in the behavior of the model.

The formal language presented in this chapter is meant to specify the elements of an OperA society as introduced in chapter 3 and as such does not include the specification of the *players* (i.e. the agents themselves) in a society. An extension to the formal language to support the specification of the internal aspects of agents would enable the complete specification of an animated society. As such the formalism presented in this chapter does not result in a system but in a conceptual model. Therefore, we are not able to verify typical system properties such as liveness or create system traces. The formal model described above enables however to check

model properties, such as the verification of whether a role objective can be achieved, given the scenes specified, or whether the roles are sufficient to realize society objectives, etc.

The present work opens several areas for further research. An obvious direction is the extension of the formal language to include the specification of the agents themselves, or integration of the formalism with existing formal languages for agents, such that an operational system can be generated and verified. Another interesting direction is the development of an automated society generator, that would enable the verification of OperA specifications and indicate eventual flaws or inconsistencies. Both aspects are not part of this dissertation, and moreover, will, on their own, probably be enough for another Ph.D. dissertation.

Chapter 6

Designing Agent Societies

*See first that the design is wise and just;
that ascertained, pursue it resolutely;
do not for one repulse forego the purpose
that you resolved to effect.*
– William Shakespeare (1564 – 1616)

Having presented the OperA model for agent societies and described its formal semantics, we will now discuss the practical implementation of agent societies. In this chapter, we present a methodology for the design of agent societies, based on the OperA framework. The method considers the influence of social organizational aspects on the functionality and objectives of the agent society and specifies the development steps for the design and development of an agent-based system for a particular domain. This chapter is a modified and extended version of [Dignum et al., 2001] and [Dignum, Weigand, 2002].

The chapter is organized as follows: section 6.2 describes the state of the art in agent-oriented software engineering. Section 6.3 introduces different coordination frameworks for agent societies. In section 6.4 our proposal for an engineering methodology for agent societies is presented, using an example of a knowledge exchange system at Achmea to illustrate its application in practice. Because societies and their environment are not static, but evolve and change continuously, the problem of modifying society models to dynamically adapt to changing requirements is discussed in section 6.5. Finally, we present our conclusions, discuss the OperA methodology and give pointers to future work in section 6.6.

6.1 Introduction

Interaction has been recognized as one of the most important characteristics of complex systems. The applicability of multi-agent approaches to the development of real-world applications will for a large part depend on the way multi-agent systems are able to model and handle interaction [Wooldridge, Ciancarini, 2001]. Current

Agent-Oriented Software Engineering (AOSE) approaches take interaction and coordination as the central focus of multi-agent systems design, and of the organizational characteristics of the domain. Development methodologies that can handle interaction must focus not just on the internal organization of each of the intervening agents but also on the social aspects of the domain [Omicini, 2001].

In our opinion, the concept of agent societies presented in chapter 3, provides an adequate way to look at distributed systems and organizational modeling. However, building agent societies is difficult. Besides all the problems of traditional distributed and concurrent systems, agent societies have the additional difficulties that arise from flexible requirements, evolutionary specification and sophisticated interactions [Wood, DeLoach, 2001]. Agents interact with others in the agent society as a means to accomplish their roles. Because of the proactive and autonomous behavior of agents it seems natural to design agent societies mimicking the behavior and structure of human organizations [Zambonelli et al., 2001b]. This perspective makes the design of the system less complex since it reduces the conceptual distance between the system and the real-world application it has to model. The above considerations can be summarized in the requirements for agent societies discussed in chapter 3:

- **internal autonomy requirement:** interaction and structure of the society must be represented independently from the internal design of the agents
- **collaboration autonomy requirement:** activity and interaction in the society must be specified without completely fixing in advance the interaction structures.

The focus of the current chapter is on how to design systems that fulfil these requirements. It is important to understand that answers to these requirements are sociological in nature and will not be found in *single agent* approaches [Frederiksson, Gustavsson, 2001]. That is, the engineering of agent societies must be grounded on a notion of behavior as a result of interaction between multiple entities, as opposed to the internal behavior of a single agent.

Agent societies that satisfy the requirements mentioned above must assume that social structures are developed independently from society participants, and furthermore that there is not one single entity responsible for the construction and control of the overall system. According to [Frederiksson, Gustavsson, 2001], this identifies an important difference between construction of society models and observation of society behavior. Finally, because coordination is a cornerstone of agent societies, in order to be able to observe and construct the behavior of a domain, engineering methodologies for agent societies furthermore need modeling primitives and principles of behavior that are based on the interaction and coordination characteristics of that domain.

6.2 Agent-Oriented SE Methodologies

Agent-oriented software engineering (AOSE) is a recent field in the area of software engineering that is concerned with the development of methodologies for the structured design of multi-agent systems. Such methodologies should provide models and methods adequate to represent and support all types of activities throughout all phases of the software lifecycle. Because of similarities between agents and objects, it

has often been claimed that existing object-oriented (OO) methodologies can be used for the development of agent-based systems. However, it has also been noted that agents possess specific characteristics that are not covered by traditional OO methodologies [Omicini, 2000], [Jennings et al, 1998]. Generic SE methodologies are therefore not tailored to deal with design aspects specific to agent systems, such as capturing the flexible and autonomous behavior of agents and the complexity of agent interactions and social organization of the system [Wooldridge et al., 2000].

One fundamental difference between agents and objects is *autonomy*, which refers to the principle that agents have control over their own actions and internal state. That is, agents can decide whether or not to perform a requested action. Objects have no control over their own methods, that is, once a publicly accessible method is invoked the corresponding actions are performed [Wooldridge, 1997]. Another characteristic of multi-agent systems that calls for specific methodological approaches is *openness*. Since components and relationships in an open system can change at any time, designers cannot be certain of the systems behavior at the time of design. Frederiksson defends that a methodological cycle for the engineering of agent societies must comprise principles of both system observation and construction [Frederiksson, Gustavsson, 2001]. Also, intelligence and interaction are deeply and inevitably coupled, and multi-agent systems reflect this insight. Multi-agent systems can provide insights and understanding about poorly understood interactions between natural, intelligent beings as they organize themselves into groups, societies and economies in order to achieve improvement.

Agent societies that model organizational domains must be able to describe specific characteristics of organizational settings. Business processes are highly dynamic and unpredictable which makes it difficult to give a complete a priori specification of all the activities that need to be performed, what are their knowledge needs, and how they should be ordered. Within organizations, there is a decentralized ownership of the tasks, information and resources involved in the business process. Different groups within organizations are relatively autonomous, in the sense that they control how their resources are created, managed or consumed, and by whom, at what cost, and in what time frame [Jennings et al, 1996]. Furthermore, often multiple, physically distributed organizations (or parts hereof) are involved in the business process, each of which attempts to maximize its own profit within the overall activity. That is, there is a high degree of natural concurrency (many interrelated tasks and actors are running at any given point of the business process) which makes it important to monitor and manage the overall business process (e.g. total time, total budget, etc.). That is, organizations are often extremely complex entities. Consequently, models for organizations, and especially those that treat interaction as a main characteristic of organizations, tend to be difficult to design and maintain.

Moreover, a methodology for designing multi-agent systems must be both specific enough to allow engineers to design the system, and generic enough to allow the acceptance and implementation of multi-agent systems within an organization, allowing for the involvement of users, managers and project teams. From an organizational point of view, the behavior of individual agents in a society can only be understood and described in relation to the social structure. Therefore, the engineering of agent societies needs to consider both the interacting and

communicating abilities of agents as well as the environment in which agent societies are situated. Furthermore, in open societies the ‘control’ over the design of participating agents lies outside the scope and design of the society itself. That is, the society cannot rely on the embedding of organizational and normative elements in the intentions, desires and beliefs of participating agents. These considerations lead to the following requirements for engineering methodologies for agent societies [Dignum, Dignum, 2001]:

- Include formalisms for the description, construction and control of the organizational and normative elements of a society (roles, norms and goals).
- Provide mechanisms to describe the environment of the society and the interactions between agents and the society, and to formalize the expected outcome of roles in order to verify the overall animation of the society.
- The organizational and normative elements of a society must be explicitly specified because an open society cannot rely on its embedding in the agent’s internal structure.
- Methods and tools are needed to verify whether the design of an agent society satisfies its design requirements and objectives.
- Provide building directives concerning the communication capability and ability to conform to the expected role behavior of participating agents.

In our opinion, none of the currently existing agent-oriented engineering methodologies, such as Gaia [Wooldridge et al., 2000] and SODA [Omicini, 2001] fulfil yet all of the requirements listed above. Most existing methodologies concentrate on just one part of the total picture or are too formal to be applicable in practice. Furthermore, most approaches start from the moment that the decision to use the agent paradigm has been made, and do not guide this choice. A promising development in this area is the Tropos project, which aspires to span the overall software development process [Giunchiglia et al., 2002b].

6.2.1 Related Work

Currently, research on AOSE is a topic of great interest, cf. for example [Wooldridge, 1997] and [Jennings, 2000], and the proceedings of the workshop series on Agent-Oriented Software Engineering [Ciancarini, Wooldridge, 2001], [Wooldridge et al., 2002], [Giunchiglia et al., 2002a]. A good overview and analysis of existing methodologies, applicable to Agent-Oriented Information Systems can be found in [Azary, Woo, 2000]. However, there is as yet no well-established and all-encompassing agent-oriented methodology that covers the whole development process from early requirements acquisition to implementation. In the following we will describe some of the most commonly used AOSE methodologies and discuss their contributions and shortcomings.

Gaia [Wooldridge et al., 2000] is the first agent-oriented software engineering methodology that explicitly takes into account social concepts. Gaia aims at providing a coherent conceptual framework for the analysis and design of multi-agent systems. Roles are seen as a kind of agent class, and treated in Gaia as first-class entities, which are associated with responsibilities, permissions, activities and protocols.

Furthermore, normative aspects are reduced to static permissions, a sort of constraints or rules and behavior is fixed in protocols. In our opinion, Gaia is not well suited to model open domains, and cannot easily deal with self-interested agents, as it does not distinguish between organizational and individual aspects, and does not provide capabilities for agent interpretation of society objectives, norms or plans.

In **SODA** [Omicini, 2001], as in our own OperA methodology, these problems are overcome by exploiting suitable coordination models as the basis for the engineering of societies, enabling open societies to be designed around suitably-designed coordination media, and social rules to be designed and enforced in terms of coordination rules. However, even though SODA distinguishes between agent and collective spaces, it sees roles as the representation of the observable behavior of agents, and therefore cannot represent the difference between the organizational perspective on the activity and aims of individuals (represented by the concept of role in OperA) from the agent perspective on its own activity and aims (represented by the concept of agent in OperA and linked to the role by a social contract). Furthermore, neither Gaia nor SODA is based on a MAS model with a clear and formal semantics.

An example of an AOSE tool and methodology, that was adapted from software and knowledge engineering concepts and tools is **DESIRE** [Brazier et al., 1997]. DESIRE allows the system designer to explicitly and precisely specify both the intra-agent functionality (i.e., the expertise required to perform the domain tasks) and the inter-agent functionality (i.e., the expertise required to perform and guide coordination, cooperation and other forms of social interaction in terms of the knowledge requirements and the reasoning capabilities). DESIRE views both individual agents and the overall system as a compositional architecture, functionality being designed as a series of interacting, task-based, hierarchically structured components. The tool also provides a formal generic agent model that can be reused as a template of pattern for a variety of agent types and application domains [Brazier et al., 2000].

In complex domains with multiple participants there is a need to integrate and represent multiple views on the domain. Recent proposals recognize this need and present layered approaches which also contribute to the refinement and clarity of the models. One such proposal is the **MASSIVE** framework (Multi Agent Systems Iterative View Engineering) [Lind, 2001], that presents a requirements-driven, view-oriented and iterative method for the development of multi-agent systems. The core of the method is a system of views that form the conceptual basis for a wide range of product models that are developed and refined throughout various software projects. A view represents a set of conceptually linked features of the target system, i.e. a view is a projection of the complete model onto a particular subject. The following views are identified:

- **Environment view:** analysis of the target system from the perspective of developers as well as from the systems perspective.
- **Task view:** the functional aspects of the target system are analyzed and a task hierarchy is generated that is then used to determine the basic problem solving capabilities of the entities in the final system.

- **Role view:** determines the functional aggregation of the basic problem solving capabilities according to the physical constraints of the target system. A role is an abstraction that links the domain dependent part of the application to the agent technology that solves the problem under consideration.
- **Interaction view:** interaction within the target system is seen as a generalized form of conflict resolution that is not limited to a particular form such as communication.
- **Society view:** classification of the society that either pre-exists within the organizational context of the system, or that is desirable from the point-of-view of the system developer. According to this classification and to well defined quality measures for the performance of the target society that depend on application of specific aspects, a society model is developed that is consistent with the roles within the society and that achieves the defined goals.
- **Architecture view:** provides a projection of the target system onto the fundamental structural attributes with respect to the system design. The system architecture is described according to various aspects and includes things such as agent management or database integration.
- **System view:** deals with system aspects that affect several of the other views or even the system as a whole. The System view, for example, handles the user interface that controls the interaction between the system and the user(s) whose task specific aspects are usually the input specification and the output presentation.

Finally, another recent layered proposal, developed almost in parallel to our work on OperA, is the **Vowels** coordination model, that integrates Agent, Environment, Interaction and Organization views [Silva, Demazeau, 2002]. The aim of this approach is to propose a cognitive multi-agent coordination model, that relates multi-agent plans and social aspects by means of social dependence notions. The model starts from the realization that until recently approaches to multi-agent design take one of the following views:

- **Agent:** coordination is based on the internal architecture of individual agents, examples are BDI architectures [Rao, Georgeff, 1995], or 3APL [Hindriks et al, 1999].
- **Environment:** coordination is primarily a social phenomenon characterized in terms of the relations between an entire system and its environment. An example is the Artificial Life paradigm [Ossowski, 1998].
- **Interaction:** coordination is based on the structure of internal interactions between entities. An example is the concept of Joint Intention [Cohen, Levesque, 1990].
- **Organization:** coordination occurs through social structures, dynamic coalition formation and mechanisms of social dependence. Examples are [Castelfranchi, 2000] and [Dunin-Keplicz, Verbrugge, 2003]. The OM in OperA is also an example of this perspective.

The advantage of the Vowels approach over other methodologies is that it attempts to integrate these different views into one framework. In particular they describe the Agent-Organization, the Interaction and the Environment coordination levels. These

levels are quite similar to the Social Model and Interaction Model of the Opera framework. However, and in opposition to OperA, the Vowels approach does not provide any formalization of their concepts and relationships.

6.2.2 Determination of Agent Appropriateness

The success and acceptance of agent technology as an industrial software standard will depend to a large extent on the development of robust agent tools and models such as FIPA or the AUML effort [Bauer et al., 2001] that guide and support the process of development and implementation of MAS in a structured and formal way. Furthermore, in order to make agent technology widely accepted and used in real-world applications, it is necessary to clearly specify the type of problems suitable for an agent approach and the benefits of agents above other technologies. It is commonly accepted that the agent paradigm is appropriate for complex, distributed problems and systems. Such systems maintain an ongoing interaction with some environment and are inherently quite difficult to design and correctly implement. Moreover, from an object-oriented software engineering view, agents can be seen as ‘active objects’, that is, as goal-directed objects with communicative and deliberative capabilities. In this sense, agents can be applied in areas where the object paradigm has limitations, especially where communication is involved [Jennings, Wooldridge, 1998].

These considerations show that an important component of any methodology for the development of agent systems is the evaluation and motivation of the choice of an agent-based approach to the organization involved in the deployment of the system. This need has been recognized both in the scientific and in the industrial communities. This is an open area for research, which will result in the definition of methods, libraries and heuristics to support and inform the application of the agent paradigm to arbitrary problems. Both empirical and methodological guidance are necessary for the identification of application areas for which agent-oriented approaches are the best. [O’Malley, DeLoach, 2001] describes a method to determine the best methodology for a particular problem, and compares both agent-oriented and object-oriented methodologies. The EURESCOM³⁸ has produced the following guidelines to help a developer decide whether or not an agent-oriented approach is appropriate given the characteristics of a domain or situation [MESSAGE, 2000]:

- Complex/diverse types of communication are required
- It is not possible and/or practical to specify the behavior of the system in a case-by-case basis
- Negotiation, cooperation and competition among different entities is involved
- Autonomous action is necessary
- Expansion or modification of the system or its environment is highly probable

³⁸ European Institute for Research and Strategic Studies in Telecommunications

These guidelines provide a good starting point to determine whether or not an agent-based approach is appropriate for a particular problem. The AgentLink consortium has also identified the following areas of potential application of multi-agent systems, which mainly divide multi-agent applications into the ones where the system itself can make decisions and those which only provide advice to human decision-makers [Luck et al., 2003]:

- **Multi-agent decision systems**, where the agents participating in the system must together make some joint decisions. Mechanisms for joint decision-making can be based on economic mechanisms, such as an auction, or alternative mechanisms, such as argumentation. Examples are systems for resource allocation, process control, information gathering and filtering, electronic commerce, business process management, etc.
- **Multi-agent simulation systems**, where the MAS is used as a model to simulate some real-world domain. Typical use is in domains involving many different components, interacting in diverse and complex ways and where the system-level properties are not readily inferred from the properties of the components. Examples of such domains include economics, human and animal societies, biological populations, road-traffic systems, games, etc.

In summary, the goal of multi-agent decision systems is the creation of a formal or non-formal model to support and/or be integrated in the organization described, whereas simulation systems have the goal of understanding the domains modeled. Furthermore, often in simulation systems, agents are used as an appropriate representation of real world components, while in decision systems, agents are used for their capabilities.

OperA models can be used both for decision and for simulation applications. In the next section we will introduce a methodology to develop agent societies according to the OperA model. Earlier versions of this methodology were introduced in [Dignum et al., 2001], [Dignum, Weigand, 2002].

6.3 Agent Society Frameworks

Multi-agent systems that are developed to model and support organizations, need coordination frameworks that mimic the coordination structures of the particular organization. These coordination types have been presented in chapter 2, section 2.5.1, and are for a large part determined by transaction costs. Depending on the type of goods or services exchanged and on the characteristics of the exchange context, organizations are better coordinated as a market, a hierarchy or a network. These organizational structures determine important autonomous activities that must be explicitly organized into autonomous entities and relationships in the conceptual model of the agent society [Dignum et al., 2001]. Different application contexts exhibit different needs with respect to coordination, and the choice of a coordination model will have great impact on the design of the agent society. Following this observation, we argue that the first step in the development of an agent society is to identify its underlying coordination model. In this section we will describe in more

detail social frameworks for agent societies that implement organizational coordination models.

So far, most agent-oriented design methodologies have not considered the influence of the type of social coordination on the functionality and objectives of an agent society. In many cases, social organization is left implicit in the design of the agent society. However, the coordination model determines important autonomous activities, which must be explicitly organized into autonomous entities in the conceptual model of the agent society. Based on ideas from organizational science research concerning the applicability of coordination models for organizations, we propose a model for agent societies that consider and reflect the implications of the coordination model of the real-life organization being modeled.

At the most basic level, social organization integrates facilitation and operation activities. The global objectives of a society are domain dependent, but all societies depend on a **facilitation layer** that provides the social backbone of the organization [Dellarocas, 2000]. Facilitation activities deal with the functioning of the society itself and are related to the underlying coordination model. On top of this facilitation layer, an **operational layer** is needed that implements the objectives of the society. Operational activities are directly related to the objectives and aims of the society.

The different coordination models – markets, hierarchies and networks - result in different structures for organizations, and as such, in different frameworks for agent societies that model those organizations. Therefore we argue that the first step in the development of agent societies is to identify the underlying coordination model. The social coordination model is used to specify the facilitation framework for an agent society. In the next stage, the framework is extended with the operational layer, that is, domain specific roles and interaction forms that characterize the problem. We can compare this process to the design of a generic enterprise model including roles such as accountants, secretaries and managers, as well as their job descriptions and relationships, and then extending it with a ‘recipe’ to build the functions necessary to achieve the objectives of the given enterprise. These are, for example, designers and carpenters (if the firm is going to manufacture chairs), and programmers and system analysts (if the enterprise is a software house). While the chosen coordination model determines the social part and is for a large part independent from the domain, domain roles are directly derived from the domain requirements.

In **markets**, agents are self-interested (determine and follow their own goals) and value their freedom of association and own judgement above security and trust issues. Openness is thus per definition a feature of markets. Facilitation is, in the most extreme case, limited to identification and matchmaking activities. Interaction in markets occurs through communication and negotiation.

Network organizations are built around general patterns of interaction or contracts. Relationships are dependent on clear communication patterns and social norms. Agents in a network society are still self-interested but are willing to trade some of their freedom to obtain secure relations and trust. Therefore, agents need to enter a social contract with the network society in which they commit themselves to act within and according to the norms and rules of the society. The society is responsible

to make its rules and norms known to potential members. Coordination is achieved by mutual interest, possibly using trusted third parties, and according to well-defined rules and sanctions.

Finally, in a **hierarchy**, interaction lines are well defined and the facilitation level assumes the function of global control of the society and coordination of interaction with the outside world. In a hierarchy, agents are cooperative, not motivated by self interest and all contribute to a common global goal. Coordination is achieved through command and control lines. Such agents are said to be benevolent, that is, assumed to always attempt to do what is requested from them [Mohamed, Huhns, 2000].

The coordination model determines interaction patterns and functionality of the facilitation layer of the agent society, that is, the interaction primitives and agent roles necessary to implement the facilitation layer are specific to each type of society (market, network or hierarchy). Moreover, coordination models provide a framework to express interaction between the activities of agents and the social behavior of the system [Ciancarini et al., 1999]. While Table 2-1 in chapter 2 provided a summary of the most relevant characteristics of organizations that determine its coordination model, Table 6-1 gives an overview of the characteristics of agent societies that model the different coordination types. In the following subsections we describe the generic facilitation and interaction frameworks for agent societies in more detail. These frameworks implement the specific functionality derived from the type of coordination holding in the domain. Later in this chapter, in section 6.4.1.1.1, we will provide the specification of the different facilitation roles related to the different coordination models.

Table 6-1: Coordination in agent societies

	MARKET	NETWORK	HIERARCHY
Type of society	Open	Trust	Closed
Agent 'values'	Self interest	Mutual interest/ Collaboration	Dependency
Exchange regulation	Price	Interdependence	Supervision
Facilitation roles	Matchmaking Banking	Gate-keeping Matchmaking Notary Monitoring	Control (root) Interface

6.3.1 Market framework

The main goal of **markets** is to facilitate exchange between agents. In a market, heterogeneous agents will strive to find partners or clients with whom to trade their services. Being open systems, market architectures assume the heterogeneity of their members, both in structure, goals and ways of acting. Markets are particularly suitable to situations in which resources are overlapping and agents need to compete for them, and are therefore a good choice to model product or service allocation problems. Being self-interested, agents will first try to solve their own local problem, and only then agents will potentially negotiate with other agents to exchange services or goods

in shortage or in excess. The decision to enter into or cancel a transaction is usually left to the agent itself.

The market metaphor has often been used to describe agent interaction, enhancing the adaptation, robustness, and flexibility of multi-agent systems. In a market model, agents representing (or providing) services and/or skills compete to perform tasks leading to the satisfaction of their own individual objectives as well as to a possible overall system's goal [Oliveira, 1999].

Facilitation activities of a market agent society are directed to help participating external agents find suitable partners through identification and matchmaking. **Matchmakers** keep track of agents in the system, their needs and possibilities and mediate in the matching of demand and supply of services. Reputation facilities are meant to build the confidence of customers as well as to offer guarantees to society members. Participants can consult a trusted third party, often a **bank**, to request information on the reputation of potential partners. Furthermore, the bank can provide financial facilities and currency specification. Finally, it is necessary to define ways to value the goods to be exchanged and determine profit and fairness of exchanges. The **market master** is responsible for the transactions and to enforce the regulation mechanism that holds in the market, which can be, for example, an auction mechanism or the Contract Net Protocol.

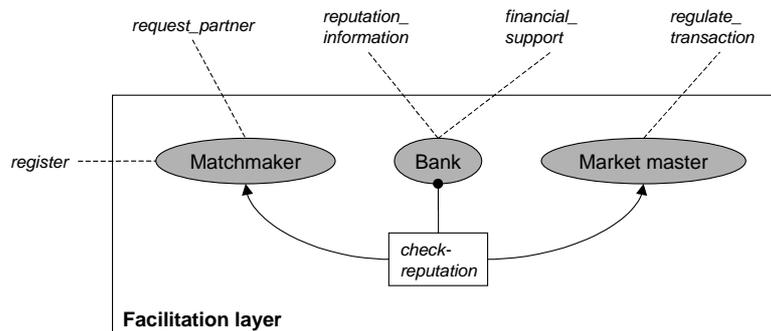


Figure 6-1: Facilitation basis for market societies

Figure 6-1 illustrates the facilitation layer of market societies. Roles are represented by ovals, and are linked to their objectives by dashed lines. These objectives are realized by interaction with operational roles specific to the application domain, which are outside the facilitation layer. Furthermore, interaction between facilitation roles may be necessary to support some role objectives. 'Check-reputation' is one of such interaction scenes, which are represented by a box linked to the roles that participate in the interaction scene. The initiator role is indicated by a bullet, and the responding roles are pointed by an arrow. In the case of the scene 'check-reputation' above, bank is thus the 'initiator', and 'matchmaker' and 'market master' are the responders.

6.3.2 Network framework

Networks are coalitions of self-interested agents that agree to collaborate for some time in order to achieve a mutual goal. Relationships between agents are dependent on clear communication patterns and social norms. The society is responsible to make its rules and norms known to its potential members. Agents in a network society are self-interested but still willing to trade some of their freedom to obtain secure relations and trust. Coordination is achieved by mutual interest, possibly using trusted third parties, and according to well-defined rules and sanctions. Network relationships are characterized by a tension between autonomy and interdependence, between loyalty to the team and individuality, between cooperation and competition. Often, networks have been considered a hybrid form of markets and hierarchies, but, as Powell has argued, this is 'historically inaccurate, overly static, and it detracts from our ability to explain many forms of collaboration' [Powell, 1990]. That is, the type of coordination in networks is neither price nor supervision but, mutual interest and interdependence (cf. Table 6-1)

Networks provide an explicit shared context, describing rules and social norms for interaction and collaboration. As in a market, the aim of agents when entering a network society is to trade their knowledge, goods or services. However, in networks, agents are motivated by a sense of community, and not solely guided by self-interest. An agent society based on the network model must be able to describe its rules of interaction, regulations, facilities and legal guarantees to applying members. This type of coalition has been also studied in the area of game theory and Distributed Artificial Intelligence (DAI) [Tsvetovat et al., 2001].

Dellarocas introduces the concept of Contractual Agent Societies (CAS) as a model for developing agent societies [Dellarocas, 2000]. CAS has been inspired by work in the areas of organizational theory, economy and interaction sociology, which model organizations and social systems after contracts. Social contracts govern the interaction of a member with the society. That is, agents enter a social contract with the network society in which they commit themselves to act within and according to the norms and rules of the society and of the role they will assume. The society is responsible to enforce the contracts formed by its members and punish potential violators (for example, through loss of reputation or eventually banishment). New agents are admitted through a process or socialization during which the agent negotiates with the society the terms of its membership. As a result, the terms of the social contracts of existing members may need to be renegotiated as well. Relationships between agents can also be described by *contracts*.

Facilitation level agents monitor and help form contracts, take care of introducing (teaching) new agents to the rules of the society, keep track of the reputation of agents. Furthermore, they keep and enforce the norms of the agent community and ensure interaction. Besides matchmakers as in market frameworks, other types of facilitation level agents in networks are gatekeepers, notaries and monitoring agents. Gatekeepers are responsible for accepting and introducing new agents to the market. Agents entering the system must be informed about the possibilities and capabilities of the society. **Gatekeepers** negotiate the terms of a social contract between the applicant and the members of the society. **Notaries** keep track of collaboration

contracts between agents. **Monitoring agents** are trusted third parties. The society will provide monitoring agents to interested parties. When a contract appoints a (set of) monitoring agents, this is the equivalent to the setting up of a (super) contract between the contracting agents and the environment (here personified by the monitoring agents). This super-contract (which can also be described using the contract language) specifies that the monitoring agents are allowed to check the contracting agents actions (e.g., look at agent states) and that the contracting agents must submit to the sanctions imposed.

Figure 6-2 illustrates the facilitation layer of a network society. The objectives of these roles are related to roles in the operational layer. Two interaction scenes between facilitation roles are identified to support the facilitation activities of the roles. Once a contract between agents enacting operational roles in the society is registered with the notary, it can appoint a monitor to control and support the execution of that contract. Furthermore, the gatekeeper can ask other facilitation roles about the reputation of an applying agent.

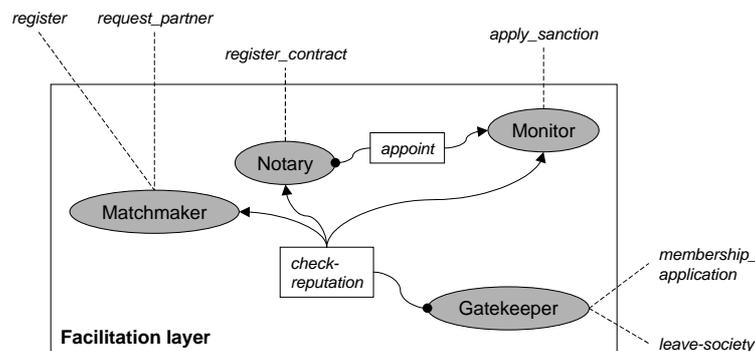


Figure 6-2: Facilitation basis for network societies

6.3.3 Hierarchy framework

In a hierarchy, the flow of resources or information is coordinated through adjacent steps by controlling and directing it at a higher level in the managerial hierarchy. Managerial decisions, and not negotiation and communication as in markets, determine the interaction possibilities and the design of hierarchical societies. Demand parties do not select a supplier from a group of potential suppliers: they simply work with a predetermined one. In hierarchical systems, each agent reigns over an arbitrarily and usually statically defined sub-hierarchy, in many cases an administrative domain of some kind. For instance, a university could be managed as follows: an agent is in charge of each lab, whereas other agents each oversee a department and a single one rules the university. These domains do not reflect the easy routing parts of the network and are not evolutionary.

Two types of integration can be distinguished in hierarchical organizations. **Vertical integration** occurs usually when the hierarchy is within a single firm and

are usually not free to enter or leave the system and communication with the outside must be regulated at the facilitation level.

Figure 6-3 illustrates the facilitation component of a hierarchical society. The basic facilitation of a hierarchical society is in principle quite simple, as all coordination and control are dependent on the root agent. The objectives of the society are assigned to operational agents by the root agent which also controls their activity. Often interface activities, even though regulated by the root, are delegated to a specific interface agent.

6.4 The OperA Methodology

In this section we present a methodology for the modeling and construction of agent societies based on an organizational, collectivist view that specifies coordination through pre-established roles, responsibilities and norms. This work applies ideas from coordination theory research in organizational sciences to the design of agent societies. The methodology considers and reflects the implications of the coordination model of the real-life organization being modeled (that is, using the coordination frameworks described above) and results in a complete design of an agent society according to the OperA model.

Our aim is to develop a practical methodology applicable to all the steps of development of a multi-agent system. Following the ideas of [Frederiksson, Gustavsson, 2001], our methodology provides the means to build and adapt society models based on the observation of the behavior of the system. The methodology takes the organizational perspective as starting point and specifies the development steps for the design and development of an agent-based system for a particular domain. Once these steps have been identified, existing methodologies can be used for the development and modeling of each step. We believe that such a generic framework, based on the organizational view, will contribute to the acceptance of multi-agent technology by organizations. Furthermore, the methodology proposed gives an answer to the development challenges posed by Katia Sycara [Sycara, 1998]:

1. How to engineer practical multi-agent systems
2. How to decompose problems and allocate tasks to individual agents
3. How to coordinate agent control and communication
4. How to make multiple agents to act in a coherent manner
5. How to make each agent reason about the other agents and the state of coordination
6. How to reconcile conflicting goals between coordinating agents

Based on the coordination model that either holds in the observed system or that is desired for the system to be developed, we define a social framework for agent communities that 'implements' the generic interaction, cooperation and communication mechanisms (2 and 3) that occur in the problem domain. The proposed methodology (1) allows to tailor this generic coordination model to the specific application and to determine the specific agent roles and interactions (5). In the following steps, the level of design detail will be successively increased to include the internal organization and reasoning capabilities of the agents (4). Finally, in our

opinion, the reconciliation of conflicting goals (6) cannot be completely fixed in the system structure as it is dependent on the interaction agreements between the agents. Depending on the Organizational Model, different reconciliation processes can be specified and possibly negotiated between role enacting agents. Conflict resolution must therefore be dealt with in the interaction contracts specified in the Interaction Model of OperA.

An important lesson-learned from the past is that the development of multi-agent systems, as any other software systems, cannot be seen isolated from the (organizational) context where it is inserted. System goals and structure must on the one hand, match organizational strategy and processes, and on the other hand, meet user expectations and requirements. An often heard complaint of managers in organizations is that lots of money and effort is spent on state-of-the-art ICT systems that are subsequently not or hardly used. In our opinion, such mistakes are due to the fact that system development mostly concentrates on the technical aspects of the system and organizational, cultural and users aspects are largely ignored or assumed accomplished. We think that system development must start with the analyses and facilitation of the social environment where it will be inserted. In the specific case of KM environment, which has formed an important background for our research, the realization that communities of users and stakeholders should be nurtured and their influence on the system taken into account, is of crucial importance to the success and acceptance of the system [Ali et al., 2002], [Carley, 1994]. Furthermore, systems must fit with the specific characteristics of the communities that are going to use them and will only succeed if community members are convinced of their benefits for themselves and for the organization. In order to support this social process, we have developed a method to facilitate the creation and management of communities [Dignum, van Eeden, 2003]. This method, the SES Model (Seduce, Engage, Support), is described in an appendix to this dissertation.

