Lecture Notes in Artificial Intelligence 2586

Subseries of Lecture Notes in Computer Science
Edited by J. G. Carbonell and J. Siekmann

Lecture Notes in Computer Science
Edited by G. Goos, J. Hartmanis, and J. van Leeuwen

Springer
Berlin
Heidelberg
New York
Barcelona
Hong Kong
London
Milan
Paris
Tokyo

Matthias Klusch Sonia Bergamaschi
Pete Edwards Paolo Petta (Eds.)

Intelligent Information Agents

The AgentLink Perspective

Springer

Series Editors

Jaime G. Carbonell, Carnegie Mellon University, Pittsburgh, PA, USA
Jörg Siekmann, University of Saarland, Saarbrücken, Germany

Volume Editors

Matthias Klusch
German Research Center for Artifical Intelligence
Multiagent Systems Group
Stuhlsatzenhausweg 3, 66123 Saarbrücken, Germany
E-mail: klusch@dfki.de

Sonia Bergamaschi
Università di Modena e Reggio Emilia
Dipartimento di Ingegneria dell'Informazione
Via Vignolese 905, 41100 Modena, Italy
E-mail: sonia.bergamaschi@unimo.it

Pete Edwards
University of Aberdeen, King's College
Department of Computing Science
Aberdeen AB24 5UE, UK
pedwards@csd.abdn.ac.uk

Paolo Petta
Austrian Research Institute for Artificial Intelligence
Schottengasse 3, 1010 Vienna, Austria
Email: paolo@oefai.at

Cataloging-in-Publication Data applied for

A catalog record for this book is available from the Library of Congress.

Bibliographic information published by Die Deutsche Bibliothek
Die Deutsche Bibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data is available in the Internet at <http://dnb.ddb.de>.
CR Subject Classification (1998): I.2.11, C.2.4, D.2.12, H.2, H.3, H.4, K.4.4

ISSN 0302-9743
ISBN 3-540-00759-8 Springer-Verlag Berlin Heidelberg New York

Springer-Verlag Berlin Heidelberg New York
a member of BertelsmannSpringer Science+Business Media GmbH

http://www.springer.de

Printed in Germany

Typesetting: Camera-ready by author, data conversion by PTP-Berlin, Stefan Sossna e. K.
Printed on acid-free paper SPIN: 10872433 06/3142 5 4 3 2 1 0

Table of Contents

European Research and Development of Intelligent Information Agents: The AgentLink Perspective

Matthias Klusch[1], Sonia Bergamaschi[2], and Paolo Petta[3]

[1] German Research Center for Artifical Intelligence, Multiagent Systems Group, Stuhlsatzenhausweg 3, D-66123 Saarbrücken, Germany.
klusch@dfki.de

[2] Dipartimento Ingegneria dell'Informazione, University of Modena and Reggio Emilia, via Vignolese 905, Modena, Italy.
sonia.bergamaschi@unimo.it

[3] Austrian Research Institute for Artificial Intelligence, Schottengasse 3, A 1010 Vienna, Austria.
paolo@oefai.at

1 Introduction

The vast amount of heterogeneous information sources available in the Internet demands advanced solutions for acquiring, mediating, and maintaining relevant information for the common user. The impacts of data, system, and semantic heterogeneity on the information overload of the user are manifold and especially due to potentially significant differences in data modeling, data structures, content representations using ontologies and vocabularies, query languages and operations to retrieve, extract, and analyse information in the appropriate context. The impacts of the increasing globalisation on the information overload encompass the tedious tasks of the user to determine and keep track of relevant information sources, to efficiently deal with different levels of abstractions of information modeling at sources, and to combine partially relevant information from potentially billions of sources. A special type of intelligent software agents [92,36,95], so called information agents, is supposed to cope with these difficulties associated with the information overload of the user. This implies its ability to semantically broker information [82] by providing pro-active resource discovery, resolving the information impedance of information consumers and providers in the Internet, and offering value-added information services and products to the user or other agents. In subsequent sections we briefly introduce the reader to the notion of such type of agents as well as one of the prominent European forums for research on and development of these agents, the AgentLink special interest group on intelligent information agents. This book includes presentations of advanced systems of information agents and solution approaches to different problems in the domain that have been developed jointly by members of this special interest group in respective working groups.

M. Klusch et al. (Eds.): Intelligent Information Agents, LNAI 2586, pp. 1–21, 2003.

1.1 Intelligent Information Agents in Brief

An *information agent* is an autonomous, computational software entity that has access to one or more heterogeneous and geographically distributed information sources, and which is able to pro-actively acquire, mediate, and maintain relevant information on behalf of its users or other agents preferably just-in-time. Information agents may be classified according to one or more of the following features.

1. *Non-cooperative* or *cooperative information agents*, depending on the ability to cooperate with each other for the execution of their tasks. Several protocols and methods are available for achieving cooperation among autonomous information agents in different scenarios, like hierarchical task delegation, contracting, and decentralised negotiation.
2. *Adaptive information agents* are able to adapt themselves to changes in networks and information environments. Examples of such agents are learning personal assistants on the Web.
3. *Rational information agents* behave in a utilitarian way in an economic sense. They act, and may even collaborate, to increase their own benefits. The main application domains of such kinds of agents are automated trading and electronic commerce in the Internet. Examples include the variety of shop bots, and systems for agent-mediated auctions on the Web.
4. *Mobile information agents* are able to travel autonomously through the Internet. Such agents enable dynamic load balancing in large-scale networks, reduction of data transfer among information servers, and migration of small business logic within medium-range corporate intranets on demand.