A preliminary step for the OperA methodology is the assessment of the applicability of the agent paradigm to the problem on hand. In the following, we will describe the methodology that, as in the OperA model, is structured in three steps: the design of the Organizational Model to implement the desired organizational structure of an agent society, the description of an agent population in the Social Model that will enact the roles described in the structure, and the specification of agent interactions in the Interaction Model, to achieve the desired society global objectives. Agents are integrated into the society design in the Social Model, by means of social contracts and agent interactions are fixed in the Interaction Model using interaction contracts. The relation between the different steps of the Opera methodology is illustrated in Figure 6-4.

As already discussed in chapter 3, the OperA model for agent societies does not include the design of the agents. Agents can be modeled using any available methodology or framework (e.g. Tropos, Gaia, 3APL, Jade, etc.). We assume however that agents can understand the society ontology and communicative acts and are thus able to communicate with the society (representatives). Future research can lead to the identification of methodological concepts that guide the development of agents dedicated to the enactment of given society roles. That is, one can conceive that the methodology can be extended to support the development of agents that will

participate in the society. Such agent development methodology will support the specification of the internal structure of agents in terms of the society requirements for the role, that is, communication, action, interface and reasoning behavior.

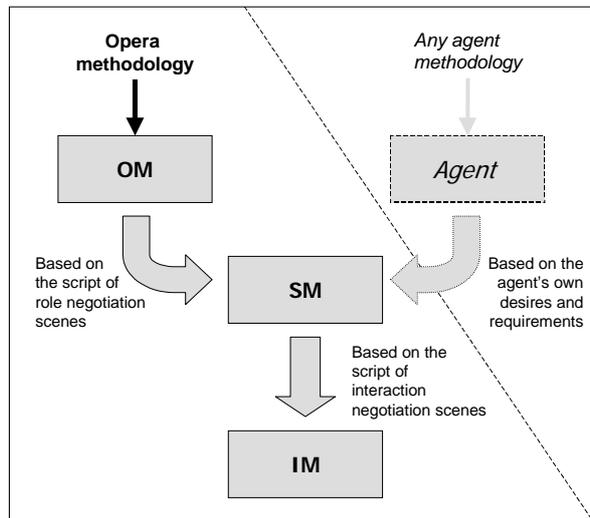


Figure 6-4: The OperA methodology steps

In the following we describe each step in detail. Section 6.4.1 describes how to design the Organizational Model for an agent society. Starting from the analysis of the environment, the methodology determines the stakeholders, requirements and use cases for the society, as well as the specification of its communicative and normative structures. Section 6.4.2 describes how to specify populations for a society starting from existing agents. As already referred in chapter 3, in OperA special interaction scenes can be specified meant to describe the way role enactment can be negotiated according to the requirements of the society designer. Such scenes are the basis for the negotiation of social contracts for agent populations. Finally, section 6.4.3 describes how to generate interaction contracts between role enacting agents. As for social contracts, OperA uses interaction negotiation scenes to describe the possibilities for interaction negotiation.

6.4.1 Organizational Model design

The first step of the OperA methodology results in the specification of the Organizational Model (OM) for an agent society. The OM design methodology consists of three levels, which provide a growing level of refinement of the resulting system into richer and more precise forms. Coordination requirements specify the coordination structure (market, hierarchy or network) of the society. Functional requirements determine the behavior of the society and its relationship with the environment. These requirements are the basis for a basic society model, of which behavior and animation can be verified and compliance to the domain requirements can be checked. The OM methodological process is described in Figure 6-5 and will

be described in more detail in the following sub-sections. The basic activities and results of the three methodological levels are as follows:

- **Coordination Level:** based on the coordination analysis of the domain, the structure of the society is determined using as basis the different coordination models (market, hierarchy and network) that have been detailed in section 6.3.
- **Environment Level:** the society model determined in the previous step is further refined with the specification of its social structure in terms of roles, global requirements and domain ontology. These are identified through the analysis of the relation between the system and its environment. Basis for this level are existing methods for ontology development and use case construction, as well as a library of common roles.
- **Behavior Level:** the organizational model of an agent society is completed with the specification of its interaction structure which results from the analysis of the interaction patterns and processes of the domain. This process is supported by a library of interaction patterns.

These steps result in a complete Organizational Model for the agent society. The model has a formal semantics based in LCR logic as described in chapter 5.

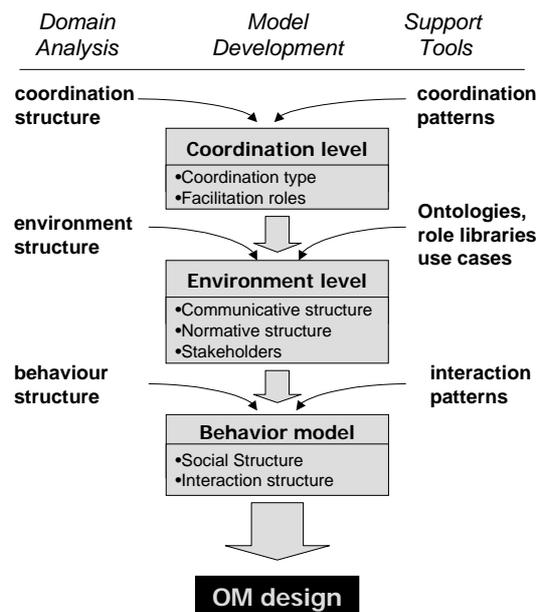


Figure 6-5: Design of the OM for an agent society

In the remainder of this chapter we will use the Knowledge Market system as an application example for the OperA methodology. Knowledge Market is a system that supports knowledge exchange between non-life insurance experts at Achmea. The system will be extensively discussed in chapter 7, including the rationale for an agent society based approach. For the moment we will only use some aspects that are

relevant to illustrate the methodology. The Knowledge Market is a distributed system where different actors, acting autonomously on behalf of a user, and each pursuing their own goals, need to interact in order to achieve their goals. Communication and negotiation are paramount. Furthermore, the number and behavior of participants cannot be fixed a priori and the system can be expected to expand and change during operation, both in number of participants and in the amount and kind of knowledge shared. These characteristics indicate a situation for which the agent society paradigm is well suited and therefore the methodology we propose can be applied.

6.4.1.1 Coordination Level

The aim of the Coordination Level of the OperA methodology is to determine which coordination type best applies to the characteristics of the problem domain. This level guides the choice of a coordination model which will lead the further design process.

The determination of the type of coordination applicable to the domain takes an organizational perspective. That is, the situation is analyzed from the point of view of the society 'owner'. We have identified the specific characteristics of each coordination model that can be used to determine the applicable model for the domain. Table 6-2 summarizes the main characteristics of the three coordination types, discussed in section 6.3. This table can be used as a blueprint to determine the most appropriate coordination type for a problem domain during the analysis phase.

Table 6-2: Social characteristics of different coordination frameworks

	MARKET	NETWORK	HIERARCHY
Society purpose	Exchange	Collaboration	Production
Leading goals	Individual goals (determined by the agent)	Combination of individual with global goals	global goals (determined by the society)
Relation forms	Negotiation (e.g. Contract Net Protocol)	Variable within society norms and rules	Fixed (e.g. Action / Workflow loop)
Communication capabilities of agents	Interaction based on standards; communication concerns exchange only	Both the interaction procedures and exchange can be negotiated	Specified by design
Interface to outside world	Usually open for agents (after identification)	Admittance procedure for agents	Closed for agents; open for data (input and output)
Type of society	Open	Semi-open or semi-closed	Closed

The identification of the appropriate type of coordination enables the specification of the facilitation needs (roles and interaction patterns) of the domain and will point out the type of social laws and norms of conduct in the domain. In practice, Table 6-2, does not give one single exact answer about the coordination type of a domain, but supports the society designer by guiding the discovery process. That is, in many domains, the coordination type will be a combination of one or more basic types. The

designer can use the characteristics described in the table to ask the relevant questions. For example: what is the (coordination) purpose in this domain? Which kind of relationships are desired/available? How does the domain treat ‘outside’ agents?

Once the coordination model that best fits the domain situation has been decided upon, the corresponding society organizational architecture is taken as the starting point for the development of the agent society. These facilitation architectures are described in section 6.3 and specify the facilitation roles and generic interactions in the society, for the cases of market, hierarchy or network. Of course, often a domain will not correspond to a single type but will have characteristics of different coordination types. In these cases, the designer can decide to take the most likely model as a starting point and adapt it with components from other types, or design its own specific architecture.

So far, agent-based methodologies that consider both the social and the agent levels of analysis and design have been developed with a specific type of society in mind. For example, the Gaia methodology is intended to support the development of closed societies where all constituents are already known at design time and in which all agents are supposed to cooperate towards a common goal. That is, Gaia is suitable to the specification of societies that follow the hierarchic type but not so well suited to the development of market-like societies that allow for heterogeneous agents to participate. The OperA methodology represents an improvement in this aspect as it supports the modeling of different organizational types.

6.4.1.1.1 *Description of facilitation roles and scripts*

Once the basic coordination type for a society has been determined, the corresponding facilitation roles can be incorporated in the society design, using a library of roles.

Table 6-3: Characteristics of facilitation roles

ROLE	COORDINATION TYPE	OBJECTIVES	ABSTRACT NORMS
Matchmaker	Market	Match suppliers and seekers	Fairness Impartiality
Market master	Market	Regulate exchange interactions	Fairness Impartiality
Banker	Market	- Payment guarantees - Provide information on reliability of partners	Can never be an exchanging party
Gatekeeper	Network	Accept members	Trust Impartiality
Notary	Network	Register partnership	Impartiality
Monitor	Network	Monitor partnership	Impartiality
Root	Hierarchy	- Delegation of society goals - Monitor execution	<i>Absolute power</i>
Interface	Hierarchy	Regulate interaction with outside of society	<i>Defined by root</i>

The facilitation roles associated with the three basic facilitation types have been informally described in section 6.3 and are listed in Table 6-3, which also provides their main characteristics.

In relation to the abstract norms associated with hierarchical models, it should be noted that those, as the whole organization of such models, are determined by the root role, and therefore little can be specified at an abstract level. For each of the roles above, OperA specifications are available. As an example, in Figure 6-6 we give the OperA specification of the gatekeeper role associated with network societies.

Role: Gatekeeper	
Objectives	$O_1 := \text{accept-members}$
Sub-objectives	$\Pi O_1 = \{ \text{ask-intentions}(\text{applicant}, \text{role}), \text{describe-society}, \text{IF } \text{decide-acceptance}(\text{applicant}, \text{role}, \text{yes}) \text{ THEN } \text{negotiate-social-contract}(\text{applicant}, \text{role}, \text{SC}) \}$
Rights	decide-acceptance
Norms	OBLIGED(inform(applicant, decide-acceptance(applicant, role, YN)).
Type	institutional

Figure 6-6: Description of Gatekeeper role

Note that these are basic institutional roles and scripts, characteristic of the different coordination models. When designing a specific system, architects are free to add or delete institutional roles, or to change specific characteristics of the roles. For instance, the role of matchmaker is often also present in network architectures.

6.4.1.1.2 Analysis of the application example

The following characteristics of the Knowledge Market example are relevant for the determination of the appropriate coordination framework. The system should typically support collaboration and synergy between members of the Non-Life Product Development Group and enable these participants to fulfil their own objectives. That is, collaboration is necessary in order to support knowledge exchange, while certification mechanisms are needed to verify the identity of participants and contribute to develop trust. Furthermore, participants want to be able to determine their own exchange rules and to be assured that there is control over who are the other participants in the environment³⁹. In this situation, a market framework is not really suitable because negotiation in a market follows fixed rules that participants must follow. Moreover, participation is open to any agent, and restriction of role or access is not possible. Also the hierarchical model can be rejected because it imposes a fixed partnership relation that is not possible, since partners and sources are not a priori known. Matching these requirements to the coordination characteristics listed in Table 6-2 shows that a **network** framework is the most appropriate coordination

³⁹ For a detailed explanation of this requirement, refer to the user survey on the KennisNet described in chapter 7.

mechanism for the Knowledge Market. Therefore, the Knowledge Market agent society will be built over a network basis that has been described in section 6.3.2.

6.4.1.1.3 Summary of Coordination Level

In short, the Coordination Level starts with the analysis of the social characteristics of the domain which results in the determination of the purpose, goals, relation forms and communication requirements for the domain. These are at this level used to determine the facilitation architecture of society, which consists on:

- The choice of a facilitation type: market, hierarchy or network, and
- The identification of the basic facilitation roles and interaction structure, associated with the coordination type

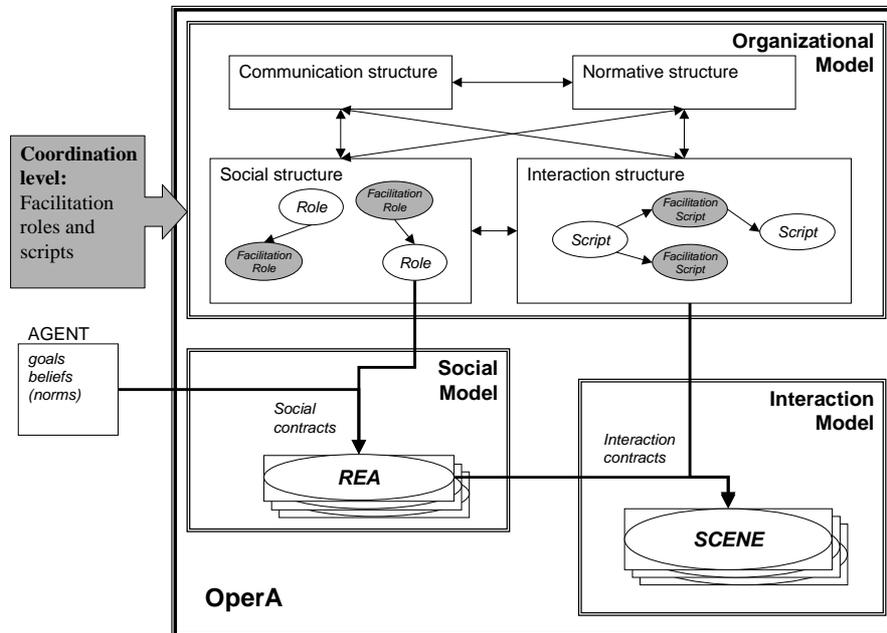


Figure 6-7: Contribution of coordination level to OperA architecture

In chapter 3 we have used a road map diagram to describe the components and relations of the different parts of an OperA model (cf. section 3.3.1, figure 3-2). In this chapter we will reuse that figure to illustrate how an OperA model is obtained from the application of the methodology. Figure 6-7 depicts the contribution of the Coordination Level to the overall OperA architecture, using a dark gray background for the components which specification is the result of this level. That is, the effect of the Coordination Level to the design of an OperA model is the choice of a basic coordination model for the system and the specification of institutional roles and scripts associated with that model. In the Knowledge Market example, following the choice for a network model, the roles of gatekeeper, notary and monitor are introduced in the system, as well as the scripts for member admittance and contract

registration. In the remainder of this chapter we will reuse this figure to describe the effect of each methodology step in the architecture of an OperA system, and therefore demonstrate that the methodology covers the complete model.

6.4.1.2 Environment Level

At Environment Level, the external behavior of the society, that is, the interaction between society and its environment, is analyzed and described. Results of this level include the identification of the global functionality of the society and the specification of domain ontologies. Organizations and their environments are interdependent, and each influence the other. In characterizing the environment of an agent society, the society designer has to decide which level of analysis is appropriate. In this step, the OM architecture chosen in the Coordination Level is refined with the domain-specific operational roles, norms and ontologies that describe the objectives for the society as envisioned by the society owner.

6.4.1.2.1 Requirements and use cases

A way to express the behavior of an agent society with respect to its environment is in terms of the expected functionality of the society, that is, what is the society expected to do or produce. Stakeholders are entities or systems in the environment that have goals or expectations towards the society. These need to be identified in order to evaluate their requirements and expectations towards the society. Overall requirements and scenarios or use cases are often used in this context⁴⁰.

The determination of the overall objectives of the society follows a process of elicitation of functional (*what*) and interaction (*how*) requirements. Furthermore, the needs and objectives for the integration with the environment must be identified. That is, what kind of interfaces are needed, whether the society must link to legacy systems, and possibly, the identification of user characteristics and skills.

In the Knowledge Market society, stakeholders are the non-life insurance department that has initiated the knowledge exchange system, and the members of the department that want to seek or offer knowledge within the group. Furthermore, the Knowledge Market must be able to communicate with the existing knowledge repository where knowledge items can be stored and consulted. The stakeholders have formulated several requirements which can be summarized as follows:

- The *non-life department* aims at supporting collaboration and extending synergy, and at the preservation, validation and organization-wide availability of existing knowledge.

⁴⁰ Requirement Engineering for multi-agent systems is currently an important topic of research and several methods have been developed, for example the Requirement Language described in [Dastani et al., 2001].

- *Knowledge owners* are willing to share their knowledge within a group they feel they can trust; that is, they wish to be able to decide on sharing decisions and conditions; furthermore, added-value of the sharing effort and fair exchange is a must (that is, the feeling that one is rewarded for sharing).
- *Knowledge seekers* need to obtain knowledge, but are not aware of all existing knowledge and knowledge owners; they also wish to be able to decide on acquisition conditions and partners and furthermore an accreditation and certification mechanism is desired, that enables them to check the level of trust and knowledge of partners.

6.4.1.2.2 *Communication and ontologies*

The next step in the environment level is to identify the functional requirements of the domain and the concepts and relationships relevant in the domain. The different stakes users will have in the society determine the requirements. The aim of the knowledge exchange network is to exchange knowledge represented as (XML)-documents describing reports, people, applications, web sites, projects, questions, etc⁴¹. Ontologies are needed to describe the different concepts relevant to the system. The following ontologies are needed for each society:

1. **OperA level ontology**, describing concepts specific of OperA such as role, role dependency, interaction script, role objective, interaction scene result, etc. These concepts were formally described in chapter 5.
2. **Model level ontology**, describing concepts related to the coordination framework of the society. It includes concepts as gatekeeper in a network society, banker and auction in a market society, and controller and service-request in a hierarchy. For the network model used in the Knowledge Market example, this ontology describes the concepts of gatekeeper, owner, seeker, source, etc.
3. **Communication ontology**, describing the illocutions to be used in the communication. The communication ontology is often independent from the domain but can be customized to the specific needs of a domain. It includes concepts as inform, request, refuse, etc.
4. **Domain level ontology**, describing concepts related to the application domain. In the Knowledge Market example, it must describe concepts related to non-life insurance and specific concepts used at Achmea.

The first three ontologies above are generic and can be reused in different OperA applications. The last ontology refers to the application domain and is therefore specific to a particular application. However, current efforts in ontology research are

⁴¹ This type of exchange ‘goods’ imposes constraints on the task and communicative components of agents since it demands a complex matching mechanism, because matches are not only at keyword level but require knowledge about relationships, processes, etc. However, this lies outside the scope of this chapter and will not be further discussed.

directed to the development of domain ontologies for several domains, which can possibly be used in combination with OperA societies.

6.4.1.2.3 Stakeholders and roles

Besides the roles directly related to the coordination model - that is, the facilitation roles identified in the Coordination Level - it is necessary to identify other society roles and their objectives and norms. These **operational roles** are related to the different stakeholders of the society and, as such, serve as a link between the society and its environment.

In this step, the aim is to characterize the different stakeholders of the society. In principle, each stakeholder is represented in the society by a role. Furthermore, stakeholders are often related to other stakeholders, which will result in dependencies between the roles representing them. In order to capture stakeholder objectives and relationships, **stakeholder tables** can be used. These tables describe the 'soft' objectives of the stakeholders, their proposed ways to achieve those objectives, and their dependency on other stakeholders to realize them. The stakeholder table for the Knowledge Market is depicted in Table 6-4.

From the table it can be seen that some of the objectives of the non-life department cannot be realized through dependencies on any of the other stakeholders. This indicates that specific roles must be defined to allow for the realization of these objectives. Stakeholder tables are the base to specify **role tables**, that describe the roles associated to stakeholders, the role objectives and the role dependencies. Role tables indicate for each role, its name, its relation to the society (that is, the rationale for the existence of the role), an informal description of its objectives and whether the realization of those objectives is dependent on other roles (i.e. the role dependencies). Objectives described in a role table are a concretization of the objectives described in the stakeholder table, in the sense that necessary activities are made explicit in terms of the concepts defined in the ontology. For example, 'join society' in itself is not an objective of any stakeholder, but necessary in order to realize their objectives. From the society, the role of applicant is needed to represent agents that are not (yet) enacting any role but are interested in joining the society.

Table 6-4: Stakeholder table for the Knowledge Market society

STAKEHOLDER	OBJECTIVES	DEPENDENCIES
Non-life department	Validate knowledge	Gatekeeper, Editor
	Distribute knowledge within group	Knowledge seeker, Knowledge owner
	Distribute knowledge outside group	Visitor
Knowledge seeker	Obtain knowledge	Non-life department, Knowledge owner
Knowledge owner	<i>Get recognition through:</i> Provide help Publish own knowledge	Knowledge seeker Non-life department Editor

Role tables are equivalent to, and can in principle be replaced by *actor diagrams* as in the Tropos methodology [Giunchiglia et al., 2002b]. The objectives of each role are identified based on the characteristics expressed by stakeholders and requirements. To support the identification and specification of agent roles, role catalogues, providing commonly occurring role descriptions, will be developed. In the Behavior Level phase, role tables are the basis for the specification of explicit role descriptions and role dependency graphs, according to the OperA framework.

In the Knowledge Market example, the roles of *knowledge owner* and *knowledge seeker* are directly related to the stakeholders of the society. The roles of *editor* and *visitor* can be deduced from the requirements of the non-life department. *Editors* are responsible to determine the validity and degree of expertise of knowledge items and knowledge owners, and support browsing of knowledge. *Visitors* represent people who are not members of the group and are allowed to consult the knowledge repository but cannot request knowledge help or publish items. Furthermore, the roles of matchmaker, notary, monitor, and gatekeeper related to the choice of a network model during the coordination level, will provide the facilitation layer of the Knowledge Market society. The *gatekeeper* determines whether an agent can participate in the exchange or not and what kind of role can be fulfilled, the *matchmaker* matches supply and demand of knowledge between participants and the *notary* registers and oversees the exchange commitments decided upon between participants. In Table 6-5 we give an example of role table for the Knowledge Market.

Table 6-5: Role table for the Knowledge Market society

ROLE	RELATION TO SOCIETY	ROLE OBJECTIVES	ROLE DEPENDENCIES
Applicant	Potential members	Join society	Gatekeeper
Knowledge seeker	Represents stakeholder: Knowledge seeker	Request partner	Matchmaker
		Browse repository	Editor
Knowledge owner	Represents stakeholder: Knowledge owner	Announce assistance	Matchmaker
		Publish knowledge item in KB	Editor
Editor	Realization of validation objective of non-life department	Validate items in repository	Knowledge owner
Visitor	Realization of distribution objective of non-life department	Browse repository	Editor
Gatekeeper	From network model	Control society members	Seeker, owner, visitor, editor
Matchmaker	From network model	Match supply and demand of knowledge	Seeker, owner
Notary	From network model	Register partnership	Seeker, owner, monitor
Monitor	From network model	Monitor partnership	Seeker, owner, monitor

6.4.1.2.4 Social Norms

In OperA, norms are first class entities that can be followed or violated by the agents. Although we consider the possibility of normative agents [Castelfranchi et al. 2000], which consciously are able to deliberate about norms, this is not a requirement on agents in OperA systems: norm compliance or violation can as well be the result of ignorance or inability of the agent to understand or adapt to the norm. In OperA norms regulate the behavior of role enacting agents in a society and are categorized in three types: **obligations**, **prohibitions** and **permissions**. Obligations and prohibitions restrict the possible actions (are derived situations) while permissions are generally used to indicate the conditions under which an action can be performed. Norms are defined both at role as well as at interaction level.

In the OperA methodology the first step to the specification of society norms is the analysis of the normative expectations and requirements for the society. That is, the design of the society must include the determination of the ethics of the society, what behaviors are good and bad and the mechanisms for insuring that generally good behavior occurs and bad behavior is minimized. The method used to capture society norms is based on the Norm Analysis Method [Stamper et al., 1988], [Salter, Liu, 2002]. Considering the different objectives and expectations on the society behavior, society norms are identified in four steps:

- **Responsibility analysis:** Involves the determination of the type of norm (obligation, prohibition or permission) and of the society roles responsible for initiating or acting the norm.
- **Resource analysis:** In some situations, it may be necessary to identify the type of resources needed to realize the norm. These can be for instance, information or knowledge resources, access rights or the availability of some artifact.
- **Trigger analysis:** Determination of the events that trigger a norm, the state of affairs that must hold when the norm is triggered and the state of affairs that must hold after the action specified by the norm is successfully executed.
- **Norm specification:** Using the information elicited in the previous steps, the norm can be specified using a semi-formal format: **whenever condition then role is deontic-operator to do action**. Norm expressions can straightforwardly be transformed into the formal rules described in chapter 3 (section 3.3.3).
- **Sanction:** Description of the possible sanction associated with the failure to comply with a norm.

For example, in a Knowledge Market society, requests from seekers must be sent to possible knowledge owners in order to facilitate interaction. Responsibility analysis of this situation identifies the obligation for the matchmaker to see to it that when a request arrives, it is distributed to relevant knowledge owners. This obligation generates the necessity for the matchmaker to access the user profile database in order to find out relevant partners. The trigger for the obligation of the matchmaker to distribute that request is the event of the arrival of a seeker request. After successful action, the knowledge owners will have been informed of the request. The normative statement describing the library norm analyzed above is: **whenever seeker-request**

then matchmaker is obliged to do distribute-request-to-relevant-partners. This example is summarized in Table 6-6, based on [Salter, Liu, 2002].

Table 6-6: Norm analysis example

NORM ANALISYS	
Situation	Handling of seeker requests
Responsibilities	Initiation: knowledge seeker Action: matchmaker
Resources	Access to member profile database
Triggers	Pre: seeker issues request Post: owners are informed of request
Norm specification	whenever seeker-request then matchmaker is obliged to do distribute-request-to-relevant-partners

The application of this method results in a semi-formalized set of concrete society norms that correspond to the abstract ethical principles that hold in the society. In OperA, at the OM level, norms are assumed to be external (expectations, behaviors or prescriptions) to the agents.

Specific implementations of the resulting society model will specify norms in different ways, either as part of the architecture of the agents or external to the agents. This translation is dependent on the domain of the institution and therefore the translation rules depend on, e.g., the ontology for that domain. For instance, the norm to pay when you have bought a product can be implemented by restricting the actions available to agents in the society after the buying action to just the paying action. However, one might also implement this norm by not allowing an agent to leave the institution before he has paid (in case he bought something). This means that the agent can still perform all kinds of actions, but always has to pay at some time [Dignum, 2002a]. Obviously, if the conventions and norms are hard-wired into the agent's protocols, it cannot decide to violate the norms. To adopt a norm does not necessarily imply to follow it. The concept of 'adopting a norm' means that the agent decides to generate goals and plans on the basis of its belief that there is such a norm and that it also concerns the agent itself. However important, we will not pursue these aspects here but refer the interested reader to [Dignum, 2002a], [Vázquez-Salceda, 2003], [Castelfranchi et al., 2000].

6.4.1.2.5 Summary Environment Level

At the Environment Level, the main characteristics of a society are identified through the analysis of the (expected) external behavior of the system. This process is based on:

- The output of the Coordination Level,
- The identification of stakeholders,
- The identification of use cases describing overall requirements, and the
- Analysis of the ethical or normative behavior expected in the society.

As output of the Environment Level, a generic organizational architecture of society is obtained, which includes

- The identification of society stakeholders and overall requirements,
- The specification of the communication primitives needed in the domain (ontologies and illocution primitives),
- The identification of organizational roles associated with stakeholders and their characteristics (described in role tables),
- Identification of the ethical or normative behavior expected in the society.

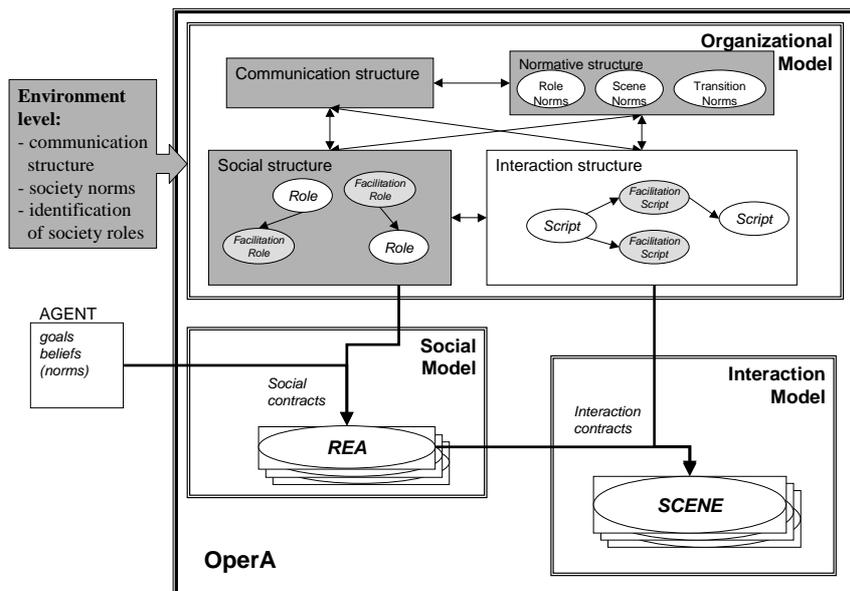


Figure 6-8: Contribution of environment level to OperA architecture

Figure 6-8 depicts the contribution of the Environment Level to the overall OperA architecture. We use a light gray background for components previously identified. As result of the Environment Level, the coordination structure is determined for the model, all organizational roles and their overall objectives are identified, and the organizational norms are described. In the Behavior Level, organizational roles will be further specified and social norms will be associated with specific roles, scripts and transitions.

6.4.1.3 Behavior Level

The purpose of the Behavior Level is to refine the society model obtained in the previous levels with the role specifications, interaction patterns and the overall interaction structure that represents and enables the aims of the system. That is, at this level a formal OperA conceptual model for the OM of the society is constructed from the roles, objectives, requirements and norms identified in the previous levels.

6.4.1.3.1 Roles

For each role, its objectives are analyzed from the perspective of the stakeholder associated with the role and a list of sub-objectives is generated, which can possibly indicate a dependency to other roles. The analysis of role objectives is done using some of the reasoning techniques proposed by the Tropos methodology [Giunchiglia et al., 2002b]:

- **Means-end analysis** is applied for the discovery of the means (resources, interactions) that contribute to achieve the objective. This results in the refinement of objectives into sub-objectives and in the identification of the rights of a role.
- **Contribution analysis:** identification of other objectives (from the role itself or from other roles) that, from the perspective of the role, can contribute positively or negatively to the realization of the objective. A possible positive contribution is an objective whose realization implies the realization of the objective under analysis. In the same way, a negative contribution is an objective whose realization will prevent the realization of the objective. Partial contributions are also possible (i.e., contribution relations between sub-objectives).

The results of this analysis are also used later for the definition of the interaction structure (scripts and transitions) of the agent society. Finally, a formal role specification can be generated from the objectives, sub-objectives, rights and norms that have been identified for the role. Role objectives identified during the Environment Level, and sub-objectives generated from the means-end analysis, are formalized in landmarks using the society ontologies. In the same way, formal role norms result for the translation of the role norms identified through Norm Analysis into LCR expressions using the concepts defined in the society ontology. Table 6-7 describes how role definitions given in chapter 3 can be generated from the methodological analysis of the domain.

Table 6-7: Generation of role definitions

ROLE DEFINITION	
Role id	Identified in Environment Level
Objectives	Formalization of objectives identified in the role table
Sub-objectives	Result of means-end analysis for each role objective
Rights	From means-end analysis and norm analysis
Norms	From the Norm analysis in Environment Level
Type	Roles associated with the coordination model are institutional, and operational roles are in principle external.

For example, the description of the role ‘Knowledge Seeker’ as given before in role table depicted in Table 6-5, is translated into the semi-formal role definition shown in Figure 6-9⁴². The specification of the role describes its characteristics

⁴² We choose to use a semi-formal notation here for readability. A complete description of the formalization of OperA is given in chapter 5.

according to the requirements identified by analysis of the domain and reflects the view of the society designer on the role. These definitions can then be transformed in a formal definition using the LCR logic and used in the formal specification and verification of the Knowledge Market model.

Role: Knowledge Seeker	
Role id	k-seeker
Objectives	o ₁ := obtain-knowledge o ₂ := browse-repository
Sub-objectives	Πo1 = { request-partner(question, partner-list), choose-best-partner(partner-list, partner), get-answer(question, partner, answer) }
Rights	access-repository
Norms	IF agreed-share(partner) THEN OBLIGED publish-repository(answer)
Type	external

Figure 6-9: Role definition for Knowledge Seeker

6.4.1.3.2 Interaction Scripts

Role dependencies indicate that roles rely on other roles for the realization of some of their (sub-)objectives. Such interactions are described in OperA as interaction scenes. The specification of scene scripts is done using the information on role dependencies and norms elicited during the Environment Level. Interactions related to facilitation aspects of the society, such as the request to the match maker for potential exchange partners or the registration of a contract, can be provided in libraries associated with the coordination model chosen. Such standard interactions form a basis for the specific interactions in the society and can obviously be modified and/or renamed to accommodate the specific requirements of the domain. Interactions between operational roles can be deduced from the role dependency relations identified during the Environment Level and must include the constraints imposed by the norms regulating the roles involved in the interaction.

Table 6-8: Scene table for the 'Request Partner' scene

SCENE TABLE	
Scene identifier	request-partner
Roles	Seeker Matchmaker
Description	Seeker requests possible partners for a knowledge need
Results	Seeker obtains (possible empty) list of possible partners for knowledge exchange
Patterns	Seeker describes knowledge need and Matchmaker distributes request to owner list and Matchmaker receives answers and Matchmaker gives seeker list of partners
Norms	whenever seeker-request then matchmaker is obliged to do distribute-request
Rationale	modified version of request-partner from network model

The problem remains of deciding which interaction scenes are needed. In principle, one interaction scene must be specified for each role dependency identified. However, depending on the domain, several dependencies can be combined into one scene. Moreover, interactions requiring different settings of roles, should also be taken as different scenes. We use **scene tables** to describe the different components of the society scenes, identified previously. Interaction patterns describe the way to achieve a scene result, envisioned by society design. Interaction patterns can be specified as UML diagrams (an example can be seen in chapter 3, figure 3-24). Table 6-8 illustrates the scene table for the 'Request Partner' scene in the Knowledge Market example.

These tables are the input to the formal specification of scene scripts in OperA, which is achieved through the formalization into LCR expressions, using the concepts defined in the society ontology, of results, patterns and norms associated with a scene according to the scene table for the domain. Furthermore, the process of formalizing scene tables also results in a refinement of the communicative and normative structures. That is, it must be checked whether the ontologies and communicative acts identified in the environment level are rich enough to describe the necessary interactions described in the scene tables, and in the same way whether society norms are enough to describe the constraints and guidelines for interaction identified. If this is not the case, a refinement cycle is needed. For example, in the formal script for the scene Knowledge-request scene described in Table 6-8 we assume that the communicative and normative structures contain all necessary elements used (e.g., 'knowledge need', 'owner list', 'receives', 'describes', 'list of partners', etc).

6.4.1.3.3 Interaction Structure

The last step is to specify the order of interaction scenes and the movements of the roles along the interaction structure described. Interaction structures specify the combination of scenes, that is, how the scenes must be related to each other, and how must agents performing roles proceed to a following scene. As a whole, the interaction structure describes the complex activities that, from the perspective of the society designer, are necessary to realize the overall objectives of the society. Furthermore, if one scene is directly followed by more than one other scene, is the realization of both needed? And in such a case, in which order? Or is the realization of either one or the other sufficient? And, under which conditions should a new instance of a scene be initiated? Such complex activities are specified by establishing the following relationships between scene scripts [Esteva et al., 2001]:

- **Causal dependency:** a scene can only happen after another one (a contract can only be registered after it has been negotiated)
- **Synchronization:** AND relations between scenes (both the announcement AND a request of knowledge must have been performed before a partnership between knowledge seeker and owner can be negotiated)
- **Parallelism:** OR relations between scenes (after admission, knowledge seekers can announce a request OR browse the repository OR leave the society)
- **Instantiation:** Conditions for creation of new scene instances (a new instance of partnership negotiation should be created for each request interaction)

Also the paths roles can follow in a interaction structure must be analyzed. The following aspects must be considered:

- **Role flow policies:** specification of the possible next scenes for a role leaving a scene and which roles it will be playing then.
- **Choice points:** specify the condition for a performing agent to choose between the possible next scenes for its role.

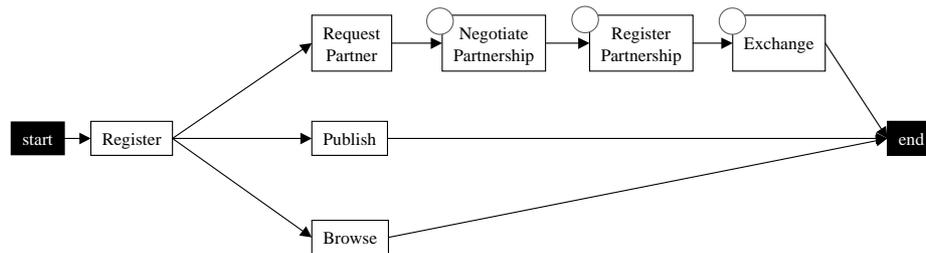


Figure 6-10: Interaction Structure for Knowledge Market

The above interaction structure relationships are taken from the ISLANDER framework [Esteva et al., 2002b]. ISLANDER presents several similarities with OperA, namely in the way relations between scenes are treated, in the fact that agent society can be viewed as a network of scenes. In fact, we have used ISLANDER concepts as basis for interaction scenes in OperA. Nevertheless, there are also quite a number of differences between the models, one of the most notorious being the fact that scenes in ISLANDER represent formal protocols, whereas in OperA scenes define landmarks for agent interaction. Therefore, we have decided not to use the same notation as ISLANDER, in order to not confuse the reader. Part of the interaction structure for the Knowledge Market, is illustrated in Figure 6-10.

6.4.1.3.4 Summary of Behavior Level

The Behavior Level consists on the analysis of the internal behavior of the system. This process is based on the Basic Organizational Model, obtained from the previous methodological levels (including coordination type, generic role descriptions and communication and normative primitives), and on the functional requirements for roles and interactions as described in use cases for the system.

The Behavior Level results in the complete specification of the Organizational Model of an OperA society, including:

- Specification of role descriptions for all society roles, including role objectives, norms and dependencies.
- Determination of the interaction scenes that will realize the interaction between roles necessary for the realization of their objectives.
- Refinement and specification of social norms, and their classification into role, scene or transition norms
- Specification of the society's interaction structure, including the description of conditions and requirements for transitions between interaction scenes

Figure 6-11 depicts the contribution of the Behavior Level to the overall OperA architecture. As result of the Behavior Level, the specification of the Organizational Model is completed, including the specification of all organizational roles, the determination of norm types and the formal specification of norms, and the design of the Interaction Structure (that is, the specification of interaction scripts and transitions).

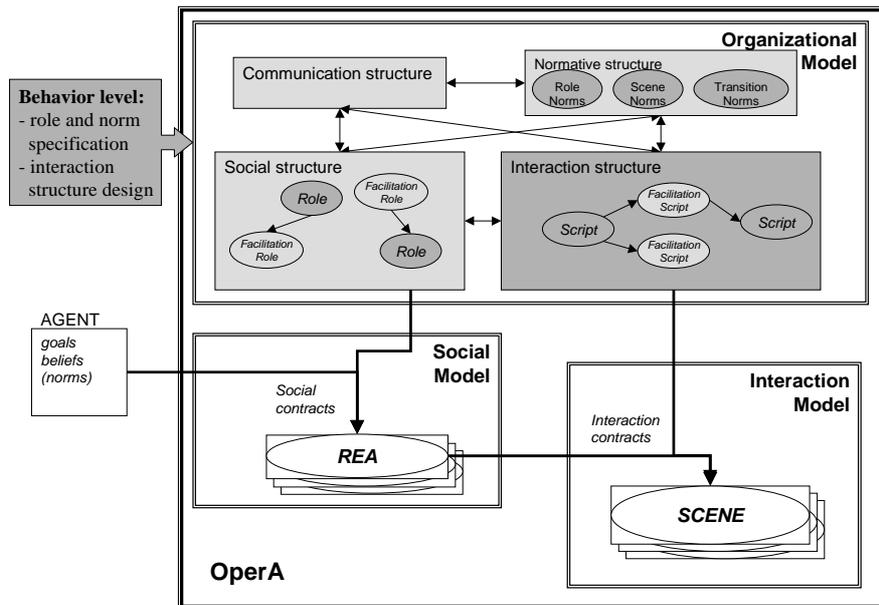


Figure 6-11: Contribution of behavior level to OperA architecture

6.4.2 Social Model design

In the previous section, we introduced the design methodology for the Organization Model of an agent society, based on the analysis of the domain and its formalization using the OperA model. As described in chapter 3, the OM alone is not enough to create an ‘alive’ society of interacting agents. That is, the OM merely provides the scenario the society, complete with scenes, role descriptions and templates for conversations. It is a static model, in the sense that there are (yet) no actors that can perform any activity. Furthermore, the OM describes the perspective of the society ‘owner’ on how the society should look like and behave, but it requires agents to effectively produce the desired results. It is therefore necessary now to describe how to design the process of appointing agents to enact the roles specified in the OM. In our opinion, the effective design of open agent societies requires the following aspects to be available [Dastani et al., 2003]:

1. A formal framework to specify the society structure and goals with verifiable and meaningful semantics, in a way that is independent from the participating agents.

2. Mechanisms through which prospective participants can evaluate the characteristics and objectives of the society (roles), to decide about participation.
3. Tools for individual agents to adapt their architecture and functionality to the requirements of an assumed role must be provided.

Point 1 is achieved with the design of the OM as described in section 6.4.1. The OM describes the intended organizational structure of the society and the laws that govern interaction among agents, in the same way as institutions are used in human societies to lend legitimacy and security to its members by establishing norms.

The design of the Social Model of an OperA society is related to points 2 and 3. The second point refers to the need to match agent and role objectives and functionality and the third point indicates that, once a decision has been reached that an agent will indeed enact a role, there must be ways to modify that agent in order to include the characteristics of the assumed role. In chapter 3, we have already discussed the problem of matching agents and roles and the current state of the art in this respect. In the following, we will describe the role negotiation scenes, where role enactment contracts are created.

As described in chapter 3, in OperA **social contracts** are used to describe the expected behavior of external agents in the society. Social contracts are generated in role negotiation scenes and specify the role enactment possibilities of agents. Social contracts describe the expected, agreed behavior of agents as enactors of society roles, that is, in OperA terms, a social contract describes an role-enacting agent, or **rea**.

From the society point of view, the agent will be ‘judged’ in terms of the commitments specified in the contract. That is, the agent is expected to realize the objectives and adopt the norms indicated in the contract. However, adopting a norm does not necessarily mean that the agent will follow the norm. That is, when one considers rational normative agents, at the moment that the norm becomes applicable, the agent will use its own current goals and motivations to determine whether following the norm is the best course of action (or whether, it is better for him to risk the sanction).

6.4.2.1 Role Negotiation Scenes

Role negotiation scenes describe how the attribution of roles to agents is to be performed, and which aspects are negotiable, according to the society design. That is, role enactment is not fixed in a protocol at design time, but can be negotiated between the agent and the society. This results in different role performances. Society design however, will specify the parameters and ranges that can be negotiated. Moreover, different society styles, will allow for a larger or narrower margin for negotiation. The design of the role negotiation scenes determines which and how role characteristics can be negotiated and has consequences for the performance of agents in the society. Scripts for the negotiation of social contracts are not usually explicitly represented in the society’s interaction structure, but are included in the ‘start’ scene which is part of every interaction structure. The design of role negotiation scenes is closely related to the type of society. On one extreme, closed societies will have no such scenes and agents are specified as part of the society design, with exactly the same characteristics

of the role itself. This is currently the case with most multi-agent systems. On the other extreme, ‘chaos’-like societies will allow any kind of enactment, and roles themselves will be minimally described.

A special kind of roles are the institutional roles, the roles responsible for the regulation and facilitation of society behavior. These roles were described in section 6.4.1.1.1. In the following we assume that agents enacting these roles, behave exactly according to the role description, that is, no negotiation is possible for those roles. We will concentrate on the negotiation of external, or operational, roles. Future research is needed to study the case of negotiation of institutional roles. Different societies will specify institutional roles, responsible for controlling the execution of social contracts (such as the role of ‘gatekeeper’ in a network society), differently which leads to different ways for the actual verification of social contracts.

The choices made in the OM regarding the design of the role negotiation scenes determine which are the aspects (objectives, sub-objectives, rights and norms) of the role for which the society is willing to negotiate, and within which parameters and which aspects are not possible to be negotiated and, therefore, must be accepted as-is. A basic algorithm for role negotiation is informally described in Figure 6-12. Of course, as any other scene, role negotiation scenes are designed to meet specific society requirements.

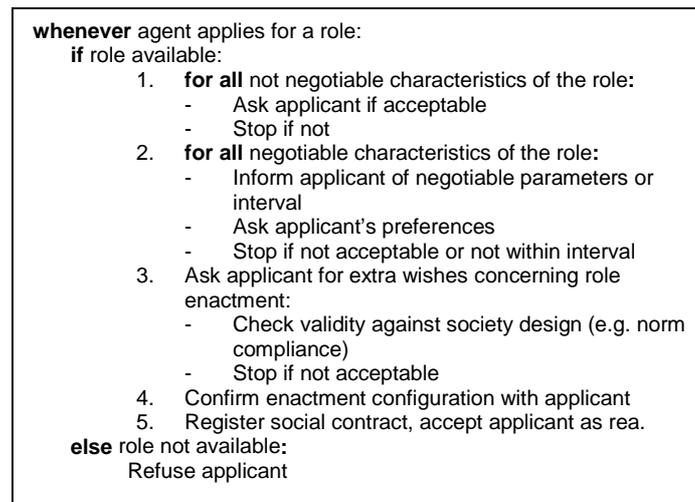


Figure 6-12: Basic role negotiation algorithm

Furthermore, the OperA framework starts from the assumption that an agent will incorporate all aspects specified in its social contracts into its architecture. How this is actually done is, however, not the concern of OperA. In chapter 3 we discussed initial research concerning the adaptation of agent architectures to requirements imposed by role enactment.

6.4.2.2 Summary of Social Model design

The design of the Social Model of an OperA society is in reality much more of an operational issue than a methodological one. That is, where for the Organizational Level, specific design steps, of increasing levels of detail, were identified which result in a complete society structure, the creation of the Social Model depends on the activities of specific agents, and is for the most part determined at ‘run time’. In summary, the design of the Social Model for an OperA society is based on:

- The role descriptions specified in the OM,
- The way role negotiation scenes are specified in the OM, and
- The characteristics of the agents that apply for society roles.

This means that one and the same Organizational Model will result in many different Social Models. Figure 6-13 depicts the contribution of Social Model design to the overall OperA architecture. Based on the Organizational Model specification of an agent society and on a set of specific agents, the social model design will describe the social contracts for society roles for those agents. The design of the agents themselves is outside the scope of an OperA model.

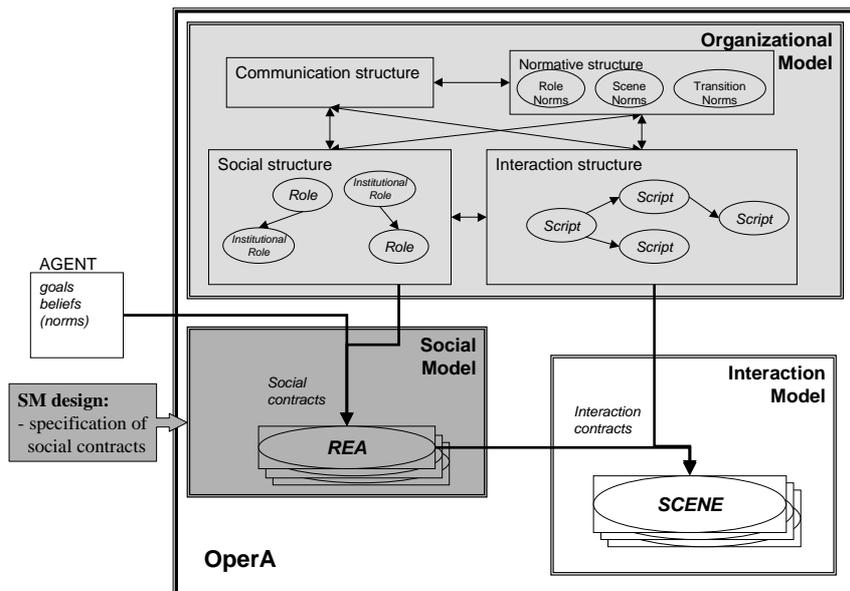


Figure 6-13: Contribution of Social Model design to OperA architecture

6.4.3 Interaction Model Design

As result of the specification of the SM, the agent society is now populated by a set of actual agents. That is, the role enacting agents specified in the Social Model can initiate interaction activities. When role enacting agents come together in an interaction scene, the actual interpretation of the scene script, that is the interaction protocol to be used must be agreed upon. In OperA, role enacting agents will, for each

scene, negotiate an **interaction contract** that defines their partnership, and fixes the way a specific interaction scene is to be played. Interaction contracts describe instances of scene scripts which inherit the organizational norms and objectives described in the interaction script and possibly extend or restrain it to accommodate the specific needs and desires of the participating agents.

The aim of the Interaction Model is therefore to set interaction contracts describing specific scene enactment agreements for the current agents enacting the scene roles. The basic idea is that agents participating in an interaction scene must first negotiate the specific protocol to play the interaction script specified in the OM. This protocol is fixed in an interaction contract and used to enact the scene. As for the Social Model, Interaction Model design is achieved at ‘run-time’ rather than described at design time.

6.4.3.1 Interaction Scene Enactment

As described in chapter 3 (section 3.6.2), the operationalization of an interaction script is achieved in two steps: negotiation and play. Firstly, during the **negotiation** part, the role enacting agents (reas) negotiate and agree on the protocol to be used in the actual play of the scene script. That agreement is fixed in an **interaction contract**, which forms the basis for interaction. The interaction contracts agreed upon will possibly describe extra norms for the reas involved, and fix the **interaction protocol** that completely specifies the enactment of the scene. Secondly, the actual **play** of the scene according to that protocol takes place. As with social contracts, it is assumed that agents will incorporate the regulations specified on interaction contracts in their architectures.

Interaction Contract	
Parties	A: rea(Anne, owner), B: rea(Bob, seeker), M: rea(matchmaker)
Scene	Request Partner
Clauses	<ol style="list-style-type: none"> 1. IF received(M, B, request-partner(question)) THEN OBLIGED M TO answer(M, B, accept-refuse) 2. IF accepted(M, B, request-partner(question) AND answer(KB, question, no) THEN OBLIGED request-available(M, A, provide-answer(question)) 3. IF informed(B, M, accept(A, provide-answer(question))) THEN OBLIGED B TO inform(A, thanks)

Figure 6-14: Interaction contract for scene ‘Request Partner’

Again, OperA does not specify any aspects related to how this incorporation can be effectuated, but leaves that to agent design. Nevertheless, system design assumes that agents will be able to comply to their contracts. For example, the contract specification in Figure 6-14 describes the enactment of the interaction scene ‘Request Partner’ depicted in Table 6-8.

This contract introduces extra confirmation steps, which were not required in the interaction pattern specified in the Organizational Model. These steps are possibly the result of the negotiation between the reas, that is, *rea(k-seeker, Bob, knowledge-request)* and *rea(k-owner, Anne, knowledge-request)*, and *rea(matchmaker)* currently playing the scene. Note that the matchmaker role is an institutional role. In this case

the enactor is irrelevant, because we assume that institutional roles are always played according to the society design specified in the OM. The agreed protocol for the realization of this interaction scene is given in Figure 6-15.

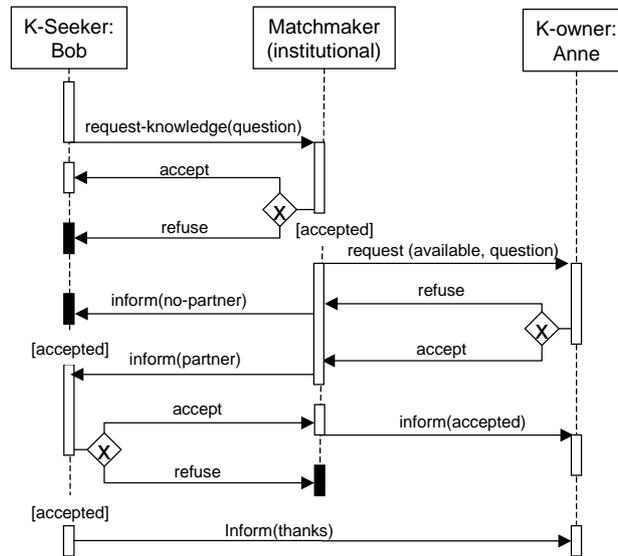


Figure 6-15: Agreed protocol for 'Knowledge Request' scene

This protocol enables the role-enacting agents involved to act the scene. In the sequence of the successful realization of this scene protocol, Bob and Anne will move on to the 'Negotiate Partnership' scene during which they will establish the interaction contract relative to their specific exchange of knowledge. An example of such contract is described in chapter 7, section 7.2.4.3.

6.4.3.2 Summary of Interaction Model design

The creation of an Interaction Model depends on the activities of specific role enacting agents, guided by the description of scenes in the scene scripts specified in the OM. That is, the generation of an Interaction Model for an OperA society depends on:

- The specific role-enacting agents and their role enactment agreements, as described in social contracts in the SM, and
- The scripts for interaction scenes specified in the OM.

For the same agent population of an OM, many different Interaction Models are possible. Similarly to the SM, this allows to incorporation of the specific requirements and characteristics of agents and enables for a more realistic treatment of autonomy. Figure 6-16 depicts the contribution of Interaction Model design to the overall OperA architecture. Based on the OM specification of an agent society and on a population of role-enacting agents described in the SM, the interaction model design will describe the interaction contracts for scene scripts.

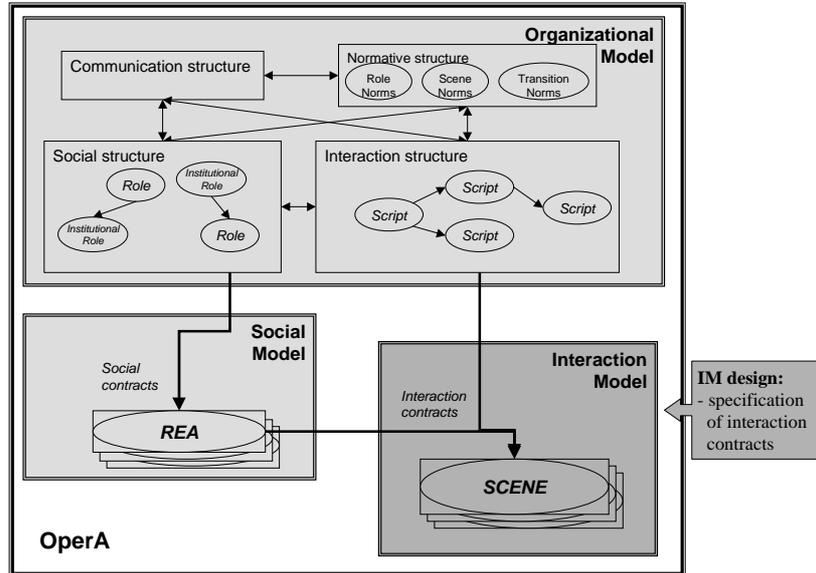


Figure 6-16: Contribution of Interaction Model design to OperA architecture

6.5 Society design update

As we argued in section 6.1, methodologies for agent societies must incorporate means for observation and construction. An agent society model results from the observation of the behavior of a system and can be used in two main ways: as basis for the **construction** of software multi-agent systems, and, as **explanation** and simulation of an existing system.

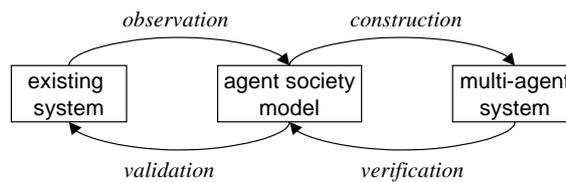


Figure 6-17: Construction and observation of agent societies

Furthermore, because we assume that heterogeneous agents can join an agent society and, therefore, the actual behavior of the system cannot be predicted a priori, mechanisms are needed to verify the activity of the multi-agent system against the society model and adapt it if necessary. Figure 6-17 illustrates the relation between systems. A society model for an application domain results from the application of a set of explanatory principles to the observed properties of an existing system or community. The model includes the description of the coordination, environment and behavior characteristics of the observed system. Using this model, a multi-agent

system can be constructed that will perform the modeled functionality. The resulting system can again be observed and the original model verified and possibly adapted. That is, the introduction of agents will influence the behavior of the observed system, creating the necessity for a dynamic engineering cycle.

The engineering of agent societies must therefore be able, on the one hand, to incorporate changes in the environment, and therefore in requirements, objectives and norms of the society, and on the other hand, to accommodate emergent behavior related to the activity of actual agents in the society.

6.6 Conclusions

We have presented a global methodology for the design of the organizational model of OperA agent societies. The method takes an organizational perspective as starting point and describes the implications of the coordination model of the organization for the architecture and design method of the agent society being developed. Furthermore, the approach specifies the development steps for the design and development of an agent-based system for a particular domain. It provides a generic frame that directly relates to the organizational perception of a problem and allows for existing methodologies to be used for the development, modeling and formalization of each step.

Although there are several agent-based software engineering methodologies available these are often either too specific or too formal and not easily used and accepted. We believe that because of its organizational-oriented approach our methodology will contribute to the acceptance of multi-agent technology by organizations. One contribution of our research is that it describes the implications of the coordination model of the organization for the architecture and design method of the agent society being developed. With respect to the specification of individual agents, that will fulfil roles in the society, OperA can be extended with existing methodologies for the development, modeling and formalization of those agents.

Further research and practice is needed to further refine the methodology. We plan to apply the methodology to different domains in order to gain better view on the problems and capabilities of it. Specifically, we plan to develop tailored frameworks for specific application domains such as Knowledge Management, Workflow Management and e-business. We also intend to develop libraries of conceptual interaction patterns and agent roles. These libraries will improve and facilitate the design of agent societies. Finally, we plan to look at the compatibility and integration of our ideas with current standardization efforts for agent development, such as Agent UML [Odell et al., 2001].

Chapter 7

Applications of OperA

*'That's what learning is, after all; not whether we lose the game,
but how we lose and how we've changed because of it and what
we take away from it that we never had before, to apply to other games.
Losing, in a curious way, is winning.'*
- Richard Bach, *The Bridge Across Forever* (1984).

In this chapter we will describe some practical applications of the OperA framework. We start this chapter with a discussion on the suitability of the agent paradigm to knowledge management in section 7.1. The first case study, Knowledge Market, described in section 7.2, concerns the application of OperA to Knowledge Management. The second case study concerns initial research on the application of OperA to non-organizational settings and is described in section 7.3. Although the case study presented in section 7.4 does not really concern a multi-agent system, but the development of a prototype for a single-agent personal assistant, we have decided to include this project here, as it illustrates the current practical application of agents to KM. Finally, in section 7.5 we present our conclusions.

Parts of the case study described in section 7.2 has been previously published in [Dignum, Dignum, 2003] and [Dignum, 2002b].

7.1 Introduction

Current developments in KM show a shift in the focus of KM from knowledge to collaboration. The aim of KM is no longer just the management of activities related to the creation, preservation and distribution of knowledge assets but the management and nurturing of collaboration between people. Such **collaboration management systems** call for approaches that are reactive and proactive in relation to the needs and expectations of its users.

The main motivating factor to the development of OperA was the design and implementation of support systems for Knowledge Management. There is currently an

increasing interest in the use of multi-agent concepts for KM, mainly motivated by the fact that, like multi-agent systems, KM collaboration environments can be seen as distributed systems where different actors, each pursuing its own goals, need to interact in order to achieve their goals and realize organizational objectives [van Elst et al., 2003a], [Bonifacio et al., 2002], [Gandon et al., 2000]. That, is, in KM environments, the integrity of the existing organizational structure and the autonomy of participants must be maintained, which calls for an autonomous and distributed representation of KM systems. Interactions in KM environments are fairly sophisticated, including negotiation, information sharing and coordination, and require complex social skills with which agents can be endowed. Furthermore, solutions for KM problems cannot be entirely prescribed from start to finish and therefore reactive and proactive problem solvers are required that can respond to changes in the environment, react to the unpredictability of business processes and act on opportunities when they arise.

These characteristics indicate the applicability of the OperA model to the development of KM environments that focus on the collaboration between people. Hence the prime application of OperA is the development of a system for knowledge exchange. Nevertheless, OperA appeared to be more widely applicable than its original purpose. It is a generic model for the design of multi-agent systems, which has the added-value of a formal semantics (cf. chapter 5) and a customized development methodology (cf. chapter 6), and as such is suitable to the development of multi-agent system for a variety of domains. Our two other case studies are therefore devoted to demonstrate: (1) the applicability of OperA to the design of non-organizational systems, which is illustrated by the second case study, CareCircle, and (2) the current state of the art in agent-based applications in KM: PAM, the single agent personal assistant described in our third case study. In this case, OperA can be used to model the interaction between human and software agents.

7.2 Knowledge Market

In this section, we will describe the development of a prototype agent society for knowledge exchange, using the OperA model and methodology. We start by describing the background and motivation for the project in section 7.2.1, after which we describe the system and its development in detail in section 7.2.2. We finalize this section with some comments on the practical evaluation of the project in section 7.2.3.

7.2.1 Background and Motivation

The Knowledge Center for Non-Life Insurance at Achmea is responsible for the development and maintenance of non-life insurance knowledge that will give business units across Achmea a leading edge in this area. The center has a need for efficient and goal directed sharing of information and knowledge. Members of the group (mainly insurance product developers and actuaries) are active across business units, geographically dispersed, and are not part of any existing organizational structure. Their knowledge and expertise are greatly valuable and useful to each other. But,

because people are not aware of each other's capabilities, often they will discuss their business problems with a direct colleague just because he/she happens to be conveniently close and not because he/she is the best person to consult with [Davenport, Prusak, 1998].

In 2001 a project was started with the objectives of structuring, initiating and organizing the sharing of knowledge between non-life insurance experts across Achmea by setting up a framework that assures the continuous availability of consistent and up-to-date knowledge [Dignum, 2002b]. The first steps towards the realization of these objectives concerned the development of a Community of Practice, the KennisNet, incorporating the facilitation of direct contacts between members and an intranet-based knowledge sharing server using existing technical infrastructure, a Lotus Notes network. The architecture of the server is depicted in Figure 7-1.

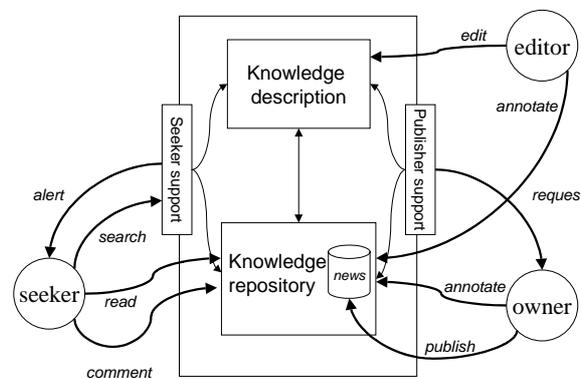


Figure 7-1: The architecture of KennisNet

Direct contacts between participants were formalized as quarterly workshops with the participation of all members that aimed to:

- assure the creation, maintenance and uniformity of domain knowledge (for example, by inviting external authorities in a relevant field and by facilitating structured discussions around a theme), and,
- enable participants to learn to know and appreciate each other, and feeding community feeling.

However, such a solution will only be effective when sharing is anchored into organizational culture and processes. Change management initiatives to enforce such culture are still crucial for the success of any collaboration support project. The development of the community of users as a Community of Practice follows the ideas of the SES model presented in the appendix to this dissertation [Dignum, van Eeden, 2003] (cf. Appendix A). The development of the knowledge repository was inspired by work by [Domingue, Motta, 1999], [Mentzas et al., 2001] and [Gandon et al., 2000]). Its functionality enables direct access to contents, publishing and browsing of knowledge items, and allows the implementation of facilities for discussion and broadcast of questions and requests.

7.2.1.1 Development of the knowledge repository

The development methodology used for the knowledge repository of KennisNet adapted the usual phases (analysis, design, implementation and evaluation) of system development to the specific case of knowledge management systems. As organizations themselves, the process of developing knowledge management solutions is a dynamic one, and should be continuously monitored and adapted to the changing goals and structure of the organization. That is, development must be seen as a continuous process, where each step may require changes in the previous ones. Furthermore, users and stakeholders must be involved in each level to assure the realization of a system that meets the needs and wishes of the organization and furthermore to assure that development keeps in pace with organizational and environmental changes.

The first step of the development, was to identify the strategic goals of the organization or group and the problems that hinder their achievement. Next, problems were analyzed from a knowledge perspective, and solutions were identified and tailored to the specific situation. The objectives and the format of the system were analyzed, discussed and decided upon in a participatory way, during several meetings in which all members of the group participated. Finally, the system was implemented in Lotus Notes. The portal to the KennisNet is depicted in Figure 7-2.

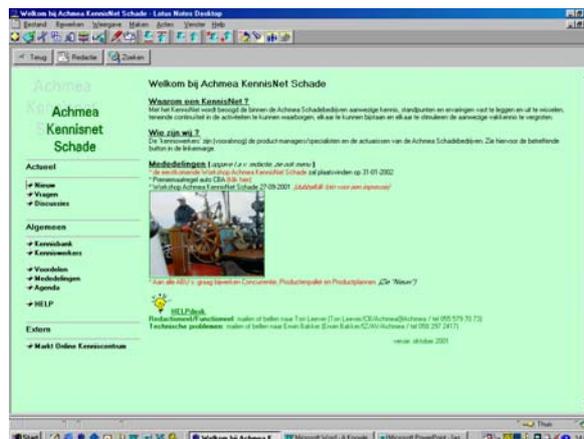


Figure 7-2: Portal to the KennisNet

The focus of the development was on the classification and presentation aspects of the repository. One of the requirements for the repository was that there should be a uniform representation of all types of knowledge items (i.e. documents, web sites, people, discussions, questions, news, databases and other applications, etc.). That is, one single search request should be able to retrieve documents, experts, related question from others, and so on. Description of knowledge items includes:

- **Identification:** Including name, datum of publication and status.
- **Content:** describing the actual meaning of an item, using aspects as: keywords, abstract, link classification ontology and comments.

- **Context:** Providing relevant information surrounding the creation and use of an item. Includes name of submitter, related projects, intended use and reasons for publication.
- **Structure:** Describing accessibility and use issues. Includes type of item (e.g. document, person, web site, application, etc.), location, contact person (who can tell you more about this item), and access conditions.

The KennisNet system provides automatic support filling in the descriptions of items. Search and retrieval in KennisNet is done on the meta descriptions of knowledge items⁴³. Furthermore, a classification ontology was developed following a participatory process to which all members of the community could contribute. Figure 7-3 depicts part of the ontology.

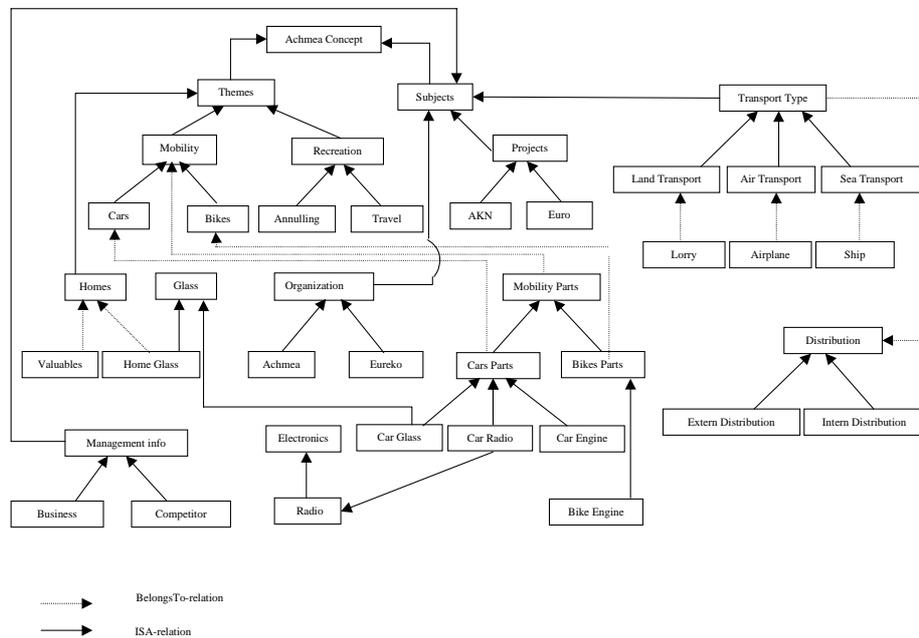


Figure 7-3: A part of the classification ontology of KennisNet

7.2.1.2 Evaluation of the knowledge repository

After the knowledge repository was fully implemented, following the centralized design described above, and running for around one year, we conducted a user satisfaction survey, which empirical results were reported in [Bondarouk, Pumareja, 2002], [Pumareja et al, 2003]. Two main conclusions from this survey were:

⁴³ The knowledge items themselves, such as people, are often not in electronic format.

1. The face-to-face structure was well appreciated and its value clear.
2. The added value and potential of the knowledge server was not always clear to the users, and the server is hardly used.

The survey pointed out that the main reason for this lack of use is that users need a more personal means of interaction to make them comfortable exchanging knowledge. The survey also indicated that knowledge owners prefer to share their expertise within a controllable, trusted group under conditions negotiated for the specific situation and partners. The community of users supported by the KennisNet operates across business unit boundaries, independently of the holding organizational structure. Sharing knowledge therefore implies that knowledge seeker and knowledge owner must be able to find each other and agree on the terms of the exchange.

Other recent studies elsewhere also show that success of knowledge sharing is dependent on the level of trust and dependency between community members and on the kind of culture prevailing in the society [Ali et al., 2002]. Knowledge is considered part of one's property and identity and therefore, people wish to keep the decision about sharing knowledge in their own hands, and want to be able to decide on a case by case basis whether an exchange is interesting to them or not. Furthermore, reciprocity in exchange is also an important aspect to be considered [Ahuja, Carley, 1998]. The above considerations can be summarized in the following requirements for an effective collaboration support system:

- Enable exchange within a controllable, trusted group under conditions negotiated by the partners for the specific situation.
- Knowledge seekers and owners must be able to find each other and agree on the terms of the exchange.
- As the value of a knowledge item cannot be fixed a priori, and knowledge requests are usually not fulfilled by a mere exchange of 'products', but require an, often not trivial, creation process, mechanisms are needed to dynamically determine exchange conditions.

In order to support the above collaboration requirements, it was decided to extend the knowledge repository with mechanisms for knowledge exchange and collaboration that keep ownership links between knowledge and people, for the support negotiation and valuation of exchange conditions⁴⁴. The design and implementation of the second phase of KennisNet focused primarily on the needs and desires of the users as formulated in the satisfaction survey. The requirement elicitation and analysis process for this second phase, focused on acquiring a good understanding of how the users do their tasks is necessary, especially if they work as a group, including the analysis of how they ideally communicate, search and acquire information from colleagues and other sources. The architecture of the resulting

⁴⁴ How much is a specific piece of knowledge worth, at a specific moment, under the specific circumstances holding and to the specific partners involved in the exchange.

system, including the Knowledge Market system, is depicted in Figure 7-4 and is further described in the following sections.

Motivated by the realization that another approach was needed to the problem at hand, the Knowledge Center started looking for more adequate models and tools to support collaboration in the group. As we have discussed in chapter 6, multi-agent systems can effectively meet the above requirements. Furthermore, the domain required, on the one hand, solutions to be independent of the design of individual components, representing the needs of each user (the internal autonomy requirement), and, on the other hand, flexibility and dynamic formation of exchanges was desired (the collaboration autonomy). These criteria motivated the choice of a development approach using the OperA model.

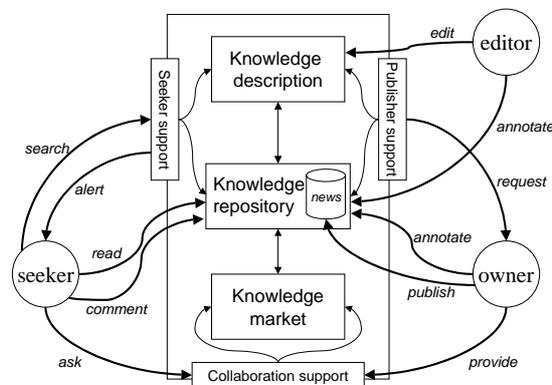


Figure 7-4: Architecture of KennisNet: Phase II

In the Knowledge Market, agents can ensure the preservation of individual needs and perspectives, and they can be employed to monitor and assist knowledge exchange, for example by taking care that deadlines are kept, reports are effectively exchanged, and eventual changes are communicated. Furthermore, agents are used to search the network for suitable partners, to publish and search results in the repository on behalf of their owners, and to monitor news and discussion groups.

7.2.2 Knowledge Market: Agent-based knowledge sharing

In this section, we will describe the development of the Knowledge Market, according to the OperA methodology. The Knowledge Market aims to support people exchanging knowledge with each other, in a way that preserves the knowledge, rewards the knowledge owner and reaches the knowledge seeker in a just-in-time, just-enough basis. In the remainder of this section the system, developed using the Opera Model, is described. The work presented refers mainly to the specification of the Organizational Model for the Knowledge Market, and describes the three methodological levels resulting respectively in the coordination, environment and behavior models. Unfortunately, due to strategic changes, the project was discontinued before we could implement the OperA model and therefore, specification of the Social and Interaction Models was never achieved. However, we

have done some initial work concerning the implementation of agent-based knowledge exchange systems, which is reported in section 7.2.3.

7.2.2.1 Coordination Level

At this level, the coordination type of the society is determined. The evaluation of KennisNet shows that collaboration and direct exchange between people are the crucial aspects to realize. People usually agree to share their knowledge with others if they feel that they will gain something from the exchange, and that they can trust their exchange partner. For example, a typical agreement within the KennisNet group says: *'I will share the result of a market survey I've just done with you, if you will let me have a copy of the report you are making for which you want to have those results'*. Therefore, a knowledge sharing system must be able to nurture and support the negotiation and realization of this kind of agreements.

In co-located groups, an exchange of favors relies on the assumption of stability of the community or group cohesiveness. There may be an inherent expectation that, since the relationships within the community are typically long lasting, sooner or later the favor is likely to be returned. However in distributed groups, although the common goal binding the members remains long-term, contacts and relationships may be relatively fluid with members entering and exiting as their task needs evolve. In this scenario, exchange of favors is likely to be based on reciprocity in a relatively short time-span [Ahuja, Carley, 1998]. That is, collaboration will need to be based on concrete, explicit commitments making clear what each partner is supposed to contribute and expects from the others.

Table 7-1: Facilitation roles in the Knowledge Market

ROLE	OBJECTIVES	ABSTRACT NORMS
Gatekeeper	<ul style="list-style-type: none"> - Accept participants 	<ul style="list-style-type: none"> - Obligated to check whether applicant is member of KennisNet - Allow only KennisNet members to request exchanges - Allow external visitors to browse repository
Notary	<ul style="list-style-type: none"> - Register agreements - Assign monitors - Impose sanctions 	<ul style="list-style-type: none"> - Obligated to register exchanges - Allowed to request exchange information from seekers and owners
Monitor	<ul style="list-style-type: none"> - Give alerts on deadlines and collaboration terms 	<ul style="list-style-type: none"> - Obligated to alert notary on sanctions
Matchmaker	<ul style="list-style-type: none"> - Register participants, skills and needs - Accept and distribute exchange requests 	<ul style="list-style-type: none"> - Obligated to distribute requests - Obligated to give distribution requests back to requester

Technology can facilitate knowledge sharing, but it is trust that enables it. Sharing knowledge therefore implies that seekers and owners must be able to find each other and agree on the terms of the exchange. Moreover, the value of a knowledge item cannot be fixed a priori but depends on many factors, and knowledge and information

requests cannot be fulfilled by a mere exchange of finished ‘products’ but require an, often not trivial, process during which the knowledge owner will develop the answer sought by the requester.

The above considerations indicate, as already described in chapter 6, section 6.4.1.1.2, that the most suitable coordination type for the Knowledge Market is the network model. However, because the domain requires support for users to find suitable partners, the role of matchmaker is also added to the facilitation layer of Knowledge Market.

Table 7-1 describes the facilitation roles of the Knowledge Market by adapting the generic features of network facilitation roles to the characteristics of the domain.

7.2.2.2 Environment Level

At this level, the global functionality and objectives of the society are determined. The starting point for this level, is the elicitation of use cases and requirements. Following the discussion in section 7.2.1.2, on the evaluation and extension of the KennisNet system, the following functionality is desired for the Knowledge Market:

- Possibility to share knowledge that is not available in the knowledge repository
- Support for coalition formation (in order to develop new solutions when knowledge is not available)
- Support for direct exchange between parties where the negotiation of exchange conditions happens on a case to case basis

These requirements indicate the need for both *direct exchange*, directed at finding relevant partners, and *indirect exchange*, through the repository, in which case the task of the system is to support publishing the results of direct knowledge exchanges. Furthermore, the Knowledge Market will use the same domain ontology as the knowledge repository, depicted in Figure 7-3. In chapter 6, we have already identified the stakeholders of the Knowledge Market and described their main requirements (cf. section 6.4.1.2.1.). The stakeholder table is depicted in Table 7-2.

Table 7-2: Stakeholder table for the Knowledge Market society

STAKEHOLDER	OBJECTIVES	DEPENDENCIES
Non-life department	Validate knowledge	Gatekeeper, Editor
	Distribute knowledge within group	Knowledge seeker, Knowledge owner
	Distribute knowledge outside group	Visitor
Knowledge seeker	Get help	Non-life department, Knowledge owner
Knowledge owner	<i>Get recognition through:</i> Provide help Publish own knowledge	Knowledge seeker Non-life department Editor

The analysis of the objectives of the different stakeholders identifies operational roles in the society, listed in the dependencies column. The characteristics of operational roles were then further specified in a role table, depicted in Table 7-3.

Table 7-3: Role Table for Knowledge Market

ROLE	RELATION TO SOCIETY	ROLE OBJECTIVES	ROLE DEPENDENCIES
<i>Applicant</i>	<i>Potential members</i>	<i>Join society</i>	<i>Gatekeeper</i>
Knowledge seeker	Represents stakeholder: Knowledge seeker	Request partner	Matchmaker
		Exchange knowledge	Knowledge owner
		Browse repository	Editor
Knowledge owner	Represents stakeholder: Knowledge owner	Announce offers	Matchmaker
		Exchange knowledge	Knowledge seeker
		Publish knowledge	Editor
Editor	Realization of validation objective of non-life department	Publish validated knowledge	Knowledge owner
		Distribute knowledge	Visitor, seeker
Visitor	Realization of distribution objective of non-life department	Browse repository	Matchmaker Editor

Another result of the Environment Level, is the specification of the normative characteristics of the society. Norms related to facilitation aspects, have been identified at the Coordination Level. Other society norms are the result of the requirements and characteristics of the domain. In chapter 6, we have described the process of norm analysis using as example the situation of handling requests by the matchmaker. Table 7-4 gives the result of norm analysis for different situations in the domain. This is not the complete listing of norms in the society, but describes the norms which have been implemented in the prototype.

Table 7-4: Norm analysis example

Description	Norm Analysis	
1. Handling of seeker requests	Responsibilities	Initiator: knowledge seeker Action: matchmaker
	Triggers	Pre: seeker issues request Post: owners are informed of request
	Specification	whenever <i>knowledge-request</i> then <i>matchmaker</i> is obliged to do <i>distribute-request-to-partners</i>
2. Answer knowledge requests	Responsibilities	Initiation: matchmaker Action: knowledge-owner
	Triggers	Pre: matchmaker issues knowledge request Post: owners answer request
	Specification	whenever <i>obtain-knowledge</i> then <i>knowledge-owner</i> is obliged to do <i>answer-request before deadline</i>
3. Privileges of visitors	Responsibilities	Initiator: visitor Action: all
	Triggers	Pre: visitor tries to issue exchange request Post: request denied
	Specification	always <i>visitor</i> is forbidden to do <i>knowledge-request</i>
	Sanction	Possible expulsion of visitor

4. Register agreements	Responsibilities	Initiator: partner (knowledge-seeker or -owner) Action: partner
	Triggers	Pre: partners have reached exchange agreement Post: contract is registered with notary
	Specification	whenever <i>agreement</i> then <i>partner is obliged to do request-register-contract(agreement description)</i>
	Sanction	Possible expulsion of partner
5. Apply sanction	Responsibilities	Initiator: monitor Action: monitor
	Triggers	Pre: Deadline expired Post: Sanction applied to breaching party
	Specification	whenever <i>contract-breached</i> then <i>monitor is obliged to do apply-sanction(breaching-party)</i>
	Sanction	
6. Handle registration requests	Responsibilities	Initiator: notary Action: notary
	Triggers	Pre: agreement registration request Post: registered contract and monitor assigned
	Specification	whenever <i>request-register-contract</i> then <i>notary is obliged to do register-contract and appoint(monitor)</i>

Note that in the examples above, we have, for the sake of simplicity, abstracted from the specification of attributes of the concepts used. Later, during the behavior level, it will be determined whether a norm refers to a role, a scene, a transition, or a group. For example, norm 4 refers to a group, that is, both the knowledge-seeker and the knowledge-owner roles are affected by this norm.

7.2.2.3 Behavior Level

Finally, the results of the previous methodological steps are combined and refined in the Behavior Level, to obtain a complete conceptual model for the Knowledge Market society. In the remainder of this section we provide a detailed description of the social and interaction structures of the Knowledge Market, which has also already been used in chapter 6 as an illustration for the OperA methodology.

7.2.2.3.1 Social Structure

In this section, we describe the social structure of the Knowledge Market. The role table obtained in the Environment Level is used as basis for the semi-formal role specifications for the external roles. These specifications, shown in Figure 7-5, can then be transformed in a formal definition using the LCR logic and used in the formal specification and verification of the Knowledge Market model. Note that, in order to keep the figures simple and readable, we have in some cases omitted the parameters of predicates. This must, of course, be part of the real specifications.

Role: Knowledge Owner	
Role id	owner
Objectives	o ₁ = register-skills(matchmaker, skills) o ₂ = answer-request(matchmaker, question) o ₃ = publish-knowledge(editor, knowledge-item)
Sub-objectives	...
Rights	access-repository
Norms	IF obtain-knowledge(matchmaker, question, deadline) THEN OBLIGED(owner, answer-request(matchmaker, YN, question) BEFORE deadline
Type	external

Role: Knowledge Seeker	
Role id	seeker
Objectives	o ₁ = request-partner o ₂ = exchange-knowledge o ₃ = browse-repository
Sub-objectives	$\Pi_{o_1} = \{ \text{get-potential-partners}(\text{question}, \text{partner-list}), \text{choose-best-partner}(\text{partner-list}, \text{partner}), \text{get-answer}(\text{question}, \text{partner}, \text{answer}) \}$ $\Pi_{o_2} = \{ \text{negotiate-exchange}(\text{question}, \text{partner}, \text{contract}), \text{register-contract}(\text{notary}, \text{contract}), \text{exchange-knowledge}(\text{partner}) \}$
Rights	access-repository
Norms	IF agreed-share(partner) THEN OBLIGED (seeker, publish-repository(answer))
Type	external

Role: Visitor	
Role id	visitor
Objectives	O ₁ = browse-repository
Sub-objectives	...
Rights	access-repository
Norms	FORBIDDEN(visitor, obtain-knowledge)
Type	external

Role: Editor	
Role id	editor
Objectives	o ₁ = validate-knowledge
Sub-objectives	$\Pi_{o_1} = \{ \text{receive}(\text{participant}, \text{knowledge-item}), \text{decide-value}(\text{participant}, \text{knowledge-item}) \}$
Rights	access-repository
Norms	IF received-item(owner, item) THEN OBLIGED(editor, verify-reputation(owner))
Type	external

Figure 7-5: Definitions of external roles and groups in the Knowledge Market

Facilitation roles are derived from the type of coordination, a network model in this case. In chapter 6, section 6.4.1.1.1. As discussed in section 7.2.2.1, besides the standard network facilitation roles gatekeeper, notary and monitor, the facilitation

layer of Knowledge Market also includes the role of matchmaker. In Figure 7-6, the facilitation roles for the Knowledge Market are depicted. Note the objectives of facilitation roles are mainly directed to handle requests from operational roles.

Role: Monitor	
Role id	monitor
Objectives	$o_1 = \text{apply-sanction}(p, \text{contract}, \text{sanction})$
Sub-objectives	
Rights	access-contract-information
Norms	IF breached-contract(p, contract) THEN OBLIGED(monitor, apply-sanction(p, contract, sanction))
Type	institutional

Role: Gatekeeper	
Role id	gatekeeper
Objectives	$o_1 = \text{handle}(\text{membership-application}(\text{applicant}, \text{decision}))$
Sub-objectives	$\Pi o_1 = \{ \text{ask-intentions}(\text{applicant}, \text{role}),$ describe-society, IF decide-acceptance(applicant, role, yes) THEN negotiate-social-contract(applicant, role, SC) }
Rights	decide-acceptance
Norms	OBLIGED(gatekeeper, inform(applicant, decide-acceptance(applicant, role, YN))).
Type	institutional

Role: Notary	
Role id	notary
Objectives	$o_1 = \text{handle}(\text{register-contract}(p_1, p_2, \text{clauses}))$
Sub-objectives	$\Pi o_1 = \{ \text{check-contract}(p_1, p_2, \text{clauses}),$ register-contract(contract(p1, p2, clauses)), appoint-monitor(monitor, contract) }
Rights	
Norms	IF requested(register-contract(p1, p2, clauses) THEN OBLIGED(notary, (register-contract(p1, p2, clauses) AND appoint-monitor(monitor, contract)))
Type	institutional

Role: Matchmaker	
Role id	matchmaker
Objectives	$o_1 = \text{handle}(\text{request-partner}(\text{participant}, \text{question}))$ $o_1 = \text{handle}(\text{register}(\text{participant}, \text{type}))$
Sub-objectives	$\Pi o_1 = \{ \text{find-potential-partners}(\text{question}, \text{members}, \text{potentials})$ $\forall p: \text{potentials distribute-request}(p, \text{question}, \text{YN}),$ answer-request(participant, partners) }
Rights	
Norms	IF requested(request-partner(participant, question) THEN OBLIGED(matchmaker, distribute-request) IF requested(register(p, type) THEN OBLIGED(matchmaker, verify-reputation(p))
Type	institutional

Figure 7-6: Definitions of facilitation roles in the Knowledge Market

Furthermore, eventual groups of roles are identified and group norms formalized in the group description, as is shown in Figure 7-7.

Group: Participant	
Roles	seeker, owner, visitor
Norms	

Group: Partner	
Roles	seeker, owner
Norms	IF reach-agreement THEN OBLIGED(partner, request-register-agreement(notary, agreement(partners, description)))

Group: Browser	
Roles	seeker, visitor
Norms	

Figure 7-7: Group specifications

Role dependencies and relations to external parties are depicted in Figure 7-8. The dependencies shown in this figure are related to the role definitions. Dependency relation diagrams, as the one depicted below, display dependencies as labeled arrows between two roles. The source role is the role where the objective is defined, the target role is the role that handles the objective, and the label indicates the objective.

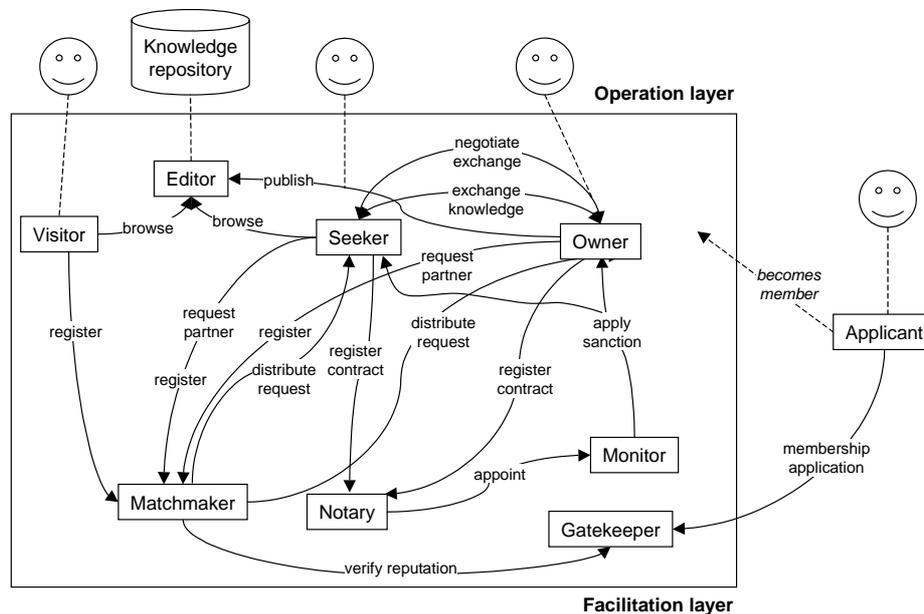


Figure 7-8: Role dependencies in the Knowledge Market

Dependencies in a network society are per default network dependencies. If a given dependency should be of another type, then this must be specified in the dependency definition. Network dependencies identify a authorization equivalence relation between the roles, that is, both are authorized to request the objective. For example, consider the objective *register* of role *visitor* below. A network relation between *visitor* and *matchmaker*, means that either the *visitor* can request the *matchmaker* to handle his registration, or the *matchmaker* can request the *visitor* to register.

7.2.2.3.2 Interaction Structure

In this section, we describe the interaction structure of the Knowledge Market. The methodology prescribes that scenes are specified for each role dependency identified in the social structure. For the Knowledge Market this means that a scene script is to be described for each of the labeled arrows depicted in Figure 7-8. Scenes are first described in informal terms in scene tables that are then translated into formal scene scripts. In Table 7-5, we list all scenes in the Knowledge Market, including their participating roles and the target scenes they are connected to. This table is not a complete scene table, but intended to give a summary of Knowledge Market scenes and their relationships.

Table 7-5: Scenes in the Knowledge Market

Scene Identifier	Roles	Connected to
Start	Gatekeeper, Applicant	Register
Register	Matchmaker, Participant	Verify Reputation Request Partner Publish Browse
Verify Reputation	Matchmaker, Gatekeeper	Register
Request Partner	Partner, Matchmaker	Distribute Request Negotiate Exchange Register End
Publish	Owner, Editor	End Exchange Knowledge
Browse	Browser, Editor	End
Distribute Request	Matchmaker, Partner	Request Partner
Negotiate Exchange	Partners	Register Contract Request Partner
Register Contract	Partner, Notary	Appoint Monitor Exchange Knowledge
Appoint Monitor	Notary, Monitor	Register Contract
Exchange Knowledge	Partners	Apply Sanction End
Apply Sanction	Monitor, Partner	Exchange Knowledge
End	Gatekeeper, Participant	- -

After having decided which scenes to specify for the Knowledge Market, the relationships between interaction scenes must be identified and formalized. For

example, the Exchange Negotiation scene must occur after a successful Partner Request scene, and is unique for each group of partners. The scenes for registration of partnerships and exchange are also unique for each partner group, and a new instance should be created each time. Since participants can choose whether to browse the repository, request a partner, or publish a source, the corresponding scenes are independent of each other and can occur in parallel. Visitors and Seekers are not allowed in publish scenes and visitors are also not allowed in partner request scenes, which indicates a transition norm on the admittance to those scenes. Figure 7-11 depicts the connections and transitions between scenes.

In the following, we provide the scene scripts for the scenes ‘Knowledge Request’, in Figure 7-9 and ‘Exchange Negotiation’, in Figure 7-10, which have been implemented in the prototype.

Interaction Scene: Partner Request	
Description	<i>Seeker requests possible partners that can answer knowledge need</i>
Roles	S: Knowledge-seeker(1), M: Matchmaker (1)
Results	DONE receive-partners(S, M, question, ListPartners)
Patterns	{ request-partner(S, M, question, deadline), distribute-request(M, knowledge-owners, answer-deadline) BEFORE request-deadline, request-deadline BEFORE answer-deadline, answer-deadline BEFORE deadline, receive-partners(S, M, question, List) BEFORE deadline, AND List = {P: DONE (answer-request(P, M, Yes, question) BEFORE answer-deadline)}
Norms	OBLIGED obtain-knowledge(M, knowledge-owners, answer-deadline) BEFORE deadline IF obtain-knowledge(matchmaker, P, question, deadline) THEN OBLIGED answer-request(P, M, YN, question) BEFORE deadline

Figure 7-9: Scene script for ‘Knowledge Request’

Interaction Scene: Negotiate Exchange	
Description	<i>Seeker requests possible partners for a knowledge need</i>
Roles	S: Seeker (≥1), Owner (≥1)
Results	DONE agreement(S, O, question, conditions)
Patterns	{ FORALL request-agree(S, O, condition): (answer(O, S, YN, condition) OR request-agree(O, S, condition)), FORALL request-agree(O, S, condition): (answer(S, O, YN, condition) OR request-agree(S, O, condition)), agreement(S, O, question, Conditions) AND Conditions = {C: answer(S, O, Yes, C)}
Norms	IF request-agree(P1, P2, C) THEN OBLIGED (answer(P2, P1, YN, C) OR request-agree(P2, P1, C'))

Figure 7-10: Scene script for ‘Exchange Negotiation’

The resulting interaction structure is displayed in Figure 7-11. Note that dashed arrows indicate an exclusive OR (only one of the paths can be followed). A detailed description of scene transitions can be found in chapter 3.

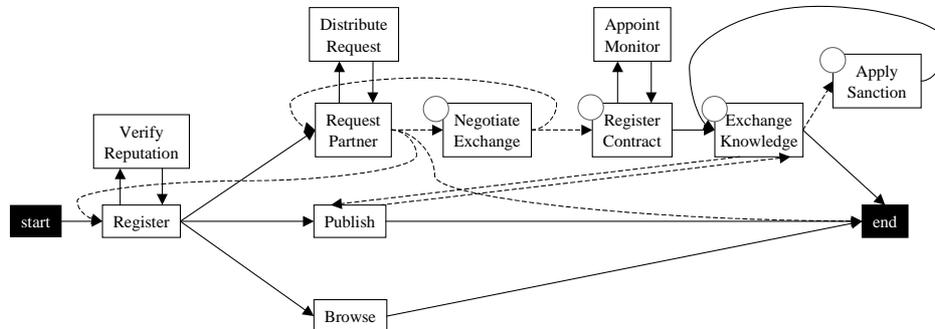


Figure 7-11: Interaction structure of Knowledge Market

7.2.2.4 Social Model

In the Social Model, the action of independent agents in the society is specified. Such agents seek to enact one of the operational roles in the society. In the Knowledge Market, agents enacting a facilitation role have capabilities and are controlled by the society. Therefore, external agents cannot apply to a facilitation role. This is not the case in a generic agent society, which allows for independence of facilitation roles. However, in most cases, society design will specify a number of institutional roles in order to keep control over the society in some way or another.

People seeking collaboration through the Knowledge Market will initiate a personal agent that acts as their avatar in the system. This agent uses the preferences and conditions specified by the user to find appropriate partners and negotiate exchange terms. Depending on the specific task, the personal agent will take either the role of *knowledge seeker* or *knowledge owner*. Requirements concerning privacy, secrecy and competitiveness between brands and departments that influence the channels and possibilities of sharing are also described in the specification of the personal assistants. Typically in the KennisNet, members do not have restrictions concerning sharing of knowledge they bring in. However, especially when new products are concerned, it can happen that agents of members involved will require such knowledge to be shared only within a restricted group.

Social contracts describe the agreements between participating agents and the Knowledge Market society. Negotiation of social contracts is done between the applicant agent and the Gatekeeper agent, which will watch over the interests of the society itself. For example, Anne is a member of the KennisNet group that is seeking knowledge on price policies from the competition. Anne will initiate an agent enacting the knowledge seeker role in the Knowledge Market. During the Start scene, the conditions for Anne's agent will be negotiated and fixed in a social contract that specifies, for instance, which parts of the repository Anne is allowed to access, which are the obligations of Anne concerning the publication of knowledge items received as result of an interaction, and whether Anne allows for items that she provides to be published or not. This negotiation process can be very simple, in which case, Anne is offered a specification of the Knowledge Seeker role and either she accepts it as it is

to be admitted or she refuses and admittance is denied. More sophisticated versions will require that agents are able to reason about goals, norms and objectives.

Social Contract	
Agent	Anne
Role	Knowledge seeker
Clauses	<ol style="list-style-type: none"> 1. PERMITTED(Anne, access-kb([KB1, KB3, KB7]) 2. OBLIGED(Anne, publish-received-knowledge(item, KB3) allows(KO, publish)) 3. $\forall p: \exists \text{contract}(p, \text{Anne}) \rightarrow \text{PERMITTED}(p, \text{publish}(p, \text{Anne's-item}, \text{kb}))$

Figure 7-12: Example of social contract

The example contract depicted in Figure 7-12, describes the social contract between agent Anne and the Knowledge Market society, by which Anne is given permission to access some of the knowledge bases in the repository, namely knowledge bases 1, 3 and 7. Anne is obliged to publish all received knowledge in knowledge base 3, given that publishing is allowed by the knowledge owner involved in that exchange, and Anne allows her knowledge (which she may need to release as counter activity in an interaction contract) to be published by her partners, in whichever knowledge base her partner can access.

7.2.2.5 Interaction Model

The following example describes a contract between two members. In this example, fictive but typically possible in the domain of non-life insurance, Anne will provide Bob with a report about competition prices, on the condition that Bob will give her comments on the report (that she will have to present to her Unit directors) and eventually share with her his new pricing concept for car insurance. This contract is generated during the ‘Negotiate partnership’ scene and registered in the ‘Register partnership’ scene. In this scene, the notary agent will assign a monitor agent to check the fulfillment of the contract between Anne and Bob. Monitoring can be a very simple activity, where status is checked when a deadline is reached. However, we have chosen to use an agent as monitor because monitors can take a more active role, reminding parties of approaching deadlines or by suggesting possible actions when sanctions occur. The clauses of this contract are informally specified in Figure 7-13.

Interaction Contract: ‘ID’	
Parties	Anne (A), Bob (B)
Clauses	<ol style="list-style-type: none"> 1. OBLIGED A DONE(A, receive(B, report-concurrent-prices) BEFORE <i>next-week</i> 2. IF received(B, report-concurrent-prices) THEN OBLIGED B (receive(A, comment-report-concurrent-prices) BEFORE <i>3-days</i> AND receive(A, concept-pricing) BEFORE <i>1-month</i>) 3. IF delayed(B, concept-pricing) THEN OBLIGED B inform(A, delayed(concept-pricing))

Figure 7-13: Example of interaction contract

In the case that either one of the agents will not fulfil its commitments, sanctions will be applied. When sanctions are not explicitly specified in the contract, the norms of the society will be used. For instance, the Knowledge Market follows the norm that

agents that do not fulfil their commitments are given less priority in exchanges. Also it is possible to consider the publication of a list of best and worst members.

7.2.3 Implementation of Knowledge Market

A prototype of the Knowledge Market was developed with the aim of checking the applicability of existing, freely available agent tools to the development of agent societies to support knowledge sharing [Khalil, 2002]. As result of this project, a knowledge exchange process between two agents and mediated by a matchmaker was implemented both in Jade⁴⁵ and Zeus⁴⁶. In these prototypes, agents exchange knowledge descriptions based on keywords. Instead of a full blown reciprocity mechanism as specified in the requirements of the Knowledge Market, we choose in the prototype for a currency-based exchange. That is, each agent receives an amount of points that can be used to ‘buy’ knowledge items and earn points by providing its knowledge to others. Furthermore, we developed a simple heuristic to determine similarity and relevance between knowledge items based on ontological proximity.

Matching knowledge supply and demand was one of the main challenges of the project. For example, if the seeker is looking for knowledge items on snow damage in motorcycles, and no exact match can be found would she rather get items on snow damage in cars, or generic motorcycle damage? The software system implemented a simple protocol for knowledge matching, based on ontological distances between concepts. More empirical research is needed in this area, in order to determine realistic requirements for knowledge matching.

7.3 Agent Societies in Non-Organizational Settings

The CareCircle case study aims at demonstrating the applicability of the OperA model to non-organizational settings, and exploit applications other than Knowledge Management. The idea is to support social interaction in communities by encouraging people to provide help to others within the community. In particular, we look at ways to organize the access to secondary (health) care services in a community. Such services often fall outside regular, institutional, services provided by health care organizations, or have long waiting lists, such as: getting medicines, help at home during recovery, transport to doctor/hospital, preparation of meals, babysitting (e.g. to make it possible for the mother to visit the doctor), etc. The proposition behind CareCircle is that electronic commerce practices and team formation can be used to support interactions, that have not traditionally been seen from the perspective of “trading” and in consequence for which there is no accepted valuation function or price discovery mechanism. The project is based on the LETS concept (Local

⁴⁵ <http://sharon.cselt.it/projects/jade>

⁴⁶ <http://www.labs.bt.com/projects/agents/zeus/index.htm>

Exchange Trading Systems) where goods and services are exchanged without recurring to the use of money [Lietaer, 2001], [Boyle, 1999].

The project explores how complementary currency trading can be used to tackle social issues arising from the changing demographic profile and the likely reduction in working hours stemming from technological change [Siebert, 2002]. In order for such schemas to be successful, a substantial part of the project effort must be directed to social aspects in order to generate interest, enthusiasm and support for the project in the community it is inserted. However, at this stage, we concentrate on describing the technological aspects of the support system. In order to support continuity of care and social integration in communities, the CareCircle system will function as follows:

- individuals may register as providers of a range of services. Their skills, availability and preferences must also be taken into account
- (other) individuals register their care needs and constraints
- care-points are earned for services provided
- care-points are used to pay for services needed by the participant⁴⁷, to obtain discounts on health insurance after so many points have been earned, or to pay for gifts as in 'air-miles' systems.

A supporting agent society manages the exchange between personal agents representing the members and their services or needs. The system provides registration, matching and conflict resolution capabilities. A point system (like air miles) is used as local currency to regulate exchange. Due to the characteristics of the domain, that is, mostly standard goods and short term relationships, the market coordination model is chosen for CareCircle. The main reason is that acceptance of the system will increase when interactions follow standard, fixed protocols, known and enforced to all participants, which leads to more trust in the functionality of the system.

Negotiation of exchange, in CareCircle involves multiple attributes (service, time, price, etc.). Both caretakers and caregivers can describe their own requirements and constraints concerning the type of service, and preferred type of partner (e.g. some people, for some care service, may prefer to be helped by a woman). The facilitation level of the agent society for CareCircle therefore contains the facilitation roles corresponding to the market structure: matchmaker, banker and market master. The society furthermore requires operation roles related to providers and receivers, and a consortium maker role whose objective is to facilitate the formation of complex interactions, requiring more than one providing partner. Figure 7-14 depicts the conceptual architecture of CareCircle.

⁴⁷ In some cases, when an individual has a clear need for services, expenditure may be incurred regardless of balance, so it is not necessary to have earned care points in order to be able to spend them.

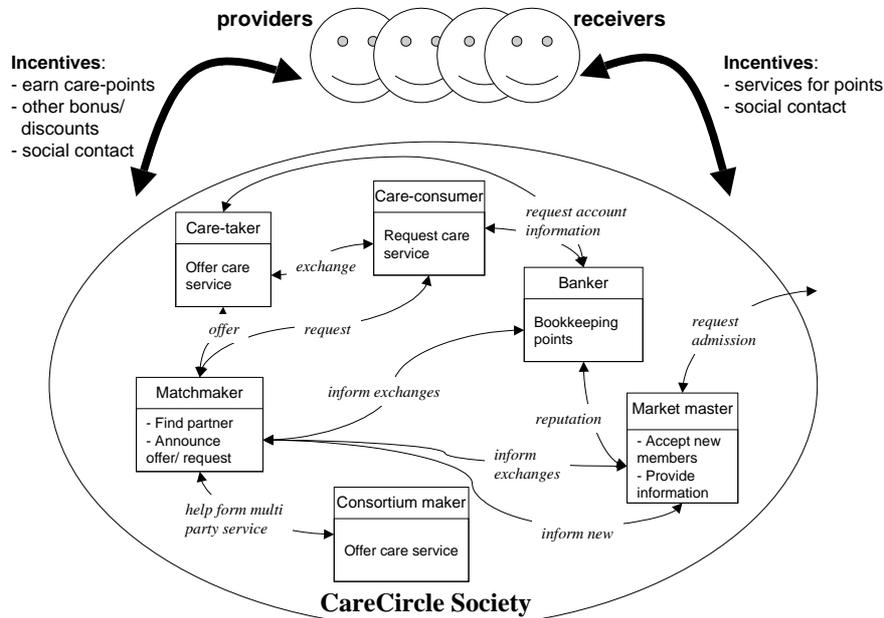


Figure 7-14: CareCircle Architecture

The purpose of the use of agent societies to support CareCircle is to enable the creation and organization of community groups, matching of supply and demand of care services, monitoring of exchanges and banking services for care-points. The facilitating institution acts as a mediator and animator for the caretakers, representing the members who bring various skills and services, and care-consumers, representing the members who bring their needs and requirements. Moreover, OperA enables the specification of open societies, in which heterogeneous agents can participate. In domains such as CareCircle, the functionality provided by an agent society includes:

- *accreditation of members*: to build confidence for consumers and caretakers as well as offering guarantees to the members.
- *profiling of members*: to help in matching caretakers' skills against consumer requirements.
- *mediation between members*: to establish the structure of consortia and the apportionment of rewards.
- *creation of consortia*: from simple partnerships for a few members, with a lifetime unlikely to exceed the contract, to fully-fledged separate institutions (spin-offs).
- *provision of value-sharing functions*: such as accounting and clearing house management, payment authorization and implementation of user-defined policies.
- *reputation model*: trust creation measures based on past performance and the societal relationships between agents which may serve to strengthen confidence.

As well as providing a basis for individual activity, the use of the agent societies enables the dynamic creation of teams to bring together skills and/or resources not available from just one person, but also raises issues of how to deliver this coordination unintrusively. Furthermore, interesting benefits arise from making the currency transferable between members of the wider trading community, following the model established in Japan by Tsutomu Hotta for the *hureai kippu* currency. Under this scheme activities such as the provision of child-care by one party in one place can be traded for care of the elderly by another party in another place [Litaer, 2002].

A pilot project will involve the development of a support system for the Care market, based on virtual enterprises and its evaluation through a simulation populated with intelligent agents. At a later stage, after the prototype is tested and enough experiment with the support system acquired, a real community pilot will be developed. The experience gained so far with the development of the conceptual model, indicates good suitability of OperA to model systems to support such interactions. The concept for this case study was developed as part of a project proposal for the Information Society Technologies area of the 5th framework program of the European Community. However, we did not develop the concept further, due to the fact that, on the one hand, even though the IST project was highly rated, it did not get funded, and, on the other hand, circumstances at Achmea and the current global economical setback made this kind of developments practically impossible.

7.4 Personal assistants for KM

Our last case study concerns the development of a personal assistant for call center agents. PAM⁴⁸ is a prototype of a system to support cross sell activities at the Customer Care Center of Achmea. PAM was one of our initial efforts to test the agent mediated knowledge management concept. At Achmea, call centers handle several million calls per year, dealing with marketing and sales, mutations of client portfolios and damage reports. Currently, call centers at Achmea are specialized in different products. Furthermore, lately, a large part of client contact has shifted from telephone to Internet. Integration of all client contact systems is the aim of the global Customer Care Center.

Call center agents have different backgrounds and expertise. On the one hand, inexperienced agents new to the work are able to handle only standard calls. Access to too many information sources proves often to be overwhelming to this group. On the other hand, expert call center agents are well knowledgeable with a specific business

⁴⁸ A couple of observations are due in relation to this project:

- In call centers, people working at the telephone desks are called '*agents*'. To avoid misunderstandings with the term agent in the software agent sense, we will in the remainder refer to them as '*call center agents*'
- The acronym PAM is derived from the Dutch: Persoonlijke Assistant voor de Medewerker (Personal Assistant for the Employee).

line (e.g. car insurance) and should be used to answer complex calls on their expertise area. It is therefore important that clients and their information requests are identified as early as possible in order to direct the call to an adequate call center agent. Furthermore, computer literacy also varies between different groups of call center agents, whose support requires a personalized and friendly interface. Furthermore, Achmea aims to extend its cross sell activities. This implies that call center agents must have enough knowledge of the complete range of products and services, and of the client's circumstances, in order to be able to suggest other products adequate for the client's situation and desires.

PAM aims to improve the way that information is gathered, managed, shared, and presented to call center agents in order to enable optimal service to the client and support cross-sell. The knowledge needed to efficiently and correctly handle client calls is embedded in a multitude of different sources, including people, telephone call handling systems, flexible scripting systems, databases, e-mail and internet. An ideal support system in this environment must meet the following requirements:

- Be able to handle structured and unstructured information, multimedia knowledge representations and links to people (filtering calls, e-mail, knowledge maps or yellow pages).
- Be easily accessible and provide an answer in a very short time (due to the fact that the client is waiting on the other side of the line).
- adaptability to the needs and preferences of users, and integration with existing work methods, tools, and interfaces.

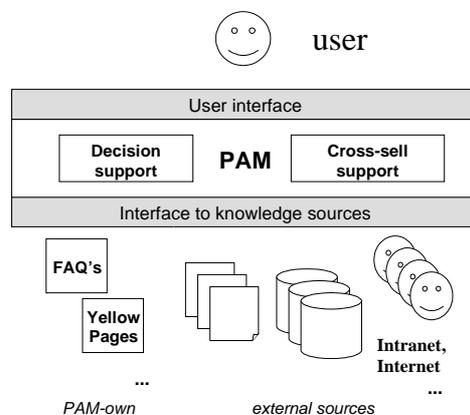


Figure 7-15: Architecture of PAM

In order to enable personalized support we chose to an implementation based on personal agents, which can be adapted to specific groups (novice, expert, manager, etc.). The overall architecture of PAM is described in Figure 7-15. In particular, PAM will:

- allow the call center agent to access relevant information wherever it is situated,

- proactively identify and timely deliver, relevant information which may not have been explicitly asked for (e.g. because the call center agent is unaware of its existence and/ or relevance for the situation on hand),
- inform the call center agent of changes made elsewhere in the business processes which have consequences upon the current decision context,
- identify and inform the parties who may be interested in the outcome of the current activities, and,
- provide performance indicators, e.g. contribution of current activity to team and individual targets.

The project resulted in the development of a paper prototype that was used for the formalization and analysis of the requirements for the implementation of the *e-call center* concept, which also included a uniform interface to all client and product information, online negotiation of schedules, possibility to combine telephone and internet interaction, and personalization of the electronic workspace. For the final implementation of the *e-call center*, the solution chosen was however not agent-based, due to the choice for a standard package from a preferred external supplier. The added-value of PAM lays in its integral approach, that incorporated the analysis of human and ICT aspects of call center interaction, the elicitation and formalization of use cases, and its conceptual description of the desired functionality.

7.5 Discussion

In this chapter, we have presented several applications of OperA in real-life settings. The first project, the Knowledge Market presented in section 7.2, demonstrates that OperA fulfills the specification requirements of collaboration management systems. Agent concepts hold great promise for responding to the new realities of knowledge and collaboration management. One of the main benefits of agent-based modeling of KM environments is that it provides a basis for the incorporation of individual initiative and collaboration into formal organizational processes. The preliminary results obtained from the conceptual modeling and the development of an agent-based prototype, using existing platforms are very promising. However, those results also indicate that dedicated tools for agent societies are needed, if practical applications are to be deployed. That is, future research in agent-oriented approaches to knowledge management and collaborative systems must include:

- Methodologies to support the analysis of knowledge management needs of organizations and its specification using software agents and agent societies
- Reusable agent-oriented knowledge management frameworks, including the description of agent roles, interaction forms and knowledge description
- Agent-based tools for organizational modeling and simulation that help determine the knowledge processes of the organization
- The role of learning in agent-based knowledge management systems, namely, how to use agent learning to support and extend knowledge sharing

In the specification of scenes and roles of the Knowledge Market, we have used a simple language to represent interaction between roles. The specification and

implementation of the Knowledge Market demonstrated that the specification of interaction requires formalisms that enable the representation of variables to unify actions of the different actors. This is also an important area for further research.

The second project, concerned the development of agent societies for non-organizational domains. The results obtained so far, confirm that the OperA model is rich enough to model other types of interactions, outside a structured organizational scope. On the social side of the project, and by this we mean the impact of the proposal on the human community where it will be inserted, results elsewhere are promising (cf. the *hureai kippu* concept in Japan). However, this area will require more research, involving different disciplines, in particular sociology, in order to study the effect of the community support proposed in the community and in the relations between people.

Although PAM, the project presented in section 7.4, does not relate directly to agent societies, we have included it here for two reasons:

1. We believe that the characteristics of the problem indicate a suitable area for the application of the OperA concept. In particular, adequate solutions will require adaptation to individual requirements and realization of global objectives.
2. The approach chosen to the development of PAM, is characteristic of current organizational choices for KM implementation. Even though innovative solutions are discussed and studied at prototype level, for the final implementation often traditional, off-the-shelf tools are chosen.

Finally, we conclude with an observation concerning the choice for practical implementations in real-life settings for the test and analysis of scientific research results. In our experience, this choice, which we have reported in this chapter, has on the one hand the great added value of getting direct first-hand experience on the applicability and usefulness of the decisions made. However, on the down side, real-life settings often bring with them all kinds of real-life problems and choices that, although not directly related to the research taking place, are nevertheless of utmost importance for the acceptance and eventual realization of the projects.

Chapter 8

Conclusions

'A whole is that which has beginning, middle and end'
– Aristotle (384 BC – 159 BC), *Rhetoric*

This dissertation focused on the design of models for organizations that support dynamic and autonomous interaction, and therefore can reflect changing requirements and objectives in an open environment. The motivation for such models came forth from the need to devise Knowledge Management solutions that incorporate the management of knowledge assets with environments that facilitate and encourage interaction between people in an open environment.

The OperA model devised in our research, uses the agent paradigm as conceptual design tool. At an abstract level, the concept of agents can refer to any autonomous entity participating in an interaction (including people). However, throughout the dissertation, we have used theoretical models of agents and agent interaction to build models for organizations. The main focus of OperA is not the design of the individual entities that form the organization, but the design of the environment where those entities interact, such that a balance can be found between the autonomy of agents, their coordination needs and the environment expectations.

This final chapter discusses the results from this project, in section 8.1, and points areas for future research, in section 8.2.

8.1 Discussion of Results

Three main assumptions underlie this research. Firstly, there is a general agreement that in most cases, autonomous entities need a social setting in order to realize their own individual goals. However, and this is our second assumption, organizations and/or societies have themselves global goals and requirements, which are not necessarily shared with any of the participating entities, but must be achieved by the (coordinated) activity of those individuals. Finally, we assume that a process of

negotiation and adjustment between individual and social requirements and characteristics is needed in order to conjugate individual autonomy with social requirements and goals. These assumptions were the basis to the research questions formulated in chapter 1, which resulted in a set of goals for this dissertation:

1. Demonstrate that organizations and other types of societies can be modeled in a way that integrates global aims and requirements with the autonomy of its participants.
2. Determine the applicability of the agent paradigm to model organizational systems in open environments.
3. Prove the applicability of such society models to support knowledge acquisition and sharing across heterogeneous sources in a virtual organization in order to support knowledge intensive tasks and processes.

In the remainder of this section, we look back at the realized work, and discuss the main findings related to these objectives.

8.1.1 Autonomy and Organizational Modeling

Our first research question relates to the need for the balance and integration of global aims with individual autonomy in organizational models. That is, can we model organizations such that organizational requirements and objectives are met, but participants have the freedom to act according to their own personalities, and the room to realize their own plans? In chapter 1, we have identified the main requirements that, in our opinion, must guide models for open, complex, dynamic organizations:

- independently from the internal design of the individual systems (**internal autonomy requirement**)
- without fixing the interaction structures completely in advance (**collaboration autonomy requirement**)

The OperA framework presented in chapter 3 was designed to fulfil both these requirements. By providing an explicit separation between the design of the organizational components (the roles) and the active entities that animate those components (the agents), we demonstrate the realization of the first autonomy requirement. The layered approach taken in OperA meets the second autonomy requirement, by enabling the description of increasingly more detailed specifications for interaction and enactment. In OperA refinement is more than an abstraction device: it is a matter of the proper balancing of responsibilities, and is fixed in contracts. Furthermore, OperA focuses on the organizational relations and requirements of multi-agent systems, and as such, takes a perspective on the system external to the agents themselves.

In order to realize the practical application of OperA, a methodology is needed that supports the development of models for open organizations. This methodology, presented in chapter 6, demonstrates that organizations can indeed be modeled in practice in a way that integrates global aims and requirements with the autonomy of its participants, and, as such, meeting the autonomy requirements above. The

methodology has been applied to the development of practical applications, as described in chapter 7.

8.1.2 Applicability of agent paradigm for open organizational models

Agents offer a way to deal with complex systems that have multiple and distinct components, and are often used as a metaphor for autonomous, intelligent entities [Luck et al., 2003]. An obvious and pragmatic approach to demonstrate the applicability of agents to model organizations would be simply to go ahead and do it. However, even though agents are adequate to represent rational entities, because of their autonomy, flexibility in content and participation, reactive and proactive behavior, communication and reasoning capabilities, when considering an organizational setting more is needed. In an organizational environment, the behavior of the global system and the collective aspects of the domain - such as stability over time, predictability and commitment to overall aims and strategies - must be considered. That is, the concept of desirable social behavior is of utmost importance when multi-agent systems are considered from an organizational point of view. This leads to a rising awareness that multi-agent systems and cyber-societies can best be understood and developed if they are inspired by human social phenomena [Artikis et al., 2001], [Castelfranchi, 2000], [Zambonelli et al., 2001a]. This is, in many ways, a novel concept within agent research, even if sociability has always been considered an important characteristic of agents.

At the beginning of the research presented in this dissertation, we tried to identify agent frameworks that met our objectives, to find out that there were no ready-to-use agent models which could be taken as a starting point. In chapter 2, we presented open societies and organizational models, and discussed current developments in agent research pointing out their characteristics and shortcomings as frameworks for organizational modeling. As there were no available means to use agents as a modeling tool for open organizations, our solution was to develop a model that met these requirements. The OperA framework was presented and extensively discussed in chapter 3. The answer of the OperA model to the autonomy requirements presented in chapter 1, is to separate the design of the organizational structure of a social system, from that of the participating agents, and to provide means to explicitly describe the agent's participation and activity in the society. Two sub research questions arise from this decision:

- How can efficient coordination between agents be achieved without compromising the agents' autonomy?
- How to define a contract language that will be used by agents to coordinate their activities within a society?

The approach taken to answer the first sub-research question, was to distinguish between roles and agents, and the specification of social contracts to described the common agreements concerning the activity of an agent while enactor of a role. Furthermore, the use of landmarks for the description of the organization aims concerning interaction, and the instantiation of these landmarks to the specific requirements of the agents involved, in terms of interaction contracts, also contributes

to the preservation of autonomy, while enabling coordination. This resulted in a layered format for OperA:

- **Organizational model (OM):** describes the desired or intended behavior and overall structure of the society from the perspective of the organization in terms of roles, interaction scripts and social norms.
- **Social model (SM):** populates the organizational model with specific agents mapped to roles through a social contract. Social contracts describe the agreed behavior for an agent within the society in terms of externally observable events. Agents enacting a role are called actors.
- **Interaction model (IM):** specifies interaction agreements between actors. Describes the actual behavior or the society.

The use of landmarks to specify organizational aims and requirements, both for roles as for interactions, provides a flexible way to represent autonomy – i.e. enabling the description of interactions ranging from very concrete, in fixed scenes with little agent autonomy, to little scene description and large agent autonomy – makes OperA suitable to model different types of organizations. By this we mean that OperA is suitable both to model and study pre-existing societies, as well as to develop new systems that will participate in a organizational context. Because it is based on an organizational, collectivist view of the environment and uses the agent paradigm to provide a natural way to view and characterize intelligent systems, we are certain that OperA can contribute to the acceptance and understanding of agent societies in organizations [Weiss, 1999]. Furthermore, OperA legitimates the concept of autonomy between society requirements and agent goals.

8.1.2.1 Formal models

Even though our primary aim was the development of practical solutions for concrete (Knowledge Management) applications, it was important to develop a formal theory for the OperA framework. The role of formal methods is to provide a clear and precise description of what a system is supposed to do, rather than a formulation of how it operates. The fact that OperA has a formal semantics, permits to give models a precise semantics, supports the use of structured design techniques and formal analysis, facilitating development, composition and reuse [Esteva et al., 2001], and can therefore be used to guide and support the designer while building and refining a conceptual model.

We have developed a language for contract representation, LCR, based on deontic temporal logic. LCR is a very expressive logic for describing interaction in multi-agent systems. This logic, described in chapter 4, makes it possible to describe and verify contracts that specify interaction between agents. In chapter 5, we extended LCR with illocution and epistemic elements, as a formal framework and integrated semantics for OperA, at all three levels of society specification (organizational, social and interaction). The formalism provides a rather realistic representation of a domain, in the sense that it treats temporal and communicative aspects and furthermore is able to represent deadlines and their influence on the behavior of the model.

8.1.2.2 Contract specification

Agreements between agents, and between an agent and the society, are specified by means of contracts. Contracts provide flexible but verifiable means to integrate society requirements and agent autonomy, and are an adequate means for the explicit specification of interactions [Verharen, 1997]. From the society perspective, it is important that these contracts adhere to the specifications described in the Organization Model. Therefore, the specification languages must be formalized syntactically and semantically so that formal verification is possible. The branching time deontic logic LCR, described in chapter 4, provides the semantics for contracts. Using the models presented in this paper, and the LCR logic, it becomes possible to give a precise and implementable specification of agent societies. In chapter 7, we have shown how concrete contracts can be specified, as in the examples in figures 7-5 and 7-6.

8.1.3 Development of KM support systems

Our original concern was the investigation of KM environments and the modeling of systems to support effective collaboration and knowledge exchange across physical, temporal and organizational boundaries. Our third research question, *'can such a society model be used to support knowledge acquisition and sharing across heterogeneous sources in a virtual organization in order to support knowledge intensive tasks and processes?'*, could therefore have been the first one. This is the reason why the main practical application described in chapter 7, the Knowledge Market, comes from the Knowledge Management area.

The development of an agent based system for knowledge sharing based on the OperA framework made possible the support of direct interactions between network members, and thus share knowledge not (yet) published in the repository. The model ensures the preservation of individual needs and perspectives, as each user is represented by its own agent, and on the other hand, functions according to the structure designed to meet the requirements of the organization. Furthermore, agents are employed to monitor and assist on the exchange in a way that benefits the organization as a whole (e.g. agents can, on behalf of their users, publish results of a specific exchange in the repository).

Other recent results in the area of Agent-Mediated Knowledge Management also show that agent concepts can fundamentally alter the nature of KM both in the way KM systems are built as well as the way organizations are analyzed and modeled [van Elst et al., 2003b]. On the one hand, the technical embodiment of these concepts can lead to advanced functionality of KM systems, e.g. personalization of knowledge presentation and matching supply and demand of knowledge. On the other hand, the rich representational capabilities of agents as modeling entities allow faithful and effective treatment of complex organizational processes. In our opinion, one of the main contributions of agent-based modeling of KM environments is that it provides a basis for the incorporation of individual initiative and collaboration into formal organizational processes.

8.1.4 Concluding Remarks

We finalize this section with a short overview on how the work presented in this dissertation answers the research questions. We have identified two major requirements for models of organizations that integrate global requirements and autonomous participants. These autonomy requirements guided the development of the OperA framework. We have enhanced the practical applicability of OperA by developing an engineering methodology that guides the design of OperA models. Moreover, the formal semantics of OperA provide a strong theoretical background and enables the verification of OperA models. This demonstrates that organizations can indeed be modeled in a way that combines global requirements and autonomy, which answers the first research question.

OperA is an agent-based framework able to accommodate different agent architectures, and it provides a flexible way to represent interaction and role enactment. It is therefore suitable to model open organizational systems and as such answers the second research question.

Finally, the third research question was approached by the development of practical case studies, namely the Knowledge Market project described in chapter 7. Due to organizational constraints and setbacks, this development stayed mainly at conceptual level. It remains therefore to be seen whether a support system based on OperA using agents will really work in practice and contribute to actual exchanges between human users. The research opened several areas for further research. In the next section we will discuss some of these new research questions and point directions for further work.

8.2 Further work

Our research for future research concerns implementation, theoretical and practical issues, further detailed in the remainder of this section.

8.2.1 Implementation

The main direction for future research is obviously the development of practical tools to build OperA models. Tools to build agent organizations will ease the design, implementation and verification of multi-agent systems. Such tools should be able to guide the engineering process, by following the development methodology, and enable the specification, and the automatic configuration of agent societies according to the OperA model. Such tools should as well support the verification of OperA models.

In the follow-up of this dissertation, we plan to develop a computational system to specify an OperA model and automatically generate a multi-agent system that implements that model. The resulting multi-agent system should include the functionality of the OM, institutional agents to enact the facilitation roles, and the capabilities to enable the incorporation of external agents that will enact operational roles. The tool should furthermore provide software mechanisms for security and robustness to enable building real-world applications, beyond pilot implementations.

8.2.2 Theory

The further development of a theory of agent-based models for open organizations, will require further research in a large number of areas, under which we emphasize a few, described in the remainder of this subsection.

8.2.2.1 Integration of agent models

The formal language presented in chapter 5 is meant to specify the elements of an OperA society as introduced in chapter 3 and does not include the specification of the players (i.e. the agents themselves) in a society. As it is now, the OperA formalism does not result in an implemented system but in a conceptual model. This means that typical system properties such as liveness cannot be verified directly, system traces cannot be generated. An extension to the formal language to support the specification of the internal aspects of agents would enable the complete specification of an animated society.

In our opinion, the development of open multi-agent systems will increasingly take place in ways such as the one proposed by OperA. That is, separating the specification of social issues from design of the individual agents, as stated in the first autonomy requirement. Therefore, a novel area of research is the description of formal languages to describe the integration of a formal social model, such as OperA, with formal agent architectures. Such language will enable to determine the exact semantic relations and system properties of open agent societies.

8.2.2.2 Coordination issues

Coordination is commonly defined as the management of dependencies among activities. The problem of coordination is then the problem of exerting control over these dependencies. Coordination is needed in cases where there is not full cooperation among the agents or groups of agents. Representational techniques of social structures will enable various investigations of computational complexity (for example, the trade-offs between time complexity and space complexity), scaling issues of large agent societies, treating emergent phenomena as a means of effective control, and a unified view of DAI methods and concepts [Findler, Malyankar, 2000].

In chapter 3, and later in chapter 6, we described frameworks for agent societies that model different types of coordination in organizations, namely markets, networks and hierarchies. However, other types can be envisioned and defined, and extensive libraries of components must be developed. Further research in coordination is needed to develop formal methods for coordination in agent societies, that describe how general models of coordination characterize relevant aspects of agent behavior in an agent society.

8.2.2.3 Delegation and responsibility

Delegation is an important issue for agents that may be forced to rely on others. To delegate is to entrust a representative to act on one's behalf. In organizations, agents,

as people, are often required to rely on others in order to achieve a goal, or delegation may make it easier for them to achieve the goal. Successful delegation implies that responsibility for the task must be shared. Through role dependencies relations, OperA provides means, even though at a simplified level, to represent delegation. However, more research must be done in order to [Norman, Reed, 2002]:

- Understand the nature of relationships upon which the ability to delegate is predicated [Castelfranchi, Falcone, 1998].
- Characterize imperative communicative acts through which an agent may delegate tasks to other individuals or groups of agents.
- Develop a formal model of delegation.

The above considerations, relate mostly to the delegation of tasks or activities. Another important area for further research concerns the delegation of responsibilities.

8.2.2.4 Ownership

OperA does not represent explicitly the notion of agent ownership, even though the issue is related to the stakeholders identified in chapter 6. In fact, each agent in a society represents or is owned by either another agent (human or artificial) or an institution. The notion of ownership is of utmost importance when determining the legal obligations associated with transactions in the 'real world', i.e. the liability and responsibility of agents' actions [Artikis, Pitt, 2001]. Agent ownership is a complex issue that crosses legal domains. There are at least two different dimensions to agent ownership [Yip, Cunningham, 2002]:

- An agent is a piece of software and as such it is an intellectual property which belongs to its owner and is governed by the law on intellectual property.
- A software agent can act like a business agent. There is a different set of laws governing the legal relationship in which a human business agent deals with a third party on behalf of some principal.

Both views raise several issues for further research. On the one hand, one must be able to answer questions such as: 'Who is the owner of an agent?', 'What is it that is owned?', 'How can an agent's ownership be transferred?', 'How can an agent owner protect his/her property?', or 'What is the legal relationship between an agent, an agent user and an agent developer?'. Another important issue addressed by the law of agency is the contract formation between agents, such as: 'What is the legal standing of a contract formed by software agents?'.

8.2.2.5 Normative systems

A normative system is defined as any set of interacting agents whose behavior can usefully be regarded as norm directed [Jones, Sergot, 1993]. Most organizations, and more specifically institutions, fall under this definition. Norms that guide organizations are usually described in regulations, with a high level of abstraction. They are established to regulate interactions between parties, and are meant to inspire trust in their participants. The level of abstraction of regulations is kept high on

purpose, as regulations should be stable for many situations and for a relatively long time. This means that norms do not provide concrete handles for their implementation [Dignum, 2002a].

One could regard OperA models as normative systems. However, in our present research we have not explicitly considered the study of norms, more than the elicitation and specification of the normative structure of an OperA model. Mechanisms are needed to explicitly specify how norms expressed in abstract regulations are translated into concrete norms for the specific organization being modeled. That is, the main question for further research is how to formally describe the relation between the abstract norms aimed by an organization (such as, *fairness* of transactions in a knowledge exchange), the concrete norms elicited in the Norm Analysis process described in chapter 6, and the final implementation of norms in the actual system, which can have been furthermore subject to a process of negotiation between agents and society, as part of the negotiation of social and/or interaction contracts.

A recent proposal in this direction, the HARMONIA framework, proposes a multi-level structure for the specification of normative systems [Vázquez-Salceda, Dignum, 2003]. In HARMONIA abstract norms and statutes of an institution are progressively concretized into concrete norms and policies, then linked with procedures that enable their realization for the specific normative system, and finally translated into computationally efficient implementations to be used by the agents. The way HARMONIA deals with norm specification has several similarities with the layered way OperA specifies organizational interaction. We plan in the near future to investigate the integration of both frameworks, which we feel will greatly enhance their applicability.

8.2.3 Practice

More work is also needed in the area of practical implementations of systems modeled in OperA. Current efforts concern the prototypical implementation of parts of the Knowledge Market system. Assuming that OperA based tools for the building of multi-agent systems are available, the implementation of Knowledge Market must be extended in at least two directions:

- Robust implementation of the complete system. We envision that this process can be incremental in the sense that a first implementation will be based on homogenous agents (following the current prototype) and be extended with the application of heterogeneous agents (built using different tools and architectures, possibly 3APL and Jade).
- Evaluation of user interaction with the system in a lab environment, as well as in a real environment if possible. This in order to determine the relative contribution of an agent-based approach compared with traditional means of knowledge exchange, both at the level of the individual users and at the level of the organization.

8.2.3.1 Applying OperA to other areas

OperA is a generic model for the design of multi-agent systems, with a formal semantics (cf. chapter 5) and a customized development methodology (cf. chapter 6). Even though the original aim of OperA was the development of KM support systems, the possibilities of the framework make it suitable for application in other areas. In chapter 7 we have reported initial research concerning the area of community support (cf. the CareCircle project, in section 7.3). We are sure that the framework is applicable to the development of other types of e-organizations and future work must be direct to the development of OperA to the modeling of other systems such as e-markets, institutions and virtual enterprises.

8.2.3.2 Linking to the environment

Another area for further practical research concerns the integration of OperA systems in the environment they belong to, and how to support that environment in the interaction with the system. We have already initiated some work in this direction, namely the support of the development of Communities of Practice reported in the appendix to this dissertation. Besides such empirical developments, we think that more research is needed at the theoretical level. That is, can formal models be developed to describe and analyze the relationship between a system and the environment it is inserted in? We plan to research the applicability of the concept of **linking pin** developed by Rensis Likert in management theory [Likert, 1961] to the problem of interaction between the society and its environment.

The environment of a society can be seen as consisting essentially of other societies. The problem of interaction between the society and the environment is then equivalent to the problem of relating different societies. Assuming that every role is related to a (human) *subject*, its stakeholder, different agent societies will be linked by roles referring to the same subject. Likert realized that managers in an organization are members of at least two groups and their behavior reflects the values, norms, and objects of both groups - a manager is a subordinate in one group and a superior in another group. So Likert puts the manager in the intersection of two groups, and because of this dual membership, he can forward information or control from one group to the other. Groups may have different norms, which leave the manager with the task to 'translate' between them. A Likert-model of an organization is pictured as a set of tiles, where each tile has one or more overlaps with other tiles. Moreover, not only managers can be linking pins and the set of tiles is not necessarily hierarchically ordered. We think that the way in which a linking pin connects two societies is also related to the coordination structure. However, more empirical and conceptual research will be needed in order to formulate specific solution for this issue.

Appendix A

Development of Communities of Practice

In this appendix, we describe initial research taking place at Achmea, concerning the development of means to facilitate the creation and maintenance of Communities of Practice in a structured, measurable and reusable fashion. As we have discussed before, it is important that the development and design of organizational software systems takes into account the specific characteristics of the communities that are going to use them and will only succeed if community members are convinced of the system's benefits for themselves and for the organization. The SES method reported in this appendix, facilitates the creation and management of communities and enables the concerted introduction of software support systems. This work has been previously published in [Dignum, van Eeden, 2003].

A.1 Introduction

Communities of Practice (CoP) are groups of people who come together to share and to learn from one another, face-to-face and virtually. They are held together by a common interest in a certain area, and are driven by the desire and need to share problems, achievements, insights, tools and best practices. CoP members deepen their knowledge by interacting on an ongoing basis, and will, over time, develop a set of shared practices [Wenger et al., 2002]. Although CoPs are not new and have existed in a variety of forms probably since mankind is around, only recently organizations have discovered the potential of CoPs to the sustainable advantage of business practice and the realization of strategic objectives and improvement of organizational performance.

As organizations grow in size, geographical scope, and complexity, it is increasingly apparent that sponsorship and support of communities of practice can improve organizational performance [Lesser, Storck, 2001]. Hence the arising of

sponsored CoPs, which are initiated, chartered and supported by company leaders, and which are set measurable targets that benefit the company. The mission and result of CoPs differ between communities and depend on the issues or practice area around which the CoP is centered, but the following aspects are usually encompassed:

- Stimulate interaction
- Decrease the learning curve of new employees
- Provide a forum for members to help each other solve everyday problems
- Development and dissemination of best practices, guidelines and procedures
- Respond more rapidly to customer needs and inquiries
- Reduce rework and prevent “reinvention of the wheel”
- Create new ideas for products and services

CoPs start from different premises and their aims and results are manifold. However it is commonly agreed that their advantages include the potential to overcome the inherent problems of a slow-moving traditional hierarchy in a fast-moving virtual economy, the ability to handle unstructured problems and to share knowledge outside of the traditional structural boundaries, and that CoPs provide adequate means of developing and maintaining long-term organizational memories. However, problems can arise due to the voluntary participation of their members, and CoPs are not always targeted to the collection and transfer of knowledge. This means that an approach to the creation and management of CoPs must combine the positive knowledge sharing capabilities of CoPs with manageable task solving capabilities and an orientation to business processes.

In this chapter, we describe why CoPs are important to Achmea and how Achmea is nurturing and organizing the deployment of CoPs within the organization. The chapter is organized as follows: In section A.2, we will introduce the current status of CoP developments at Achmea. In section A.3, the SES model used and developed at Achmea to create CoPs is introduced. Finally, in section A.4, we present our conclusions and point to areas for further research.

A.2 Communities of Practice at Achmea

Achmea realizes that a flexible, innovative and personal response to the requirements of customers is in great demand in financial services, insurance, security and health care. The realization of the central theme of the mission of Achmea, ‘**Achmea unburdens**’ has large consequences for the structure and processes of the organization. On the one hand, it requires the harmonization of processes across the organization, and on the other hand, Achmea needs to achieve larger synergy between people across the different business units. Furthermore, Achmea aims at a position of sustainable adaptability and advantage in its environment, which requires an innovative and flexible approach to customers and their needs and plans, and therefore a better management of knowledge and expertise in the organization [Drucker, 1995].

However, the current organizational structure, based on business unit independence, is not always conducive for the realization of synergy. Lessons learned

from several pilot projects in the past, concerning the analyze of CoP effectiveness, were leading to shape the activities around KM at Achmea. These projects confirm that both motivation of the management and the employees, as well as the characteristics of the collaboration, communication and information infrastructure are crucial for the success of CoPs [Burlage, et al., 1998]. An environment that encourages and facilitates the development of networks of people, is one of the ways to achieve the objectives above [Davenport, Prusak, 1998]. In our opinion, communities must be developed in a participatory way, involving members and stakeholders. On the other hand, communities should contribute to the realization of the strategic priorities of the company, what leads to a need for the explicit specification and monitoring of targets and objectives of a CoP. Even though CoPs are of great value for Achmea, and Achmea is very keen on the creation of communities, experience shows that forcing top-down the creation of communities does not work if the target group does not already have any common interests, activities and objectives [Gongla, Rizzuto, 2001]. The bottom line is that support of communities must focus on building social capital (including trust, norms, reciprocity, identity) as this provides a continuous basis for sustainable advantage and innovation.

CoP literature distinguishes between two types of communities: **self-organized** and **sponsored** (cf. [Nickols, 2000]). Self-organizing CoPs are created bottom-up, by the members, as a way to watch over and organize their own interests. Owing to their voluntary nature, they are fragile (as control attempts can result in their disappearance), but extremely adaptable and evolving according to members' interests. Sponsored communities are supported and initiated by the management, and are expected to produce measurable results that benefit the company. The internal structure of a sponsored CoP must however be decided by the members. An important aspect to keep in mind when dealing with sponsored communities is that *"a CoP reflects the members' understanding of what is important. ...Even when a community's action conforms to an external mandate, it is the community – and not the mandate – that produces the practice"* [Wenger, 1998].

In order to combine the advantages of both types, Achmea identifies groups - active in areas essential for the core business - that exhibit some of the characteristics of self-organization and then actively sponsors their activities. The process of matching community goals and interests to organizational strategic aims can only succeed if community members are convinced of the possible targets and organizational benefits of community activities and if they are active and explicitly involved in the shaping up of the CoP. When self-organizing communities become sponsored communities with clear targets and outward-directed activities, the need arises for the CoP to develop a clear profile of themselves as group.

The development of an own identity is an important means of connection within a group and provides an adequate interface for communication with the outside world. An important activity in the process of facilitating CoPs at Achmea, is therefore the development of a community identity. Such identify must on the one hand, be supported and shared by the members and, on the other hand, must be conform the organization's own identity and meet corporate requirements concerning look-and-feel and style.

A.3 The SES Model for community facilitation

The above observations make clear that CoPs are of great importance for Achmea due to their potential to contribute to synergy across business units and thus, to the realization of the strategic priorities of the organization. Several CoPs have been initiated or are currently under development. These projects confirm that motivation of the management and the employees, as well as the choice of infrastructure for collaboration, communication and information are crucial for the success of CoPs.

The **SES (Seduce, Engage, Support) Model** was developed at Achmea to facilitate CoPs across the organization structure. The aim of SES is to combine lessons learned, success stories and collective experiences, skills and tools from previous projects, in a way that is easily identified and understood by the organization. In this section the SES model is first introduced in a generic way, after which each of its components is explained in more detail. One main contribution of the method is its simplicity and adaptability to the needs of different groups. The SES model identifies four groups of actors involved in the activity and development of a community:

- **Initiators:** the individuals who realize that the organization can profit from the nurturing and encouragement of such a group and take lead in the creation of the CoP.
- **Members:** The persons that participate in the CoP, and whose mutual concerns, interests and activities form the body of the community [Talbot, 1995]
- **Stakeholders:** the group who can affect or be affected by the results and policies of the CoP [Vidgen, 1997].
- **Organization:** The corporate context in which the CoP is inserted.

SES is a participatory method and borrows ideas from community-centered development [Preece, 2000] in the sense that the characteristics and needs of the community members are leading and prior to any decisions concerning technology and social structure.

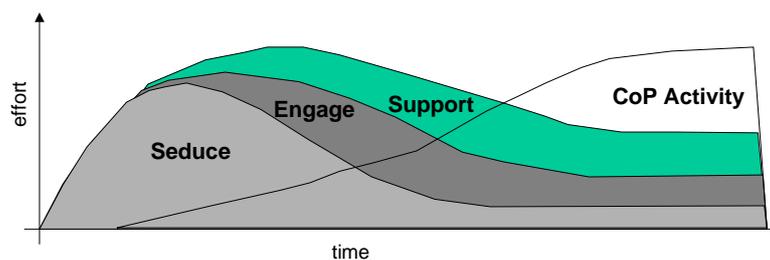


Figure A-1: The activity phases of the SES model

After the initial awareness of the potential of a group, the development of CoPs evolves along three main phases. During the first phase, **seduction**, the context and aims of a CoP are identified and described, potential members are made aware of their connections and common interests or objectives, and a ‘marketing campaign’ is

started directed at the organization, which shows the added values and benefits of the CoP for the whole. In the second phase, **engagement**, both community members and organization are involved in the process of setting up the CoP. The aim is to design a community that is as closely related as possible to the requirements and wishes of the members and whose tasks and targets are well embedded in the strategic priorities of the organization. The aim of the third phase, **support**, is to consolidate the CoP, by developing CoP-specific methods and tools for the organization, management and innovation of CoP activities. Figure A-1 gives an overview of the relative effort placed on each development phase throughout the life of a CoP.

A.3.1 Seduce

The process of creating a CoP usually starts with the identification of similarities, of business processes (in the case of an activity-based CoP) or background (for professional CoPs) in a group of people. A group of stakeholders, which may or may not be potential members of the CoP, realize that the organization can profit from the nurturing and encouragement of such a group and take lead on the creation of the CoP, as their activities are essential for the achievement of strategic priorities of the organization. Often, but not always, members of the group already have some form of contact with each others, mostly at local level, but in some cases also across locations and business units. Such contacts are however mostly informal and not supported by the organization. The formation of a CoP will thus enable that the knowledge and results of the group are optimally managed and distributed through the organization.

The aim of the Seduce phase is to create a feeling of anticipation about the CoP on both its potential members and the organization as a whole. This phase also aims at the clarification of the context and objectives for the CoP. One of the first activities in the Seduce phase must therefore be the identification of the target groups: initiators, stakeholders, members, organization. The activities of the Seduce phase are described in Table A-1.

Table A-1: Activities during the Seduce phase of CoP development

Target group	Activities	Aims	Tools
Stakeholders	<ul style="list-style-type: none"> - Interviews - Organize group discussions 	<ul style="list-style-type: none"> - Clarify CoP purpose - Define CoP targets - Make context explicit 	ACCELERATION ROOM Questionnaires
Members	<ul style="list-style-type: none"> - Organize meetings - Form pilot group 	<ul style="list-style-type: none"> - Introduce CoP concept - Increase awareness - Develop trust - Identify common interests 	ACCELERATION ROOM Mood boards
Organization	<ul style="list-style-type: none"> - Identify possible members - Publicity campaign 	<ul style="list-style-type: none"> - Show added value of CoP - Attract new members - Secure support 	Intranet Newsletters

Although the order of activities is not fixed, and indeed Seduce activities will continue well into the CoP lifecycle in order to keep the excitement and involvement of members and stakeholders alive, it is advisable to start this phase with activities directed to the stakeholders. In this way, their objectives, concerns and targets for the

community which form the basis for the participation of members, are identified and agreed upon at an early stage. However, in the development of any community, the members and their view on the organization and content of the CoP are central. The process of re-accessing and consolidating CoP aims and activities is dynamic and must be a constant part of the community lifecycle.

In later stages of development of a CoP, seduce activities are geared at the publicity and distribution of results and actions, so that the support for the groups is maintained and funding secured. The generation of quick wins at an early development stage is therefore crucial. Successes of one CoP are a great booster for other CoPs and for groups which are considering the creation of CoPs.

A.3.2 Engage

This step of development of CoPs is geared to the involvement of all target groups in the further shaping up of the CoP. The aims of the Engage phase for each of the target groups are summarized in Table A-2.

Table A-2: Activities during the Engage phase of CoP development

Target group	Activities	Aims
Members	<ul style="list-style-type: none"> - Interviews of pilot group - Group synopsis sessions - Identification of requirements and functionality 	<ul style="list-style-type: none"> - insure involvement - enable group's identification - link CoP to member objectives - Identify functionality requirements
Stakeholders	<ul style="list-style-type: none"> - Group synopsis sessions - Group discussions 	<ul style="list-style-type: none"> - Agree on targets and processes - Appoint champion - Identify functionality requirements
Decision-makers	<ul style="list-style-type: none"> - Make CoP objectives explicit - Identify strategic priorities 	<ul style="list-style-type: none"> - Match CoP aims to strategic priorities

Obviously, the engagement of participants is of utmost importance for the development of the CoP. Decision-makers and stakeholders may have strong rationale for the CoP and employ all kinds of efforts to facilitate it, at the end of the day, if participants are not willing, a CoP will never come from the ground. In our experience, identity is the key issue for individual participation in a group. This means, both the development of a group identity, as well as the assurance that individual objectives and concerns are incorporated in the objectives of the CoP. People need to get a clear, positive answer to the question '*what's in it for me*' in order to adopt the CoP as their own. Therefore the assurance that individual objectives and concerns are incorporated in the objectives of the CoP must be part of the development of the CoP, which is achieved using group synopsis methods. The focus of the Seduce phase towards CoP members is to involve them from the very beginning of the development of the CoP and make sure that personal requirements and desires are incorporated in the functionality and targets of the CoP. Towards the engagement of stakeholders and organizational decision-makers, activities are twofold: (1) appointing a champion will assure the bridge between the CoP and the organization, and (2), defining a clear and explicit link between the targets of the CoP

and the strategic objectives of the organization helps clarify the benefits of the CoP towards decision-makers, and insures their support.

The engagement process is a continuous one, to be kept throughout the lifecycle of the community. An important issue is how knowledge sharing evolves within CoPs, through the continuous motivation of people, the creation of trust and overcoming individual objections [Gurteen, 1999]. Furthermore, measurable characteristics of the CoP and reward/sanction systems for participation are specified during the engage phase.

A.3.3 Support

Once a CoP has been identified, and participants and stakeholders are engaged in its formation, active support for the CoP is fundamental for its success. By support we mean the facilitation of community activities in terms of infrastructure, funding, social structure and monitoring.

Infrastructure. Infrastructure support includes the creation and facilitation of time and space for CoP activities, that is, that members are enabled to participate in community life and meet others; and the availability of a technical infrastructure to back up CoP targets and activities. The existence of adequate information and collaboration support systems is an important aspect, but not a self-sufficient one. A often heard complaint of managers in organizations is that lots of money and effort is spent in a state-of-the-art ICT system that is subsequently not or hardly used. In our opinion, such mistakes are due to the idea that the system will make the community. In our approach, we concentrate on the formation of the community, and are only starting the development of ICT after its requirements and functionality are agreed upon and shared by the group which is going to use it. Furthermore, systems to support a CoP must fit with the specific characteristics of communities. The OperA model described in this dissertation, meets these requirements and is well suited to design systems to support communities.

Funding. Like most things in life, a well functioning community will need funds to back up its activities. Crucial items are meetings, training and the implementation of a sound infrastructure. In order to secure funding, the activities related to the engagement of stakeholders and organization as described above are of great importance. The community itself contributes to the continuation of this engagement both by being accountable and reporting on the usage of funds, as well as by maintaining an explicit link of its activity to strategy and goals of organization. When the targets and expectations on a CoP are well set and realistic, their achievement will have positive consequences to company revenues and therefore justify funding of the CoP.

Social structure. The activity and achievements of CoPs are much influenced by their social organization. Therefore, the design of a social structure for the CoP must go beyond the identification of members, and their requirements and preferences, but include the specification of the social structures (roles and communal behavior norms) and interactions between actors [de Moor, 1999]. The following are roles usually present in CoPs:

- leader/facilitator: who ensures the livelihood and effectiveness of CoP
- content manager: who updates and organizes CoP knowledge
- web manager: who is responsible for the maintenance of virtual space
- events coordinator: who coordinates face-to-face meetings
- trainer: who is responsible for community education and skill development

The identification and manning of social roles results often in the success or failure of a community. This is especially so for the role of CoP leader or facilitator, and special care must be taken on his/her appointment.

Monitoring. Finally, the support of a CoP should include the specification and realization of processes that monitor the activity of the CoP. Monitoring activities check how well the CoP is meeting its targets, whether members are satisfied with CoP activities, and provides a way for change and continuous adaptation to a changing environment. Monitoring will as well ensure the engagement of stakeholders and corporate management. Monitoring objectives and tools are community specific and should be agreed upon by its members.

The development of means, such as CoPs, to manage and nurture collaboration between people calls for approaches that are reactive and proactive in relation to the needs and expectations of the community members. Furthermore, due to the distributed nature of many communities active and envisioned at Achmea, community support must rely for a large part on virtual, internet based, systems. Nurturing communities is hard enough when the members are in a single location with good connectivity and increase considerably when the members are spread around different locations, possibly in different areas and with different languages and cultures. Members of distributed communities are not always aware of each other's capabilities and often they will discuss their business problems with a direct colleague just because he/she happens to be conveniently close and not because he/she is the best person to consult with. Links between members of distributed CoPs can be strengthened by tools for interaction support. There is therefore a need for collaboration management systems that meet the following requirements [Dignum, Dignum, 2003]:

- Methodologies to support the analysis of knowledge management needs of organizations and its specification using software agents and agent societies
- Reusable agent-oriented knowledge management frameworks, including the description of agent roles, interaction forms and knowledge description
- Agent-based tools for organizational modeling and simulation that help determine the knowledge processes of the organization

Experience with the KennisNet community, and in particular with the Knowledge Market system presented in chapter 7, indicates that OperA fulfills the specification requirements of collaboration management systems. Agent concepts hold great promise for responding to the new realities of knowledge and collaboration management. One of the main benefits of agent-based modeling of KM environments is that it provides a basis for the incorporation of individual initiative and collaboration into formal organizational processes. The preliminary results obtained

from the conceptual modeling and the development of an agent-based prototype, using existing platforms are very promising. However, those results also indicate that dedicated tools for agent societies are needed, if practical applications are to be deployed.

A.4 Conclusions

Most KM efforts attempt to document and share information, ideas and insights so that they can be managed, organized and shared. However, methods to leverage tacit knowledge often do more harm than good [McDermott, 2000]. CoPs are an answer to this problem by providing the means for people to interact and to access each other's expertise and insights directly. In this appendix, we presented ongoing work on the development of sponsored communities of practice at Achmea. The development process is based on the empowerment of communities, and stresses the role of the participants. Our work resulted in the development of the SES method which is also presented in this paper. The SES model focuses on the critical factors to develop communities, by seducing and engaging individuals and management, and by identifying and providing the means to support CoP activities. The model is now being applied to the creation of several CoPs (including the KennisNet community described in chapter 7) and best-practice on its use is being gathered. Experience with the method and its application to different CoPs is being used to fine tune and adapt the different steps.

Many communities at Achmea are spread across business units, which requires special requirements on the support of distributed groups. Therefore, CoP support can be best assisted by the OperA framework to model interaction in communities. Initial results in this area were discussed in chapter 7, when we described the Knowledge Market project.

Although so far the SES method has been only applied within one organization, the communities involved are fairly diverse, and we are confident that the method is generic enough to be applied in other organizational settings and we are currently seeking external application domains for SES. Further research will therefore focus on the analysis of the results of application of the SES method to the development of several CoPs and its further refinement.

References

- [Abecker et al, 1998] Abecker, A., Bernardi, A., Hinkelmann, K., Kühn, O., Sintek, M.: Towards a Technology for Organizational Memories, *IEEE Intelligent Systems & Their Applications*, **13**(3), 1998.
- [Ahuja, Carley, 1998] Ahuja, M., Carley, K.: Network Structure in Virtual Organizations, *Journal of Computer Mediated Communication*, **3**(4), June 1998.
- [Ali et al., 2002] Ali, I., Pascoe, C., Warne, L.: Interactions of Organizational Culture and Collaboration in Working and Learning. *IEEE Journal Educational Technology & Society*. **5**(2), April 2002.
- [Althoff et al, 1998] Althoff, K-D., Bomarius, F., Tautz, C.: Using Case-Based Reasoning Technology to Build Learning Software Organizations, *Proc. 13th Biennial European Conference on Artificial Intelligence (ECAI98)*, Workshop on “Building, maintaining, and using organizational memories”, 1998.
- [Andrade et al., 2001] Andrade, L., Fiadeiro, L., Gouveia, J., Koutsoukos, G., Lopes, A., Wermelinger, M.: Coordination Patterns for Component-Based Systems. In: Musicante M., Haeusler H. (Eds.): *Proc. of Brazilian Symposium on Programming Languages*, 2001, pp. 29 – 39.
- [Artikis et al., 2001] Artikis, A., Kamara, L., Pitt, J.: Towards an Open Agent Society Model and Animation, *Proc. of the 2nd. Agent-Based Simulation Workshop*, Passau, 2001, pp. 48 – 55.
- [Artikis et al., 2002] Artikis A., Pitt J., Sergot M.: Animated Specifications of Computational Societies, In: *Proceedings of Autonomous Agents and Multi-Agent Systems (AAMAS'02)*, 2002, pp. 1053-1062.
- [Artikis, Pitt, 2001] Artikis, A., Pitt, J.: A Formal Model of Open Agent Societies. *Proc. 5th Conference on Autonomous Agents*, ACM, 2001, pp. 192 – 193.
- [Austin, 1962] J.L. Austin: *How to do things with words*, Harvard University Press, 1962.
- [Azary, Woo, 2000] Azary, O., Woo, C.: Analysis and Design of Agent-Oriented Information Systems (AOIS), *Working paper 99-MIS-004*, Faculty of Commerce and Business Administration, University of British Columbia, Vancouver, Canada, 2000.

- [Basili et al, 1994] Basili, V., Caldiera, G., Rombach, H-D.: *Experience Factory*, John Wiley & Sons, 1994.
- [Bauer et al., 2001] Bauer, B., Mueller, J., Odell, J.: Agent UML: A Formalism for Specifying Multi-agent Software Systems. In: Ciancarini, P., Wooldridge, M. (eds.): *Agent-Oriented Software Engineering*, LNCS 1957, Springer, 2001, pp. 91-104
- [Belnap, Perloff, 1988] Belnap, N., Perloff, M.: Seeing To It That: a canonical form for agentives. *Theoria* **54**, 1988, pp. 175-199.
- [Benjamins et al, 1998] Benjamins, V.R., Fensel, D., Gomez Perez, A.: Knowledge Management through Ontologies, *Proc. of the 2nd. International Conference on Practical Aspects of Knowledge Management (PAKM'98)*, Basel, Switzerland, 1998.
- [Biddle, 1979] Biddle, B.J., Thomas, E.J.: *Role Theory: Concepts and Research*, New York, R. E. Krieger Publishing Co., 1979.
- [Bieber et al., 2002] Bieber, M., Engelbart, D., Furuta, R., Hiltz, S., Noll, J., Preece, J., Stohr, E., Turoff, M., van de Walle, B.: Toward Virtual Community Knowledge Evolution, *Journal of Management Information Systems*, **18**(4), 2002, pp. 11 – 36.
- [Boman, 1999] Boman, M.: Norms in Artificial Decision Making. *AI and Law* **7**, 1999, pp. 17 – 35.
- [Bond, Gasser, 1988] Bond, A., Gasser, L.: *Readings in Distributed Artificial Intelligence*, Morgan Kaufmann, (1988)
- [Bondarouk, Pumareja, 2002] Bondarouk, T., Pumareja, D.: Achmea KennisNet Schade System Implementation Evaluation Report. *Internal report*, University of Twente, The Netherlands, 2002.
- [Bonifacio et al, 2000] Bonifacio, M., Bouquet, P., Manzardo, A.: A Distributed Intelligence Paradigm for Knowledge Management, Bringing Knowledge to the Business Process. *Proc. from the AAI Spring Symposium*, Technical Report SS-00-03, AAAI Press, 2000
- [Bonifacio et al., 2002] Bonifacio, M., Bouquet, P., Traverso, P.: Enabling Distributed Knowledge Management: Managerial and Technological Implications, *Novatica and Informatik/Informatique*, **III** (1), 2002.
- [Boutelier, 1994] Boutelier, C.: Toward a Logic for Qualitative Decision Theory. In: Doyle, J., Sandewall, E., Torasso, P. (Eds.): *Principles of Knowledge Representation and Reasoning*, KR'94, (1994).
- [Boyle, 1999] Boyle, D.: The New Alchemists. *Resurgence Magazine*, issue 192, 1999.
- [Bratman, 1987] Bratman, M.: *Intention, Plans and Practical Reason*. Harvard University Press, 1987.
- [Bratman et al., 1988] Bratman, M., Israel, D., Pollack, M.: Plans and Resource-Bounded Practical Reasoning. *Computational Intelligence* **4**, 1988, pp. 349 – 355.
- [Brazier et al., 1997] Brazier, F., Dunin-Keplicz, B., Jennings, N., Treur, J.: DESIRE: Modeling Multi-Agent Systems in a Compositional Formal Framework. In: Huhns, M., Singh M. (eds.): *International Journal of Cooperative Information Systems*, **6**(1), 1997.
- [Brazier et al., 2000] Brazier, F., Jonker, C., Treur, J.: Compositional Design and Reuse of a Generic Agent Model, *Applied Artificial Intelligence Journal*, **14**, 2000, pp. 491-538.
- [Broersen et al., 2001] Broersen, J., Dastani, M., Huang, Z., Hulstijn, J., van der Torre, L.: The BOID Architecture. In: *Proceedings of the 5th International Conference on Autonomous Agents* (Agents'2001), Montreal, 2001.

- [Brooks, 1986] Brooks, R.: A Robust Layered Control System for a Mobile Robot. *IEEE Journal of Robotics and Automation* **2**(1), 1986, pp. 14 – 23.
- [Burghart, 1998] Burghart, T.: Distributed Computing Overview, Quoinc Inc., 1998, http://www.quoininc.com/quoininc/dist_comp.html.
- [Burlage et al., 1998] Burlage, A., van Eeden, P., Sloodman, K.: Networking Knowledge within Circles of Interest in the Eureka Group. *Proceedings 51st ESOMAR Congress and Exhibition: The Power of Knowledge*, Berlin, 1998.
- [Carley, 1994] Carley, K.: Sociology: Computational Organization Theory. *Social Science Computer Review*, **12**(4): 611-624, 1994.
- [Carriero, Gelernter, 1992] Carriero, N., Gelernter, D.: Coordination Languages and their Significance, *Communications of the ACM*, **35**(2), 1992, pp. 97-107.
- [Castelfranchi et al., 2000] Castelfranchi, C., Dignum, F., Jonker, C., Treur, J.: Deliberative Normative Agents: Principles and Architectures. In: Jennings, N., Lesperance, Y. (Eds.): *Intelligent Agent VI*, LNAI 1757, Springer-Verlag, 2000, pp. 364 – 378.
- [Castelfranchi, 1995] Castelfranchi, C.: Commitments: from Individual Intentions to Groups and Organizations. In: V. Lesser (Ed.): *Proc. 1st. International Conference on Multi-Agent Systems (ICMAS'95)*, MIT Press, 1995, pp. 41-48.
- [Castelfranchi, 2000] Castelfranchi, C.: Engineering Social Order. In: Omicini, A., Tolksdorf, R., Zambonelli, F. (Eds.): *Engineering Societies in the Agents World*, LNAI 1972, Springer-Verlag, 2000, pp. 1 – 19.
- [Castelfranchi, Falcone, 1998] Castelfranchi, C., Falcone, R.: Principles of Trust for MAS: Cognitive Anatomy, Social Importance, and Quantification. In: *Proceedings of the Third International Conference on Multi-Agent Systems*, 1998, pp. 72 – 79.
- [Castillo et al., 1998] Castillo, A., Kawaguchi, M., Paciorek, N., Wong, D.: Concordia™ as Enabling Technology for Cooperative Information Gathering. In: *Proc. of Japanese Society for Artificial Intelligence Conference*, Tokyo, Japan, 1998.
- [Chaib-Draa, Dignum, 2002] Chaib-Draa, B., Dignum, F. (Special Issue Eds.): Trends in Agent Communication Language. *Computational Intelligence*, **18**(2), 2002.
- [Chisholm, 1963] Chisholm, R.: Contrary-to-Duty Imperatives and Deontic *Logic Analysis* **24**, 1963, 33-36.
- [Cholvy, Garion, 2001] Cholvy L., Garion C.: An Attempt to Adapt a Logic of Conditional Preferences for Reasoning with Contrary-To-Duties. In: *Fundamenta Informaticae* **47**, 2001.
- [Churchman., 1971] W. Churchman: *The Design of Inquiring Systems: Basic Concepts of Systems and Organization*, Basic Books, New York, 1971.
- [Ciancarini et al., 1999] Ciancarini, P., Omicini, A., Zambonelli, F.: Coordination Models for Multi-Agent Systems. In: *AgentLink News* **3**, July (1999)
- [Ciancarini, Wooldridge, 2001] Ciancarini, P., Wooldridge, M. (Eds.): *Agent-Oriented Software Engineering I*. Revised Papers and Invited Contributions of AOSE 2000. LNAI 1957, Springer-Verlag, 2001.
- [Cleveland, 1997] Cleveland, H.: The Global Confidence Game. In: Harman, W. and Porter, M. (Eds.) *The New Business of Business*, World Business Academy, Berrett-Koehler, 1997.
- [Cohen, Levesque, 1990] Cohen, P., Levesque H.: Intention is Choice with Commitment. *Artificial Intelligence* **42**(2-3), March 1990, pp. 213-261.
- [Cohen, Levesque, 1991] Cohen, P., Levesque H.: Teamwork.. *Nous* **25**(4), 1991, pp. 487 - 512.

- [Comte, Lenzer, 1998] Comte, A., Lenzer, G.: *Auguste Comte and Positivism: The Essential Writings*. History of Ideas Series, Transaction Books, 1998.
- [Cost et al., 2000] Cost, R., Chen, Y., Finin, T., Labrou, Y., Peng, Y.: Using Colored Petri Nets for Conversation Modeling. In: F. Dignum, M. Greaves (Eds.): *Issues in Agent Communication*, LNAI 1916, Springer-Verlag, 2000, pp. 178 – 192.
- [Cranefield et al., 2002] Cranefield, S., Finin, T., Willmott, S.: *Proceedings of the Second International Workshop on Ontologies in Agent Systems*, AAMAS 2002, Bologna, Italy, July 2002, <http://cis.otago.ac.nz/OAS2002/>.
- [Dastani et al., 2001] Dastani, M., Jonker, C., Treur, J.: A Requirement Specification Language for Configuration of Dynamics of Multi-Agent Systems. In: Wooldridge, M., Ciancarini P., Weiss, G. (Eds.): *Agent-Oriented Software Engineering II*. LNAI 2222, Springer-Verlag, 2001.
- [Dastani et al., 2002] Dastani, M., Dignum, F., Dignum, V.: Organizations and Normative Agents. (2002). *Proc. First Eurasian Conference on Advances in Information and Communication Technology – Eurasia ICT*, Tehran, Iran, LNCS 2510, 2002, pp. 982 - 989.
- [Dastani et al., 2003] Dastani, M., Dignum, V., Dignum, F.: Role Assignment in Open Agent Societies. In: *Proc. of AAMAS'03, 2nd International Joint Conference in Autonomous Agents and Multi-Agent Systems*, Melbourne, Australia, July 2003.
- [Dautenhahn, 2000] Dautenhahn, K.: Reverse Engineering of Societies - A Biological Perspective. *Proc. of AISB Symposium "Starting from Society - application of social analogies to computational systems"*. Convention of the Society for the Study of Artificial Intelligence and the Simulation of Behavior (AISB-00), University of Birmingham, England, 2000.
- [Davenport, Prusak, 1998] Davenport, T., Prusak, L.: *Working Knowledge: How Organizations Manage what They Know*, Harvard Business School Press, 1998.
- [Davidsson, 2000] Davidsson, P.: Emergent Societies of Information Agents. Klusch, M, Kerschberg, L. (Eds.): *Cooperative Information Agents IV*, LNAI 1860, Springer, 2000, pp. 143–153.
- [Davidsson, 2001] Davidsson, P.: Categories of Artificial Societies. In: A. Omicini, P. Petta, R. Tolksdorf (Eds.): *Engineering Societies in the Agents World II*, Springer Verlag LNAI 2203, 2001.
- [Delgado, 2000] Delgado, J.: Agent-Based Information Filtering and Recommender Systems, *Ph.D. Thesis*, Nagoya Institute of Technology, Japan, 2000.
- [Dellarocas, 2000] Dellarocas, C.: Contractual Agent Societies: Negotiated Shared Context and Social Control in Open Multi-agent Systems. In: *Proc. of Workshop on Norms and Institutions in Multi-Agent Systems*, Autonomous Agents-2000, Barcelona, 2000.
- [DeLoach, 1999] DeLoach, S.: Multi-agent Systems Engineering: A Methodology and Language for Designing Agent Systems. In: *Proc. of Workshop on Agent-Oriented Information Systems (AOIS'99)*, 1999.
- [Dennett, 1987] Dennett, D.: *The Intentional Stance*. MIT Press, Cambridge, MA, (1987).
- [Dignum et al., 1996] Dignum, F., Weigand, H., Verharen, E.: Meeting the Deadline: on the Formal Specification of Temporal Deontic Constraints. In: Ras, Z., Michalewicz, M. (Eds.): *Foundations of Intelligent Systems*, LNAI 1079, Springer, 1996, pp. 243 – 252.
- [Dignum et al., 2000] Dignum, F., Morley, D., Sonenberg, E., Cavedon, L.: Towards Socially Sophisticated BDI Agents. In: *Proceedings of the 4th International*

- Conference on Multi-Agent Systems (ICMAS-2000)*, Boston, July 2000, IEEE Computer Society, pp. 111-118.
- [Dignum, 1999] Dignum, F.: Autonomous Agents with Norms. *AI and Law* 7, 1999, 69 – 79.
- [Dignum, 2001] Dignum, F.: Agents, Markets, Institutions and Protocols. In: Dignum, F., Sierra, C. (Eds.): *Agent Mediated Electronic Commerce*, LNAI 1991, Springer, 2001, pp. 98 – 114.
- [Dignum, 2002a] Dignum, F.: Abstract Norms and Electronic Institutions. In: *Proc. Int. Workshop on Regulated Agent-Based Social Systems: Theories and Applications (RASTA'02)*, at AAMAS, Bologna, Italy, July, 2002.
- [Dignum, Greaves, 2000] Dignum F., Greaves, M. (Eds.): *Issues in Agent Communication*. LNAI 1916, Springer-Verlag, 2000.
- [Dignum, Kuiper, 1999] Dignum, F., Kuiper, R.: Specifying Deadlines with Dense Time using Deontic and Temporal Logic. In: *International Journal of Electronic Commerce*, 3(2), 1999, pp. 67 – 86.
- [Dignum, Linder, 1997] Dignum, F., van Linder, B.: Modeling Social Agents: Communication as Actions. In: M. Wooldridge J. Muller and N. Jennings (Eds.): *Intelligent Agents III*, LNAI 1193, Springer-Verlag, 1997, pp. 205 – 218.
- [Dignum, Weigand, 1995] Dignum, F., Weigand, H.: Communication and Deontic Logic. In: R. Wieringa, R. Feenstra (Eds.): *Information Systems - Correctness and Reusability. Selected papers from the IS-CORE Workshop*, World Scientific Publishing Co. 1995, pp. 242 – 260.
- [Dignum et al., 2001] Dignum, V., Weigand, H., Xu L.: Agent Societies: Towards Framework-Based Design. In: Wooldridge, M., Ciancarini P., Weiss, G. (Eds.): *Agent-Oriented Software Engineering II*. LNAI 2222, Springer, 2001, pp. 33 - 49.
- [Dignum et al., 2002a] Dignum, V., Meyer, J-J., Weigand, H., Dignum, F.: An Organizational-Oriented Model for Agent Societies. In: *Proc. Int. Workshop on Regulated Agent-Based Social Systems: Theories and Applications (RASTA'02)*, AAMAS, Bologna, Italy, July, 2002.
- [Dignum et al., 2002b] Dignum, V., Meyer, J-J., Weigand, H.: Towards an Organizational Model for Agent Societies Using Contracts. In: *Proc. of AAMAS, the 1st International Joint Conference in Autonomous Agents and Multi-Agent Systems*, Bologna, Italy, July, 2002.
- [Dignum et al., 2003] Dignum, V., Meyer, J.-J., Dignum, F., Weigand, H.: Formal Specification of Interaction in Agent Societies. In: Hinchey, M., Rash, J., Truszkowski, W., Rouff, C., Gordon-Spears, D., (Eds.): *Formal Approaches to Agent-Based Systems (FAABS'02)*. LNAI 2699, Springer 2003.
- [Dignum, 2002b] Dignum, V.: A Knowledge Sharing Model for Peer Collaboration in the Non-Life Insurance Domain. In: *Proc. German Workshop on Experience Management, Lecture Notes in Informatics*, German Society for Informatics, Berlin, 2002.
- [Dignum, 2003] Dignum, V.: Using Agents to Support Knowledge Sharing. In: *Proceedings of Workshop on Autonomy, Delegation and Control: From Inter-agent to Organizations and Institutions*. AAMAS'03, Melbourne, 2003.
- [Dignum, Dignum, 2001] Dignum, V., Dignum, F.: Modeling Agent Societies: Coordination Frameworks and Institutions. In: Brazdil, P, Jorge, A.: *Progress in Artificial Intelligence*. LNAI 2258, Springer-Verlag, 2001, 191 - 204.

- [Dignum, Dignum, 2003] Dignum, V., Dignum, F.: Knowledge Market: Agent-Mediated Knowledge Sharing. In: Marik, V., Müller, J., Pechoucek, M. (Eds.): Multi-Agent Systems and Applications, LNAI 2691, Springer, 2003, pp. 168 – 179.
- [Dignum, Heimannsfeld, 1999] Dignum, V., Heimannsfeld, K.: Knowledge Management for Requirements Engineering. *Proc. of the Knowledge Acquisition Workshops - KAW99*, Banff, Canada, 1999.
- [Dignum, van Eeden, 2003] Dignum, V. van Eeden, P.: Seducing, Engaging and Supporting Communities at Achmea. In: *Proceedings of the 4th European Conference on Knowledge Management*, Oxford, UK, September 18-19, 2003.
- [Dignum, Weigand, 2002] Dignum, V., Weigand, H.: Towards an Organization-Oriented Design Methodology for Agent Societies. In: V. Plekhanova (Ed.): *Intelligent Agent Software Engineering*. Idea Publishers, 2002.
- [Domingue, Motta, 1999] Domingue, J.B., Motta, E.: A Knowledge-Based News Server Supporting Ontology-Driven Story Enrichment and Knowledge Retrieval. *Proc. EKAW'99*, 1999.
- [Drucker, 1989] Drucker, P.: *The New Realities : In Government & Politics, in Economics & Society, in Business, Technology & World View*. Harper & Row, 1989.
- [Drucker, 1995] Drucker, P.: *Managing in a Time of Great Change*, Tuman Talley, New York, 1995.
- [Dunin-Keplicz, Verbrugge, 2002] Dunin-Keplicz, B., Verbrugge, R: Collective Intentions, *Fundamenta Informaticae* **51**(3), 2002, pp. 271 – 295.
- [Dunin-Keplicz, Verbrugge, 2003] Dunin-Keplicz, B., Verbrugge, R: Calibrating Collective Commitments. In: Marik, V., Müller, J., Pechoucek, M. (Eds.): Multi-Agent Systems and Applications, LNAI 2691, Springer, 2003, pp. 73 – 83.
- [Edmonds, 1999] Edmonds, B.: Capturing Social Embeddedness: a Constructivist Approach. *Adaptive Behavior* **7**, 1999, pp. 323-348.
- [Emerson, 1990] Emerson, E.: Temporal and Modal Logic. In: J. van Leeuwen (Ed.): *Handbook of Theoretical Computer Science*, Elsevier Science Publishers, 1990.
- [Esteva et al., 2001] Esteva, M., Rodriguez, J., Sierra, C., Garcia, P., Arcos J.: On the formal specifications of electronic institutions, In: Dignum F., Sierra C. (Eds.): *Agent-mediated Electronic commerce (The European AgentLink Perspective)*, LNAI 1991, Springer, 2001, pp. 126-147.
- [Esteva et al., 2002a] Esteva, M., Padget, J., Sierra, C.: Formalizing a Language for Institutions and Norms. *Intelligent Agents VIII*, LNAI 2333, Springer, 2002, pp. 348-366.
- [Esteva et al., 2002b] Esteva, M., de la Cruz, D., Sierra, C.: ISLANDER: an Electronic Institutions Editor. In: C. Casterfranchi, W. L. Johnson (Eds.): *Proc. of the First International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2002)*, Bologna, Italy, 2002, pp. 1045-1052.
- [Fensel, 2001] Fensel, D.: *Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce*. Springer-Verlag, 2001.
- [Ferber et al., 2001] Ferber, J., Gutknecht, O., Jonker, C., Müller, J-P., Treur, J.: Organizational Models and Behavioral Requirements Specification for Multi-Agent Systems. In: Demazeau, Y., Garijo, F. (Eds.): *Multi-Agent Systems Organizations*. Proc. of 10th European Workshop on Modeling Autonomous Agents in a Multi Agent World, 2001.

- [Ferber, Gutknecht, 1998] Ferber, J., Gutknecht, O.: A Meta-Model for the Analysis and Design of Organizations in Multi-Agent Systems. In: *Proc. of the 3rd. International Conference on Multi-Agent Systems (ICMAS'98)*, IEEE Computer Society, 1998.
- [Fergusson, 1995] Fergusson, I.: Integrated Control and Coordinated Behavior: A case for agent models. M. Wooldridge, N. Jennings (Eds.): *Intelligent Agent: Theories, Architectures and Languages*, LNAI 890, Springer, 1995, pp. 203 – 218.
- [Findler, Malyankar, 2000] Findler, N.V., Malyankar, R.M.: Social Structures and the Problem of Coordination in Intelligent Agent Societies. *Invited talk at the Special Session on Agent-Based Simulation, Planning and Control in IMACS World Congress 2000*, Lausanne, Switzerland, 2000.
- [Finin et al., 1997] Finin, T., Labrou, Y., Mayfield, J.: KQML as an Agent Communication Language, in Bradshaw, J. (Ed.): *Software Agents*, MIT Press, Cambridge, 1997.
- [Finkelstein, Kramer, 2000] Finkelstein, A., Kramer, J.: Software Engineering: A Roadmap. In: Finkelstein, A.: *The Future of Software Engineering*, ACM Press, 2000, pp. 3- 24.
- [FIPA, 2002] FIPA Communicative Act Library Specification. Document Number: SC00037J, <http://www.fipa.org/specs/fipa00037/SC00037J.pdf>.
- [Fischer, 1994] Fischer, M.: A Survey of METATEM, the Language and its Applications. In: Gabbay, D., Ohlbach, H. (Eds.): *Proceedings of the 1st. International Temporal Logic Conference*, LNAI 827, Springer, 1994, pp. 480 – 505.
- [Flores, Ludlow, 1980] Flores, F., Ludlow, J.J.: Doing and Speaking in the Office. In: G. Fick and H. Sprague Jr. (Eds.): *Decision Support Systems: Issues and Challenges*, Pergamon Press, NY, 1980
- [Fox, Gruniger, 1998] Fox, M., Gruninger, M.: Enterprise Modeling. *AI Magazine*, AAAI Press, Fall, 1998, pp. 109-121.
- [Franklin, Gasser, 1996] Franklin, S., Gasser, L.: Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents, In: *Intelligent Agents III*, Springer-Verlag, 1997, pp. 21-35.
- [Frederiksson, Gustavsson, 2001] Frederiksson, M., Gustavsson, R.: A Methodological Perspective on Engineering of Agent Societies. In: Omicini, A., Petta, P., Tolkdorf, R. (eds.). *2nd International Workshop on Emerging Societies in the Agent's World (ESAW'01)*, Prague, 2001.
- [Frei, Faltings, 1998] Frei, C., Faltings, B.: A Dynamic Hierarchy of Intelligent Agents for Network Management. In: *Proc. of 2nd International Workshop on Intelligent Agents for Telecommunications Applications (IATA'98)*, Paris, 1998, pp. 1-16
- [Gammack, 1998] Gammack, J.: Organisational Memory and Intelligent Decision Support in Shared Information Systems, *Research Working Paper IT/98/05*, Department of Information Technology, Murdoch University, Western Australia, 1998, http://www.it.murdoch.edu.au/research/working_papers/it9805.doc.
- [Gandon et al., 2000] Gandon, F., Dieng, R., Corby, O., Giboin, A.: A Multi-Agent System to Support Exploiting an XML-based Corporate Memory. In: *Proc. PAKM'00*, Basel, 2000.
- [Gasser, 2001] Gasser, L.: Perspectives on Organizations in Multi-agent Systems. In: Luck, M., Marik, V., Stepankova, O., Trappl, R. (Eds.): *Multi-Agent Systems and Applications*. LNAI 2086, Springer-Verlag, Berlin, (2001), 1-16.
- [Genesereth, Nilsson, 1987] Genesereth, M., Nilsson, N.: *Logical Foundations of Artificial Intelligence*, Morgan Kaufman, 1987.

- [Georgeff, Lansky, 1987] Georgeff, M., Lansky, A.: Reactive Reasoning and Planning. IN: Proceedings of the 6th National Conference in AI (AAAI-87), Seattle, WA, 1987, pp. 677 – 682.
- [Giunchiglia et al., 2002a] Giunchiglia, F., Odell, J., Weiss, G. (Eds.): *Agent-Oriented Software Engineering III. Revised Papers and Invited Contributions, AOSE 2002*. Lecture Notes in Artificial Intelligence, Volume 2585, Springer-Verlag, 2002.
- [Giunchiglia et al., 2002b] Giunchiglia, F., Mylopoulos, J., Perini, A.: The Tropos Development Methodology: Processes Models and Diagrams. In: Giunchiglia, F., Odell, J., Weiss, G. (Eds.): *Agent-Oriented Software Engineering III*. LNAI 2585, Springer-Verlag, 2002, pp. 162 – 173.
- [Giunchiglia, Bouquet, 1997] Giunchiglia, F., Bouquet, P.: Introduction to Contextual Reasoning. In: Kokinov, B. (Ed.): *An Artificial Intelligence Perspective, Perspectives on Cognitive Science*, 3, NBU Press, Sofia, Bulgaria, 1997.
- [Goffman, 1959] Goffman, E.: *The Presentation of Self in Everyday Life*. Double Day Anchor Books, Garden City, NY, 1959.
- [Gomez-Perez, Benjamins, 1999] Gomez-Perez A., Benjamins, V.R.: Overview of Knowledge Sharing and Reuse Components: Ontologies and Problem-Solving Methods: In: Benjamins, V.R. (ed.): *Proc. of the IJCAI-99 Workshop on Ontologies and Problem-Solving Methods: Lessons Learned and Future Trends*, 1999.
- [Grosz, Labrou, 1999] Grosz, B., Labrou, Y.: An Approach to Using XML and a Rule-Based Content Language with an Agent Communication Language, *Proc. of 16th. International Joint Conference on Artificial Intelligence (IJCAI-99)*, Workshop on Agent Communication Languages, 1999.
- [Grosz, Kraus, 1996] Grosz, B., Kraus, S.: Collaborative Plans for Complex Group Actions. *Artificial Intelligence* **86**, 1996, pp. 269-358.
- [Gruber, 1993] Gruber, T.: Toward Principles for the Design of Ontologies Used for Knowledge Sharing. In: Guarino, N., Poli, R. (Eds.): *Formal Ontology in Conceptual Analysis and Knowledge Representation*, Kluwer, 1993
- [Gurteen, 1998] Gurteen, D.: Knowledge, Creativity and Innovation, *Journal of Knowledge Management*, **2**(1), 1998, pp. 5-13.
- [Haddadi, 1996] Haddadi, A.: *Communication and Cooperation in Agent Systems*, LNAI 1056, Springer, 1996.
- [Halpern, Moses, 1992] Halpern J., Moses, Y.: A Guide to the Completeness and Complexity of Modal Logic of Knowledge and Belief. *Artificial Intelligence* **54**(3), 1992, pp. 319-379.
- [Harman, Porter, 1997] Harman, W., Porter M. (Eds.): *The New Business of Business: Sharing Responsibility for a Positive Global Future*. World Business Academy. Berrett-Koehler Publishers, 1997.
- [Hindriks et al., 1999] Hindriks, K., de Boer, F., van der Hoek, W., Meyer, J-J.: Agent Programming in 3APL. *Journal of Autonomous Agents and Multi-Agent Systems* **2**(4), Kluwer Academic Press, 1999, pp. 357-401.
- [Hintikka, 1992] Hintikka, J.: *Knowledge and Belief*. Cornell University Press, Ithaca (NY), 1962.
- [Horty, Belnap, 1995] Horty, J., Belnap, N.: The Deliberative Stit: A Study of Action, Omission, Ability and Obligation. *Journal of Philosophical Logic* **24**, 1995, pp. 583-644.
- [Huhns, Abdulla, 1999] Huhns, M., Abdulla, M.: Benevolent Agents, *IEEE Internet Computing* **3**(2), March-April 1999, pp. 96--98

- [Huhns, Stephens, 1999] Huhns, M., Stephens, L.: Multiagent Systems and Societies of Agents. In: Weiss, G. (Ed.), *Multi-agent Systems: a Modern Approach to Distributed Artificial Intelligence*, MIT Press, 1999.
- [Jennings et al, 1996] Jennings, N., Faratin, P., Johnson, M., Norman, T., O'Brien, P., Wiegand, M.: Agent-Based Business Process Management. *International Journal of Cooperative Information Systems*, 5(2&3), 1996, pp. 105-130.
- [Jennings et al., 1998] Jennings, N., Sycara, K., Wooldridge, M.: A Roadmap for Agent Research and Development. *Journal of Autonomous Agents and Multi-Agent Systems* 1, Kluwer Academic Press, 1998, pp. 275-306.
- [Jennings, 2000] N. Jennings: On Agent-Based Software Engineering, *Artificial Intelligence* 117, 2000, pp. 277-296.
- [Jennings, Wooldridge, 1998] Jennings, N., Wooldridge, M. (eds.), *Agent Technology: Foundations, Applications and Markets*, Springer, 1998.
- [Jones, Sergot, 1993] Jones, A., Sergot, M.: On the Characterization of Law and Computer Systems: The Normative Systems Perspective. In: J.-J. Meyer, R. Wieringa (Eds.): *Deontic Logic in Computer Science: Normative System Specification*, John Wiley & Sons, 1993, pp. 275 – 307.
- [Jones, Sergot, 1996] Jones, A., Sergot, M.: A Formal Characterization of Institutionalized Power. *Journal of the IGPL* 4(3), 1996, pp. 429-445.
- [Jonker et al., 2000] Jonker, C., Klusch, M., Treur, J.: Design of Collaborative Information Agents. In: Klusch M., Kerschberg, L. (eds.): *Cooperative Information Agents IV*. LNAI 1860, Springer-Verlag, 2000, pp. 262 – 283
- [Kalenka, 2001] Kalenka, S.: Modelling Social Interaction Attitudes in Multi-Agent Systems. Ph.D. thesis, Department of Electronic Engineering, University of London, 2001.
- [Kearney, 1998] Kearney, P.: Personal Agents: A Walk on the Client Side, in [Jennings and Wooldridge, 1998].
- [Khalil, 2002] Khalil, A.: *Multi-Agent System for Achmea*. MSc. Thesis, Utrecht University, 2002.
- [Klusch, 1999] Klusch, M. (Ed.): *Intelligent Information Agents: Agent-based Information Discovery and Management in the Internet*, Springer, 1999.
- [Klusch, Kerchberg, 2000] Klusch, M., Kerschberg, L. (Eds.): *Cooperative Information Agents IV*, Proceedings of Fourth International Workshop CIA-2000 on Cooperative Information Agents, Boston, LNAI 1860, Springer-Verlag, 2000.
- [Knownet, 2000] KnowNet Initiative, 2000, <http://www.knownet.org>.
- [Kollingbaum, Norman, 2002] Kollingbaum, M., Norman, T.: Supervised Interaction - Creating a Web of Trust for Contracting Agents in Electronic Environments. In: Castelfranchi, C., Johnson, W. (Eds.): *Proceedings 1st International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'02)*, ACM Press, New York, 2002, pp. 272 – 279.
- [Kumar et al., 2002] Kumar, S., Huber, M., Cohen, P., McGee, D.: Towards a Formalism for Conversation Protocols Using Joint Intention Theory. In: Chaib-Draa, B. Dignum F. (Special Issue Eds.): *Trends in Agent Communication Language, Computational Intelligence* 18(2), 2002, pp. 174 – 228.
- [Lassila, Swick, 1999] Lassila, O., Swick, R.: Resource Description Framework (RDF): Model and Syntax Specification. W3C Recommendation, 1999, <http://www.w3.org/Press/1999/RDF-REC>.

- [Lawrence, Lorsch, 1967] Lawrence, P., Lorsch, J.: *Organization and Environment: Managing Differentiation and Integration*. Harvard Business School Press, 1967.
- [Leake et al, 1999] Leake, D., Birnbaum, L., Hammond, K., Marlow, C., Yang, H.: Task-Based Knowledge Management. *Proc. of the AAI Workshop on Exploring Synergies of Knowledge Management and Case-Based Reasoning*, Orlando, Florida, AAI Technical Report WS-99-10, 1999.
- [Lésperance et al., 1996] Lésperance, Y., Levesque, H., Lin, F., Marcu, D., Reiter, R., Scherl, R.: Foundations of a Logical Approach to Agent Programming. In: Wooldridge, M., Müller, J., Tambe, M. (Eds.): *Intelligent Agents II*, LNAI 1037, Springer, 1996, pp. 331 - 346.
- [Liao et al, 1999] Liao, M., Abecker, A., Bernardi, A., Hinkelmann, K., Sintek, M.: Ontologies for Knowledge Retrieval in Organizational Memories. *Workshop on Learning Software Organizations (LSO) at SEKE'99*, 1999.
- [Lietaer, 2001] Lietaer, B.: *The Future of Money: Creating New Wealth, Work and a Wiser World*. Century Publishers, 2001.
- [Lietaer, 2002] Lietaer, B.: *The Future of Payment Systems*. White Paper, http://www.unisys.com/products/insights/insights_compendium/payment.pdf, Unisys Corporation, 2002.
- [Likert, 1961] Likert, R.: *New Patterns of Management*. New York: McGraw-Hill Book Company, 1961.
- [Lind, 2001] Lind, J.: *Iterative Software Engineering for Multiagent Systems: The MASSIVE Method*. LNAI 1994, Springer-Verlag, 2001.
- [Lomuscio et al., 2001] Lomuscio, A., Wooldridge, M., Jennings, N.: A Classification Scheme for Negotiation in Electronic Commerce. In: Dignum, F., Sierra, C. (Eds.): *Agent Mediated Electronic Commerce*, LNAI 1991, Springer, 2001, pp. 19 - 33.
- [Luck et al., 2003] Luck, M., McBurney, P., Preist, C.: *Agent Technology: Enabling Next Generation Computing: A Roadmap for Agent Based Computing*. AgentLink II, 2003.
- [Maes, 1990] Maes, P. (Ed.): *Designing Autonomous Agents*. MIT Press, 1990.
- [Malone, Crowston, 1994] Malone, T., Crowston, K.: The Interdisciplinary Study of Coordination. *ACM Computing Surveys* **26**(1), March, 1994.
- [McCarthy, 1987] McCarthy, J.: Generality in Artificial Intelligence. *Communications of the ACM* **30**(12), 1987, pp. 1030-1035.
- [Mentzas et al., 2001] Mentzas, G., Apostolou, D., Young, R., Abecker, A.: Knowledge Networking: a Holistic Solution for Leveraging Corporate Knowledge. *Journal of KM* **5**(1), MCB, 2001.
- [MESSAGE, 2000] MESSAGE: Methodology for Engineering Systems of Software Agents - Initial Methodology. EURESCOM Project P907-GI, (2000). URL: <http://www.eurescom.de/~pub-deliverables/P900-series/P907/D1>.
- [Meyer et al., 1998] Meyer, J.-J., Wieringa, R., Dignum, F.: The Role of Deontic Logic in the Specification of Information Systems. In Chomicki J., Saake, G. (eds.): *Logics for Databases and Information Systems*, Kluwer Academic Publishers, 1998, pp. 71-115
- [Meyer, van der Hoek, 1995] Meyer J.-J., van der Hoek, W.: *Epistemic Logic for Computer Science and Artificial Intelligence*, Cambridge Tracts in Theoretical Computer Science 41, Cambridge University Press, 1995.
- [Meyer, Wieringa, 1993] Meyer. J.-J., Wieringa, R.: *Deontic Logic in Computer Science*, Wiley, 1993.

- [Miceli et al., 1996] Miceli, M., Cesta, A., Rizzo, P.: Distributed Artificial Intelligence from a Socio-Cognitive Standpoint: Looking at Reasons for Interaction, *Artificial Intelligence and Society* **9**, 1996, pp. 287-320.
- [Minsky, Rozenshtein, 1989] Minsky, N., Rozenshtein, D.: Controllable Delegation: An Exercise in Law-governed Systems. In: Meyrowitz, N. (Ed.): *Proc. of the ACM Conference on Object-Oriented Programming Systems, Languages and Applications (OOPSLA'89)*, SIGPLAN Notices, 1989, pp. 371-380.
- [MIT Manifesto, 1999] MIT 21st. Century Manifesto Working Group: *What do We Really Want? A manifesto for the Organisations of the 21st. Century*. Sloan School of Management, MIT, 1999, <http://ccs.mit.edu/>.
- [Mohamed, Huhns, 2000] Mohamed, A., Huhns, M.: Multi-Agent Benevolence as a Societal Norm, *Workshop on Norms and Institutions in Multi-agent Systems, at the 4th International Conference on Autonomous Agents (Agents 2000)*, Barcelona, 2000.
- [Morciniec et al., 2001] Morciniec, M., Salle, M., Monahan, B.: Towards Regulating Electronic Communities with Contracts, *Proc. of 2nd Workshop on Norms and Institutions in MAS, Autonomous Agents 2001*, Montreal, 2001.
- [Moses, Tennenholtz, 1995] Moses, Y., Tennenholtz, M.: Artificial Social Systems. *Computers and AI* **14**(6), 1995, pp. 533-562.
- [Müller et al., 1995] Müller, J., Pischel, M., Thiel, M.: Modelling Reactive Behavior in Vertically Layered Agent Architectures. In: Wooldridge, M., Jennings, N. (Eds.): *Intelligent Agent: Theories, Architectures and Languages*, LNAI 890, Springer, 1995, pp. 261 - 276.
- [Negroponte, 1995] Negroponte, N.: *Being Digital*, Hodder & Stoughton, 1995.
- [Newell, 1993] Newell, A.. Reflections on the Knowledge Level. *Artificial Intelligence* **59**, 1993, pp. 31-38.
- [Nickles, Weiss, 2003] Nickles, M., Weiss, G.: Empirical Semantics of Agent Communication in Open Systems. In: *Proceedings of the 2nd International Workshop on Challenges in Open Agent Environments, AAMAS'03*. Melbourne, Australia, 2003.
- [Noll, 2003] Noll, J.: Process Enactment: A Foundation for Managing Knowledge Intensive Work Processes, MKIDS White Paper, 2003, <http://cse.scu.edu/~jnoll/noll-mkids.pdf>.
- [Nonaka, 1991] Nonaka, I.: The Knowledge-Creating Company. Reprinted in 1998 in: *Harvard Business Review on Knowledge Management*, Harvard Business School Press.
- [Norman, Reed, 2002] Norman, T., Reed, C.: A Model of Delegation for Multi-Agent Systems. In: d'Inverno, M., Luck, M., Fisher, M., Preist, C. (Eds.): *Foundations and Applications of Multi-Agent Systems*, LNAI 2403, Springer-Verlag, 2002, pp. 185 – 204.
- [Nouwens, Bouwman, 1995] Nouwens, J., Bouwman H.: Living Apart Together in Electronic Commerce: The Use of Information and Communication Technology to Create Network Organisations. Steinfield, C. (ed.): Steinfield, C. (ed.): Special Issue in Electronic Commerce, *Journal of Computer Mediated Communication* **1**(3), 1995.
- [Nwana, Ndumu, 1998] Nwana, H.S., Ndumu, D.T.: A Brief Introduction to Software Agent Technology. In: [Jennings, Wooldridge, 1998].
- [O'Malley, DeLoach, 2001] O'Malley, S., DeLoach, S.: Determining When to Use an Agent-Oriented Software Engineering Paradigm. In: Wooldridge, M., Ciancarini P., Weiss,

- G. (Eds.): *Agent-Oriented Software Engineering II*. LNAI 2222, Springer, 2001, pp. 188 - 205.
- [Odell et al., 2001] Odell, J., Van Dyke Parunak, H., Bauer, B.: Representing Agent Interaction Protocols in UML. In: Ciancarini P., Wooldridge, M. (eds.): *Agent-Oriented Software Engineering*, LNCS 1957, Springer, 2001, pp. 121 – 140.
- [Odell et al., 2003] Odell, J., Van Dyke Parunak, H., Fleischer, M.: The Role of Roles in Designing Effective Agent Organizations. In: Alessandro Garcia et al (Eds.): *Software Engineering for Large-Scale Multi-Agent Systems*, LNCS, Springer, 2003.
- [Oliveira, 1999] Oliveira, E.: Applications of Agent-based Intelligent Systems. *Proceedings of 4th SBAI- Brazilian Symposium of Intelligent Automation*, S.Paulo, Brasil, 1999.
- [Omicini, 2000] Omicini, A.: From Objects to Agent Societies: Abstractions and Methodologies for the Engineering of Open Distributed Systems. Corradi A., Omicini, A., & Poggi, A. (eds.). *WOA 2000 – Dagli Oggetti agli Agenti: Tendenze evolutive dei sistemi software*, Bologna: Pitagora Editrice, 2000.
- [Omicini, 2001] Omicini, A.: SODA: Societies and Infrastructures in the Analysis and Design of Agent-based Systems. In: Ciancarini P., Wooldridge, M. (eds.): *Agent-Oriented Software Engineering*, LNCS 1957, Springer-Verlag, 2001, pp. 185 – 193.
- [Ossowski, 1998] Ossowski, S.: *Co-ordination in Artificial Agent Societies*, LNAI 1535, Springer, 1998.
- [Pacheco, Carmo, 2003] Pacheco O., Carmo, J.: A Role Based Model for the Normative Specification of Organized Collective Agency and Agents Interaction. *Journal of Autonomous Agents and Multi-Agent Systems*, to be published, 2003.
- [Papadopoulos, Arbab, 1998] Papadopoulos, G., Arbab, F.: Co-ordination Models and Languages. Marvin V. Zelkowitz (ed.) *Advances in Computers*, 46, Academic Press, pp. 329-400, 1998.
- [Panzarasa et al., 2001] Panzarasa P., Jennings N. R., Norman T.: Formalizing collaborative decision-making and practical reasoning in multi-agent systems. *Journal of Logic and Computation*, **12**(1), 2002, pp. 55-117 .
- [Parsons, 1956] Parsons, T.: Suggestions for a Sociological Approach to the Theory of Organizations. *Administrative Science Quarterly* **1**, 1956, pp. 63-85.
- [Parunak, Odell, 2002] Parunak, H. V. D., Odell, J.: Representing Social Structures in UML. In: Wooldridge, M., Weiss, G., Ciancarini P. (Eds.): *Agent-Oriented Software Engineering II*, LNCS 2222, Springer, 2002.
- [Pfeffer, 1997] Pfeffer, J.: *New Directions for Organization Theory: Problems and Prospects*. Oxford University Press, 1997.
- [Pitt et al., 1999] Pitt, J., Mamdani, A., Charlton, P.: The Open Agent Society and Its Enemies. In: Stathis, K. (ed.): *Local Nets: Proceedings of International Workshop on Community-Based Interactive Systems*, Siena, 1999, pp 89–104.
- [Popper, 1945] Popper, K.: *The Open Society and its Enemies*, Routhledge, London, 1945. (Golden Jubilee Edition, 1995).
- [Popper, 1982] Popper, K.: *The Open Universe: An Argument for Indeterminism*, Hutchinson, London, 1982.
- [Pörn, 1974] Pörn, I.: Some Basic Concepts of Action. In: Stenlund, S. (Ed.): *Logical Theory and Semantic Analysis*. Reidel, Dordrecht, 1974.
- [Powell, 1990] Powell, W.: Neither Market nor Hierarchy: Network Forms of Organisation. *Research in Organisational Behavior* **12**, 1990, pp. 295-336.

- [Preece et al., 1999] Preece, A., Hui K., Gray, P.: KRAFT: Supporting Virtual Organisations through Knowledge Fusion. In: Finin T., Grosz B. (Eds): *Artificial Intelligence for Electronic Commerce: Papers from the AAAI-99 Workshop*, AAAI Press, 1999, pp. 33-38.
- [Pumareja et al., 2003] Pumareja, D., Bondarouk, T., Sikkil, K.: Supporting Knowledge Sharing Isn't Easy - Lessons Learnt from a Case Study. *Information Resource Management Association International Conference (IRMA'03)*, Philadelphia, 2003.
- [Rao, Georgeff, 1992] Rao, A., Georgeff, M.: An Abstract Architecture for Rational Agent. In: C. Rich, W. Swartout, B. Nebel (Eds.): *Proc. of KR'92*, Morgan Kaufmann, 1992, pp. 439 - 449.
- [Rao, Georgeff, 1995] Rao, A., Georgeff, M.: BDI Agents: From Theory to Practice. In: Lesser, V. (Ed.): *Proc. of the 1st Int. Conference in Multi-Agent Systems (ICMAS'95)*, IEEE Computer Society Press, 1995, pp. 312-319.
- [Reeves et al., 2002] Reeves, D., Wellman, M., Grosz, B., Chan, H.: Automated Negotiation from Declarative Contract Descriptions. In: Spencer, B. (Ed.): Special Issue on Agent Technologies for Electronic Commerce, *Computational Intelligence Journal* **18**, Blackwell Publishing, 2002.
- [Sallé, 2002] Sallé, M.: Electronic Contract Framework: An Overview. In: Proc. of AAAI-2002 Workshop on Agent technologies for B2B E-Commerce, 2000.
- [Salter, Liu, 2002] Salter, A., Liu, K.: Using Semantic and Norm Analysis to Model Organizations. In: J. Bráz, M. Piattini, J. Filipe (Eds.): *Proc. of ICEIS 2002*, Ciudad Real, Spain, 2002, pp. 847 – 850.
- [Sandholm, Lesser, 1995] Sandholm T., Lesser V.: Issues in Automated Negotiation and Electronic Commerce: Extending the Contract Net Framework. *Proceedings 1st. International Conference on Multiagent Systems (ICMAS-95)*, 1995, pp. 328-335.
- [SAS, 1999] SAS White Paper, The Convergence of Business Intelligence and Knowledge Management, SAS Institute and Intraspect Software Partnership, 1999.
- [Schmid, Stanoevsk-Slabeva, 1998] Schmid, B., Stanoevsk-Slabeva, K.: Knowledge Media: An Innovative Concept and Technology for Knowledge Management in the Information Age. *Beyond Convergence: 12th Biennial International Telecommunications Society Conference*, Stockholm, Sweden, June 1998.
- [Scott, 1987] Scott, W.R.: *Organizations: Rational, Natural and Open Systems*, Prentice-Hall, 1987.
- [Searle, 1969] Searle, J.R.: *Speech Acts: An Essay in the Philosophy of Language*, Cambridge University Press, 1969.
- [Searle, Vanderveken, 1985] Searle, J.R., Vanderveken, D.: *Foundations of Illocutionary Logic*, Cambridge University Press, 1985.
- [Serrano, Ossowski, 2002] Serrano, J., Ossowski, S.: An Approach to Agent Communication Based on Organizational Roles. In: M. Klusch, S. Ossowski, O. Shehory (Eds.): *Cooperative Information Agents VI*, LNAI 2446, Springer, 2002, pp. 241-248.
- [Shoham, Tennenholtz, 1995] Shohan, Y., Tennenholtz, M.: On Social Laws for Artificial Agent Societies: Off-line Design. *Artificial Intelligence* **73**(1-2), 1995, pp. 231-252.
- [Sichman, Conte, 1998] Sichman, J.S., Conte, R.: On Personal and Role Mental Attitudes: A Preliminary Dependency-Based Analysis. In: Oliveira, F. (Ed.): *Advances in AI: Proc. of the 14th Brazilian Symposium on AI*, LNAI 1515, Springer, 1998.
- [Siebert, 2002] Siebert, H. (Ed.): *Economic Policy for Aging Societies*. Springer-Verlag, 2002.

- [Silva, Demazeau, 2002] Silva, J., Demazeau, Y.: The Vowels Co-ordination Model. In: Casterfranchi, C., Johnson, W. L. (Eds.): *Proc. of the 1st. International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2002)*, Bologna, Italy, 2002, pp. 1129-1136.
- [Simon, 1957] Simon, H.: *Models of Man*, Wiley, New York, 1957.
- [Singh, 1998] Singh, M.: Agent Communication Languages: Rethinking the Principles. *IEEE Computer* **31**, 1998, pp. 40-47.
- [Smith et al., 1998] Smith, I., Cohen, P., Bradshaw, J., Greaves, M., Holmback, H.: Designing Conversation Policies Using Joint Intention Theory. *Proceedings of the Third International Conference on Multi Agent Systems (ICMAS-98)*, Paris, IEEE Press, 1998, pp. 269-276.
- [Smith, 1980] Smith, R.G.: The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver. *IEEE Transaction on Computers* **C-29**(12), 1980, pp. 1104-1113.
- [So, Durfee, 1998] So, Y-P., Durfee, E.: Designing Organizations for Computational Agents. In: Prietula, M., Carley, K., and Gasser, L.: *Simulating Organizations*, AAAI Press, 1998, pp. 47 – 64.
- [Sowa, 2000] Sowa, J.: Ontology, Metadata and Semiotics. B. In: Ganter, B., Mineau, G. (Eds.): *Conceptual Structures: Logical, Linguistics and Computational Issues*, LNAI 1867, Springer, 2000, pp. 55-81.
- [Staab et al, 2000] Staab, S., Erdmann, M., Mädche, A., Decker, S.: An Extensible Approach for Modelling Ontologies in RDF(S). *Proceedings of the ECDL-2000 Workshop "Semantic Web: Models, Architectures and Management"*, Lisbon, 2000.
- [Staab, Schnuur, 1999] Staab, S., Schnurr H-P.: Knowledge and Business Processes: Approaching an Integration. *OM '99 - Proceedings of the international Workshop on Knowledge Management and Organisational Memory (IJCAI-99)*. Stockholm, 1999.
- [Stamper et al., 1988] Stamper, R., Althaus, K., Backhouse, J.: MEASUR: Method for Eliciting, Analyzing and Specifying User Requirements. In: T. Olle, A. Verrijn-Stuart, L. Bhabuts (Eds.): *Computerized Assistance During the Information System Life-cycle*, Elsevier Science, Amsterdam, 1988, pp. 67 – 116.
- [Stein, Zwass, 1995] Stein, E., Zwass, V.: Actualizing Organizational Memory with Information Systems. *Information Systems Research* **6**(2), June 1995, pp. 85-117.
- [Sycara et al., 1998] Sycara, K., Decker, K., Zeng, D.: Intelligent Agents in Portfolio Management. In: Jennings, N., Woolridge, M. (Eds.): *Agent Technology: Foundations, Applications, and Markets*, Springer, 1998, pp. 267-283.
- [Sycara et al., 2003] Sycara, K., Paolucci, M., van Velsen, M., Giampapa, J.: The RETSINA MAS Infrastructure. *Journal of Autonomous Agents and Multi-Agent Systems* **7**(1 & 2), Kluwer Academic Publishers, 2003.
- [Sycara, 1998] Sycara, K.: Multiagent Systems. *AI Magazine* **19**(2) , 1998, pp. 79-92.
- [Tambe, 1997] Tambe, M.: Towards Flexible Teamwork. *Journal of Artificial Intelligence Research* **7**, 1997, pp. 83 – 124.
- [Tan, Thoen, 2000] Tan, Y-H., Thoen, W.: DocLog: An Electronic Contract Representation Language. *Proceedings of DEXA 2000 Workshop*, London, 2000, pp. 1069 – 1073.
- [Treacy, Wiersema, 1997] Treacy, M., Wiersema, F.: *The Discipline of Market Leaders*, Perseus Press, 1997.

- [Tsvetovat et al., 2001] Tsvetovat, M., Sycara, K., Chen, Y., Ying, J.: Customer Coalition in Electronic Markets. In: Dignum, F., Cortés, U. (Eds.): *AMEC III*, LNAI 2003, Springer, 2001, pp. 121-138.
- [van der Torre, 2003] Van der Torre, L.: Contextual Deontic Logic: Normative Agents, Violations and Independence. In: Special Issue on Computational Logic in Multi-Agent Systems, *Annals of Mathematics and Artificial Intelligence* **37**(1-2), 2003, pp. 33-63.
- [van Elst et al., 2003a] L. van Elst, V. Dignum, A. Abecker (Eds.): *Agent-Mediated Knowledge Management: Selected Papers*, LNAI 2926, Springer-Verlag, 2003.
- [van Elst et al., 2003b] van Elst, L., Dignum, V., Abecker, A.: Towards Agent-Mediated Knowledge Management. In: L. van Elst, V. Dignum, A. Abecker (Eds.): *Agent-Mediated Knowledge Management: Selected Papers*, LNAI 2926, Springer-Verlag, 2003.
- [van Engers, 2001] van Engers, T.: *Knowledge Management: The Role of Mental Models in Business Systems Design*. Ph.D. Thesis, Free University of Amsterdam, 2001.
- [van Schooten, 1999] van Schooten, B.: Building a Framework for Developing Interaction Models: Overview of Current Research on Dialogue and Interactive Systems. *CTIT technical report 99-04* (ISSN 1381-3625), University of Twente, The Netherlands, 1999.
- [Varian, Resnick, 1997] Varian, H., Resnick, P.: Recommender Systems. *Communications of the ACM* **40**(3), 1997.
- [Vasconcelos et al., 2002] Vasconcelos, W., Sabater, J., Sierra, C., Querol, J.: Skeleton-Based Agent Development for Electronic Institutions. In: Casterfranchi, C., Johnson, W. L. (Eds.): *Proc. of the 1st. International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2002)*, Bologna, Italy, 2002, pp. 696-703.
- [Vázquez-Salceda, 2003] Vázquez-Salceda, J.: The Role of Norms and Electronic Institutions in Multi-Agent Systems Applied to Complex Systems: The HARMONIA framework. *Ph.D. Thesis*, Technical University of Catalonia, Barcelona, 2003
- [Vázquez-Salceda, Dignum, 2003] Vázquez-Salceda, J., Dignum, F.: Modeling Electronic Institutions. In: Marik, V., Mueller, J., Pechoucek, M. (Eds.): *Proc. of Central and East European Conference in Multi-Agent Systems (CEEMAS'03)*, Prague, June 2003. LNAI 2691, Springer, 2003.
- [Venkatraman, Singh, 1999] Venkatraman, M., Singh, M.: Verifying Compliance with Commitment Protocols. *Journal of Autonomous Agents and Multi-Agent Systems* **2**(3), Kluwer Academic Publishers. 1999, pp. 217-236
- [Verharen, 1997] Verharen, E.: A Language-Action Perspective on the Design of Cooperative Information Systems, *Ph.D. thesis*, University of Tilburg, 1997.
- [Verharen, Dignum, 1997] Verharen, E., Dignum, F.: Cooperative Information Agents and Communication. In: Klusch, M. (Ed.): *Proc. of the 1st. International Workshop on Cooperative Information Agents*, LNAI 1202, Springer-Verlag, 1997.
- [von Wright, 1950] von Wright, G.: Deontic Logic. *Mind* **60**, 1950, pp. 1 – 15.
- [Weber, 1978] Weber M.: *Economy and Society*. Roth, G., Wittich, C. (Eds.). Berkeley, CA, University of California Press, 1978.
- [Webster, 1995] Webster, C.: Communication Ethics and Human-Computer Cognitive Systems. *Proc. of the 1st Int. Conf. on Cognitive Technology*, 1995, <http://edutools.cityu.edu.hk/ct1995/webster.htm>.

- [Weigand et al., 1999] Weigand, H., Hoppenbrouwers, S., de Moor, A.: The Context of Conversations – Texts and Communities, *Proc. of the Language-Action Perspective Workshop*, 1999.
- [Weigand et al., 2003] Weigand, H., Dignum, V., Meyer, J.-J., Dignum, F.: Specification by Refinement and Agreement: Designing Agent Interaction Using Landmarks and Contracts. In: Petta, P., Tolksdorf, R., Zambonelli, F. (Eds.): *Engineering Societies in the Agents World III: Proceedings ESAW'02*, LNAI 2577, Springer-Verlag, 2003, pp. 257-269.
- [Weiser, 1993] Weiser, M.: Ubiquitous Computing. *IEEE Computer Hot Topics*, October, 1993, quoted in <http://www.ubiq.com/hypertext/weiser/UbiHome.html>.
- [Weiss, 1999] Weiss, G. (Ed.): *Multi-agent Systems: a Modern Approach to Distributed Artificial Intelligence*, MIT Press, 1999.
- [Wiederhold, 1992] Wiederhold, G.: Mediators in the Architecture of Future Information Systems. *IEEE Computer Magazine*, March 1992.
- [Williamson, 1975] Williamson, O.: *Markets and hierarchies: Analysis and Antitrust Implications*. New York: Free Press, 1975.
- [Winograd, Flores, 1986] Winograd, T. Flores, F.: *Understanding Computers and Cognition: A New Foundation for Design*, Ablex, 1996
- [Wood, DeLoach, 2001] Wood, M., DeLoach, S.: An Overview of the Multi-agent Systems Engineering Methodology. In: Ciancarini P., Wooldridge, M. (Eds.): *Agent-Oriented Software Engineering*, LNCS 1957, Springer, 2001, pp. 207 - 221.
- [Wooldridge et al, 2000] Wooldridge, M., Jennings, N., Kinny D.: The Gaia Methodology for Agent-Oriented Analysis and Design. *Journal of Autonomous Agents and Multi-Agent Systems* 3(3), Kluwer, 2000, pp. 285-312.
- [Wooldridge et al., 2002] Wooldridge, M., Weiss, G., Ciancarini, P. (Eds.): *Agent-Oriented Software Engineering II. Revised Papers and Invited Contributions, AOSE 2001*. LNAI 2222, Springer-Verlag, 2002.
- [Wooldridge, 1996] Wooldridge, M.: Time, Knowledge, and Choice. In: Wooldridge, M., Mueller, J. P., Tambe, M. (Eds.): *Intelligent Agents II*, Springer-Verlag, 1996.
- [Wooldridge, 1997] Wooldridge, M.: Agent-Based Software Engineering. *IEEE Proc. Software Engineering* 144(1), 1997, pp. 26-37.
- [Wooldridge, 1999] Wooldridge, M.: Intelligent Agents. Weiss, G. (Ed): *Multi-agent Systems: a Modern Approach to Distributed Artificial Intelligence*, MIT Press, 1999
- [Wooldridge, 2000] Wooldridge, M.: *Reasoning about Rational Agents*. MIT Press, 2000.
- [Wooldridge, Ciancarini, 2001] Wooldridge, M., Ciancarini, P.: Agent-Oriented Software Engineering: The State of the Art. In: Ciancarini P., Wooldridge, M. (eds.): *Agent-Oriented Software Engineering*, LNCS 1957, Springer, 2001, pp. 1 – 28.
- [Wooldridge, Jennings, 1995] Wooldridge, M., Jennings, N.: Intelligent Agents: Theory and Practice. In: *The Knowledge Engineering Review* 10(2), 1995, pp. 115-152.
- [Wooldridge, Jennings, 1999] Wooldridge, M., Jennings, N.: Software Engineering with Agents: Pitfalls and Pratfalls, *IEEE Internet Computing* 3(3), 1999, pp. 20-27.
- [Yip, Cunningham, 2002] Yip, A., Cunningham, J.: Some Issues on Agent Ownership. In: *The Law of Electronic Agents: Selected Revised Papers*, Proceedings of LEA Workshop, CIRSIFID, University of Bologna, 2002, pp. 13 – 22.

-
- [Yu, Singh, 2002] Yu, B., Singh, M.: Emergence of Agent-Based Referral Networks, *Proc. of 1st. International Joint Conference on Autonomous Agents and Multi-Agent Systems*, AAMAS, 2002, pp. 1208-1209.
- [Zambonelli et al., 2001a] Zambonelli F., Jennings, N., Wooldridge, M.: Organisational Abstractions for the Analysis and Design of Multi-Agent Systems. In: Ciancarini P., Wooldridge, M. (eds.): *Agent-Oriented Software Engineering*, LNCS 1957, Springer-Verlag, 2001, pp. 235 – 251.
- [Zambonelli et al., 2001b] Zambonelli, F., Jennings, N., Omicini, A., & Wooldridge, M.: Agent-Oriented Software Engineering for Internet Applications. Omicini, A., Zambonelli, F., Klusch, M., & Tolkdorf, R. (eds.). *Co-ordination of Internet Agents: Models, Technologies, and Applications*, Springer, 2001, pp. 326 – 346.
- [Zambonelli, 2002] Zambonelli, F.: Abstractions and Infrastructures for the Design and Development of Mobile Agent Organizations. In: Wooldridge, M., Weiss, G., Ciancarini P. (Eds.): *Agent-Oriented Software Engineering II*, LNCS 2222, Springer-Verlag, 2002, pp. 245 – 262.

Summary

The motivation for this research came forth from the need to devise support systems for Knowledge Management (KM) that incorporate the management of knowledge assets with the facilitation and encouragement of interaction between people in an open environment. If, on the one hand, it is commonly accepted that organizations cannot achieve sustainable advantage without a concerted management of its knowledge assets and the support of creativity and innovation, on the other hand, knowledge needs vary, knowledge assets that are today extremely valuable can be tomorrow obsolete, the needs and requests from the environment are also not constant, and people have different views on how, when and why they are willing to share their knowledge or request support from others. Due to these characteristics, support systems for KM cannot be completely designed in fore hand, and must accommodate changing needs and tasks and support users with different goals and requirements. These considerations indicate that modeling techniques and frameworks for KM support systems, must be developed based on the interaction between autonomous components within the boundaries determined by common goals and behavior norms.

Three main assumptions underlie this research. Firstly, in many cases, autonomous entities need a social setting in order to realize their own individual goals. However, and this is our second assumption, organizations and/or societies have themselves global goals and requirements, which are not necessarily shared with any of the participating entities, but must be achieved by the (coordinated) activity of those individuals. Thirdly, we assume that a process of negotiation and adjustment is needed between individual and social requirements and characteristics in order to conjugate individual autonomy with social requirements and goals.

The OperA model presented in this dissertation, uses the agent paradigm as conceptual design tool. In our opinion, the concept of agents presents two powerful bases for organizational interaction because it enables both the reference to any autonomous entity participating in an interaction (including people), and provides theoretical models for entities and interaction. The OperA framework accommodates

different agent architectures and provides therefore a flexible way to represent interaction and role enactment, because it abstracts from the specific internal representations of the individual agents. OperA separates the modeling of organizational requirements and aims, from the modeling of the individual entities and provides flexible means to integrate organizational structure (the Organizational Model) with individual aims (the Social Model) and interactions (the Interaction Model). Contracts are used to link the different models and create specific instances that reflect the needs and structure of the current environment and participants. The main focus of OperA is not the design of the individual entities that form the organization, but the design of the environment where those entities interact and, such that a balance can be found between the autonomy of agents, their coordination needs and the environment expectations. Due to its level of abstraction, OperA is suitable both to model and study existing societies, as well as to develop new systems that participate in a organizational context. In our opinion, OperA can contribute to the acceptance and understanding of agents societies in organizations, because it is based on an organizational, collectivist view of the environment and uses the agent paradigm to provide a natural way to view and characterize intelligent systems.

We have enhanced the practical applicability of OperA by developing a engineering methodology that guides the design of OperA models. This methodology has been used to apply the OperA framework to different case studies covering different aspects of interaction in organized environments, ranging from KM systems to non-monetary markets for the exchange of health services. This wide range of applications demonstrates the possibilities of the concept. Furthermore, OperA enables the use of technology to support interaction and collaboration in KM environments, in ways that enrich the organization and take in account individual requirements and motivations.

Even though our primary aim was the development of practical solutions for concrete (KM) applications, it was important to develop a formal theory for the OperA framework. A language for contract representation, LCR, based on deontic temporal logic was developed. LCR is a very expressive logic for describing interaction in multi-agent systems that makes it possible to describe and verify contracts that specify interaction between agents. LCR was furthermore extended with illocution and epistemic elements, as a formal framework and integrated semantics for OperA, at all three levels of society specification (organizational, social and interaction). The formalism provides a rather realistic representation of a domain, in the sense that it treats temporal and communicative aspects and furthermore is able to represent deadlines and its influence in the behavior of the model. Moreover, the formal semantics of OperA provide a strong theoretical background and enables the verification of OperA models.

Samenvatting

De motivatie voor dit onderzoek ontstond uit de behoefte om ondersteuningsystemen voor kennismanagement (KM) te ontwikkelen die het mogelijk maken om het beheer van kennisbronnen te combineren met het faciliteren en stimuleren van interactie tussen mensen in een open omgeving. Aan de éne kant, is het gewoonlijk geaccepteerd dat organisaties niet zonder een kennis beheerplan en de ondersteuning van creativiteit en innovatie kunnen (het doel van de organisatie bij KM). Aan de andere kant kunnen de behoeften aan soort en hoeveelheid kennis verschillen per medewerker. Bovendien kan kennis die vandaag onmisbaar is, morgen niets meer waard zijn. Kennisvragen blijven veranderen, en mensen hebben hun eigen ideeën over hoe, wanneer en waarom zij hun kennis willen delen of kennis van anderen willen vragen (de autonomie van de medewerker). Mede hierdoor, kunnen KM ondersteuningsystemen niet a-priori volledig ontworpen worden, maar moeten aanpasbaar zijn aan de veranderende behoeften en taken, en ondersteuning kunnen bieden aan gebruikers met verschillende doelen en eisen. Uit het voorgaande blijkt dat ontwerptechnieken en -raamwerken voor het ontwikkelen van ondersteuning van KM moeten uitgaan van systemen voor interactie tussen autonome onderdelen binnen de grenzen van gezamenlijk doelen en gedragsnormen.

De volgende drie aannames vormen de basis voor dit onderzoek. Ten eerste hebben autonome entiteiten meestal een sociale omgeving nodig voor de realisatie van hun eigen individuele doelen. Ten tweede, hebben organisaties en maatschappijen ook hun eigen doelen en eisen, die niet dezelfde hoeven te zijn als die van de entiteiten waaruit zij bestaan. Zulke doelen kunnen echter alleen bereikt worden door de gecoördineerde actie van de deelnemers. Als laatste, gaan wij er van uit dat er een onderhandelingsproces nodig is dat zorgt voor het op een lijn brengen van individuele en sociale eisen en kenmerken, zodat individuele autonomie samen kan gaan met het bereiken van sociale eisen en doelen.

Het OperA model dat in dit proefschrift wordt ontwikkeld, maakt gebruik van het agent paradigma als onderliggend concept. Het agent paradigma bevat theoretische

modellen voor zowel individuele entiteiten (agenten) als hun onderlinge interactie. Daarom denken wij dat het agentconcept een krachtig fundament biedt voor het modelleren van interacties in de omgeving van KM. Het OperA raamwerk maakt het mogelijk om verschillende agentarchitecturen te combineren omdat het abstraheert van de inhoud van de specifieke agenten. Het levert aldus een flexibele manier op om interacties en rollen te representeren. In OperA worden organisatie-eisen en -doelen apart gemodelleerd van die van de agenten (die de rollen binnen de organisatie invullen). Dit zorgt voor een flexibele manier om organisatiestructuur (het Organisatie Model) te combineren met individuele doelen (het Sociaal Model) en interacties (het Interactie Model). Wij gebruiken contracten om de verschillende modellen te combineren en om specifieke instanties te creëren die de actuele behoeften en structuur of de huidige omgeving en deelnemers uitbeelden. Het hoofddoel van OperA is uitdrukkelijk niet het ontwerp van individuele agenten maar het ontwerp van de omgeving waar die agenten kunnen interacteren om een evenwicht te bereiken tussen het autonome gedrag van de agenten, hun coördinatiebehoeften en de verwachtingen en eisen van de omgeving. Vanwege zijn abstractieniveau is OperA zowel geschikt voor het modelleren en analyseren van bestaande organisaties als de ontwikkeling van nieuwe systemen binnen een organisatie. Aangezien OperA gebaseerd is op een organisatorisch perspectief voor het modelleren van de interacties, denken wij dat OperA kan bijdragen aan de acceptatie van agent modelleertechnieken bij het bedrijfsleven.

Wij hebben de praktische toepasbaarheid van OperA vergroot door de ontwikkeling van een ontwerpmethodologie voor het ontwerpproces. Het OperA raamwerk werd gebruikt in combinatie met deze methodologie voor het ontwikkelen van modellen voor verschillende aspecten van interacties binnen organisaties, van KM tot non-monetaire markten waarop zorgdiensten kunnen worden uitgewisseld. Deze grote verscheidenheid van onderwerpen geeft een goede indicatie van de toepasbaarheid van het concept. In KM omgevingen maakt het OperA model het mogelijk om automatiseringssystemen optimaal te richten op de behoeften van zowel de gebruikers als de organisatie in zijn totaliteit.

Hoewel het hoofddoel van dit onderzoek het ontwikkelen van praktische oplossingen voor concrete KM applicaties was, was het van groot belang om een formele theorie te ontwikkelen die het OperA model een stevig fundament kon geven. Wij hebben hiervoor een formele taal voor contract specificatie, LCR, ontworpen. LCR is een expressieve taal voor het beschrijven van interacties in multi-agent systemen. Door de toevoeging aan LCR van formele concepten voor communicatie, was het mogelijk om LCR te gebruiken als formele semantiek van OperA op alle drie de specificatieniveaus (organisatie, sociaal en interactie). Door de combinatie van temporele, deontische en communicatieve aspecten in LCR is een realistische representatie van het domein mogelijk en kunnen de OperA modellen ook geverifieerd worden.

Sumário

A motivação para a investigação descrita nesta tese surgiu da necessidade de desenvolver sistemas de apoio para a gestão de conhecimento (KM) que combinem a gestão de recursos com a facilitação e encorajamento de interacção entre pessoas, num ambiente aberto. Se, por um lado, é geralmente aceite que empresas e organizações não conseguem sobreviver sem um plano concertado de gestão de conhecimentos e de apoio à criatividade e inovação, pelo outro lado as necessidades individuais de conhecimento são altamente variáveis, recursos que hoje são indispensáveis, tornam-se amanhã obsoletos, e cada pessoa tem as suas próprias ideias sobre a razão, momento e maneira de pedir ou fornecer os seus próprios conhecimentos e experiências a outros. Estas características indicam que o desenvolvimento completo à priori de sistemas de apoio para KM não é possível, mas deve ser adaptável duma maneira dinâmica às várias necessidades e tarefas dos utilizadores, e mais, possibilitar o uso desses sistemas por utilizadores com objectivos e requisitos diferentes. Daqui se depreende que técnicas e estruturas de desenvolvimento de sistemas de apoio para KM, devem partir de modelos de interacção entre componentes autónomos, cujos limites são definidos por objectivos e normas de comportamento comuns.

Esta investigação baseia-se em três suposições principais. Em primeiro lugar, entidades autónomas necessitam de um ambiente social para a realização dos seus próprios objectivos. Em segundo lugar, sociedades em geral e organizações em particular, têm os seus próprios objectivos e requisitos, que só são possíveis de atingir através da acção coordenada entre os seus participantes, mas que não são necessariamente partilhados por estes. Finalmente, assumimos que a conjugação entre a autonomia individual e as características sociais pode ser atingida através dum processo de ajustamento e negociação entre requisitos individuais e sociais.

O modelo OperA apresentado nesta tese baseia-se no paradigma de agentes como ferramenta de desenho conceitual. Este paradigma dá-nos duas bases fortes para interacção social, devido ao facto de que, por um lado, possibilita a referência

abstracta a qualquer tipo de entidade participante em interações (incluindo pessoas) e, por outro lado, fornece modelos teóricos tanto para indivíduos como para interações. Dado que o modelo distingue entre o conceito de agente e o papel representado pelo agente numa organização, e abstrai dos aspectos específicos da estrutura interna dos agentes, é possível combinar numa estrutura OperA diferentes arquiteturas de agentes numa maneira muito flexível, o que possibilita a descrição de sociedades abertas. OperA separa a especificação das estruturas sociais (o Modelo de Organização) da descrição de objectivos individuais (o Modelo Social) e da descrição das interações (o Modelo de Interação) e usa o conceito de contrato para combinar estes modelos e especificar instâncias específicas que reflectem as necessidades concretas do ambiente e participantes correntes. Ou seja, o foco de OperA não é a representação em si das entidades individuais, mas o desenho de ambientes em que essas entidades possam interagir numa maneira que combina a sua autonomia com as necessidades e esperanças da organização. Devido ao nível de abstracção de OperA, o modelo é apropriado tanto para a modelação e estudo de organizações existentes, como para o desenvolvimento de sistemas de apoio em contextos organizationais. Devido ao facto que o desenvolvimento do modelo OperA é baseado numa visão social e colectiva de organizações, equivalente a ideias usadas em ciências sociais e de gestão, estamos convencidos que OperA pode contribuir para o aceiteamento de sistemas de agentes em organizações como uma maneira natural para a representação e caracterização de sistemas complexos e inteligentes.

A possibilidade de aplicação prática de OperA é demonstrada com o desenvolvimento de uma metodologia de desenho para modelos OperA. Esta metodologia foi aplicada no desenvolvimento de vários casos de estudo, cobrindo aspectos diferentes de interação em ambientes organizados, desde a área de KM até a modelos para mercados não-monetários para troca de serviços de saúde. As possibilidades do modelo e conceitos desenvolvidos nesta tese é demonstrada por esta variedade de aplicações. Na área específica de KM, OperA possibilita o uso de tecnologia no apoio da colaboração entre utilizadores, de maneiras que contribuem para o enriquecimento da organização e levam em conta as motivações e requisitos individuais.

Se bem que o objectivo principal desta investigação tenha sido o desenvolvimento de aplicações práticas para apoio à gestão de conhecimentos, o desenvolvimento de uma teoria formal para o modelo OperA foi de grande importância. A Linguagem de Representação de Contratos (LCR) baseada em lógicas temporais e deónticas que apresentamos, é suficientemente expressiva para descrever sistemas de multi-agentes, pois torna possível a descrição e verificação de contractos de interação entre agentes. A linguagem LCR foi também estendida com elementos ilocucionários que permitem a formalização da comunicação e fornece assim uma estrutura formal e uma semântica integrada para modelos OperA a todos os níveis (organização, social e interação). Devido a este facto, o formalismo LCR possibilita a representação de domínios realistas e possibilita a verificação formal da aplicação de OperA a estes domínios.

Curriculum Vitae

Virginia Dignum

- 2 May 1964:** Born in Lisbon, Portugal.
- 1976 – 1982:** High school studies, Escola Secundária Sebastião e Silva, Oeiras, Portugal.
- 1982 – 1987:** Study Applied Mathematics and Computer Science, Science Faculty, University of Lisbon, Portugal.
- 1987 – 1989:** Study Computer Science, Faculty Mathematics and Computer Science, Free University Amsterdam, The Netherlands.
- 1989 – 1994:** Freelance junior assistant professor, trainer, system designer, scientific programmer, Swaziland, Portugal, The Netherlands.
- 1994 – 1995:** Scientific Programmer, Algorithmic Research, Nuenen, The Netherlands.
- 1995 – 1997:** Information analyst, Origin, Eindhoven, The Netherlands.
- 1997 – 1999:** Senior Consultant, Arthur Andersen, Advanced Knowledge Technology Group, Rotterdam, The Netherlands.
- 1999 – 2003:** Researcher, Achmea Holding N.V., Zeist, The Netherlands.
Ph.D. student at the Institute for Information and Computer Science, Faculty of Mathematics and Computer Science, Utrecht University, The Netherlands.

SIKS Dissertation Series

1998

- 1998-1 Johan van den Akker (CWI): *DEGAS - An Active, Temporal Database of Autonomous Objects*
- 1998-2 Floris Wiesman (UM): *Information Retrieval by Graphically Browsing Meta-Information*
- 1998-3 Ans Steuten (TUD): *A Contribution to the Linguistic Analysis of Business Conversations within the Language/Action Perspective*
- 1998-4 Dennis Breuker (UM): *Memory versus Search in Games*

1999

- 1999-1 Mark Sloof (VU): *Physiology of Quality Change Modelling; Automated modelling of Quality Change of Agricultural Products*
- 1999-2 Rob Potharst (EUR): *Classification using decision trees and neural nets*
- 1999-3 Don Beal (UM): *The Nature of Minimax Search*
- 1999-4 Jacques Penders (UM): *The practical Art of Moving Physical Objects*
- 1999-5 Aldo de Moor (KUB): *Empowering Communities: A Method for the Legitimate User-Driven Specification of Network Information Systems*
- 1999-6 Niek J.E. Wijngaards (VU): *Re-design of compositional systems*
- 1999-7 David Spelt (UT): *Verification support for object database design*
- 1999-8 Jacques H.J. Lenting (UM): *Informed Gambling: Conception and Analysis of a Multi-Agent Mechanism for Discrete Reallocation.*

2000

- 2000-1 Frank Niessink (VU): *Perspectives on Improving Software Maintenance*
- 2000-2 Koen Holtman (TUE): *Prototyping of CMS Storage Management*

- 2000-3 Carolien M.T. Metselaar (UVA): *Sociaal-organisatorische gevolgen van kennistechnologie; een procesbenadering en actorperspectief.*
- 2000-4 Geert de Haan (VU): *ETAG, A Formal Model of Competence Knowledge for User Interface Design*
- 2000-5 Ruud van der Pol (UM): *Knowledge-based Query Formulation in Information Retrieval.*
- 2000-6 Rogier van Eijk (UU): *Programming Languages for Agent Communication*
- 2000-7 Niels Peek (UU): *Decision-theoretic Planning of Clinical Patient Management*
- 2000-8 Veerle Coupé (EUR): *Sensitivity Analysis of Decision-Theoretic Networks*
- 2000-9 Florian Waas (CWI): *Principles of Probabilistic Query Optimization*
- 2000-10 Niels Nes (CWI): *Image Database Management System Design Considerations, Algorithms and Architecture*
- 2000-11 Jonas Karlsson (CWI): *Scalable Distributed Data Structures for Database Management*

2001

- 2001-1 Silja Renooij (UU): *Qualitative Approaches to Quantifying Probabilistic Networks*
- 2001-2 Koen Hindriks (UU): *Agent Programming Languages: Programming with Mental Models*
- 2001-3 Maarten van Someren (UvA): *Learning as problem solving*
- 2001-4 Evgueni Smirnov (UM): *Conjunctive and Disjunctive Version Spaces with Instance-Based Boundary Sets*
- 2001-5 Jacco van Ossenbruggen (VU): *Processing Structured Hypermedia: A Matter of Style*
- 2001-6 Martijn van Welie (VU): *Task-based User Interface Design*
- 2001-7 Bastiaan Schonhage (VU): *Diva: Architectural Perspectives on Information Visualization*
- 2001-8 Pascal van Eck (VU): *A Compositional Semantic Structure for Multi-Agent Systems Dynamics.*
- 2001-9 Pieter Jan 't Hoen (RUL): *Towards Distributed Development of Large Object-Oriented Models, Views of Packages as Classes*
- 2001-10 Maarten Sierhuis (UvA): *Modeling and Simulating Work Practice. BRAHMS: a multiagent modeling and simulation language for work practice analysis and design*
- 2001-11 Tom M. van Engers (VUA): *Knowledge Management: The Role of Mental Models in Business Systems Design*

2002

- 2002-01 Nico Lassing (VU): *Architecture-Level Modifiability Analysis*
- 2002-02 Roelof van Zwol (UT): *Modelling and searching web-based document collections*
- 2002-03 Henk Ernst Blok (UT): *Database Optimization Aspects for Information Retrieval*
- 2002-04 Juan Roberto Castelo Valdueza (UU): *The Discrete Acyclic Digraph Markov Model in Data Mining*
- 2002-05 Radu Serban (VU): *The Private Cyberspace Modeling Electronic Environments inhabited by Privacy-concerned Agents*

- 2002-06 Laurens Mommers (UL): *Applied legal epistemology; Building a knowledge-based ontology of the legal domain*
- 2002-07 Peter Boncz (CWI): *Monet: A Next-Generation DBMS Kernel For Query-Intensive Applications*
- 2002-08 Jaap Gordijn (VU): *Value Based Requirements Engineering: Exploring Innovative E-Commerce Ideas*
- 2002-09 Willem-Jan van den Heuvel (KUB): *Integrating Modern Business Applications with Objectified Legacy Systems*
- 2002-10 Brian Sheppard (UM): *Towards Perfect Play of Scrabble*
- 2002-11 Wouter C.A. Wijngaards (VU): *Agent Based Modelling of Dynamics: Biological and Organisational Applications*
- 2002-12 Albrecht Schmidt (UVA): *Processing XML in Database Systems*
- 2002-13 Hongjing Wu (TUE): *A Reference Architecture for Adaptive Hypermedia Applications*
- 2002-14 Wieke de Vries (UU): *Agent Interaction: Abstract Approaches to Modelling, Programming and Verifying Multi-Agent Systems*
- 2002-15 Rik Eshuis (UT): *Semantics and Verification of UML Activity Diagrams for Workflow Modelling*
- 2002-16 Pieter van Langen (VU): *The Anatomy of Design: Foundations, Models and Applications*
- 2002-17 Stefan Manegold (UVA): *Understanding, Modeling, and Improving Main-Memory Database Performance*

2003

- 2003-01 Heiner Stuckenschmidt (VU): *Ontology-Based Information Sharing in Weakly Structured Environments*
- 2003-02 Jan Broersen (VU): *Modal Action Logics for Reasoning About Reactive Systems*
- 2003-03 Martijn Schuemie (TUD): *Human-Computer Interaction and Presence in Virtual Reality Exposure Therapy*
- 2003-04 Milan Petkovic (UT): *Content-Based Video Retrieval Supported by Database Technology*
- 2003-05 Jos Lehmann (UVA): *Causation in Artificial Intelligence and Law - A modelling approach*
- 2003-06 Boris van Schooten (UT): *Development and specification of virtual environments*
- 2003-07 Machiel Jansen (UvA): *Formal Explorations of Knowledge Intensive Tasks*
- 2003-08 Yongping Ran (UM): *Repair Based Scheduling*
- 2003-09 Rens Kortmann (UM): *The resolution of visually guided behavior*
- 2003-10 Andreas Lincke (UvT): *Electronic Business Negotiation: Some experimental studies on the interaction between medium, innovation context and culture*
- 2003-11 Simon Keizer (UT): *Reasoning under Uncertainty in Natural Language Dialogue using Bayesian Networks*
- 2003-12 Roeland Ordelman (UT): *Dutch speech recognition in multimedia information retrieval*
- 2003-13 Jeroen Donkers (UM): *Nosce Hostem - Searching with Opponent Models*

- 2003-14 Stijn Hoppenbrouwers (KUN): *Freezing Language: Conceptualisation Processes across ICT-Supported Organisations*
- 2003-15 Mathijs de Weerd (TUD): *Plan Merging in Multi-Agent Systems*
- 2003-16 Menzo Windhouwer (CWI): *Feature Grammar Systems - Incremental Maintenance of Indexes to Digital Media Warehouses*
- 2003-17 David Jansen (UT): *Extensions of Statecharts with Probability, Time, and Stochastic Timing*
- 2003-18 Levente Kocsis (UM): *Learning Search Decisions*