Regarding the basic skills of information agents, we can differentiate between communication, knowledge, collaboration, and rather low-level task skills as shown in figure 1. The most prominent key supporting technologies for the implementation of information agents are listed below each of their skills.

Communication skills of an information agent include either communication with information systems and databases, human users, or other agents. In the latter case, the use of a commonly agreed upon agent communication language (ACL) such as FIPA ACL and KQML [47] has to be considered on top of, for example, middleware platforms or specific APIs. Representation of and processing ontological knowledge and metadata, profiles and natural language input, translation of data formats as well as the application of machine learning techniques [58], enable an information agent to acquire and maintain knowledge about itself and its environment. High-level collaboration with other agents can rely, for example, on suitable coordination techniques [65] such as service brokering, matchmaking, negotiation, and collaborative (social) filtering, whereas collaborating with human users mainly corresponds to the application of techniques stemming from the domain of human-agent interaction (HAI) [55,4] and affective computing (AC) [73,76,89]. Many (systems of) information agents have been developed or are currently under development in academic and commercial research labs such as IMPACT [84], RETSINA [85], InfoSleuth [59], ARIADNE

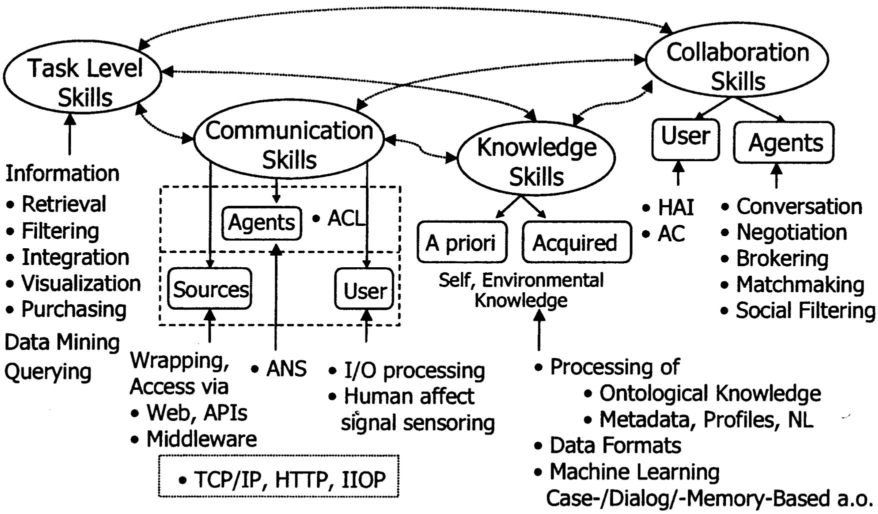

Fig. 1. Classes of information agent skills and key supporting technologies

and KRAFT (in this volume), but most of them still have not made it out to the real world of Internet users broadly.

Among the most prominent forums for the exchange and dissemination of topical research and development of information agents we name, for example, the international CIA workshop series on cooperative information agents [44], the conferences on advances in agent technology such as Autonomous Agents and Multi-Agent Systems (AAMAS), and cooperative information systems (COOPIS) [68], and last but not least the European special interest group on intelligent information agents (I2A SIG) as part of the AgentLink network of excellence for agent-based computing [1].

1.2 The European AgentLink SIG on Intelligent Information Agents

The European AgentLink special interest group on intelligent information agents (I2A SIG) [28] has been founded in 1998 as part of the EU-funded network of excellence for agent-based computing, named AgentLink, in the context of the research framework program ESPRIT (project 27.225, 1998 - 2000), and continued its work in AgentLink II funded by the EU in the FP5 IST program (project IST-1999-29003, 2000 - 2003). The I2A SIG has been co-ordinated by Matthias Klusch from the German Research Center for Artificial Intelligence (DFKI) since 1998, jointly with Sonia Bergamaschi from University of Bologna, Italy, since July 2001.

The mission of the AgentLink I2A SIG is to promote advanced research on and development of intelligent information agents across Europe. Since its foundation the SIG provides a strong and visible forum for the presentation, exchange, and cross fertilisation of ideas and work performed at nodes of the AgentLink network, and beyond. It helped to foster existing contacts and to establish numerous collaborations among academic researchers and different business organisations; the results of some of these collaborations between members of the SIG are presented in this book. In correspondence with the most prominent domains of topics within the R&D area of information agents, members of the SIG constituted the following working groups.

– *Working group AIA on adaptive information agents.*
  Adaptive information agent systems employ learning techniques to adapt to one or all of the following: users, agents, the environment. Examples include personal assistants for information searches on the Web or collaborating information agents that adapt themselves as a system in changing environments. The AIA group initially did focus on a small number of key areas. The first of these to be identified were: adaptive user-interfaces, user-profiling and personalisation technologies. Further, it explored beneficial links with MLNet (European Network of Excellence in Machine Learning), and EUNITE (European Network of Excellence on Intelligent Technologies for Smart Adaptive Systems), and identified industrial contacts with interests in AIA related technology. The AIA group is concerned with all aspects of adaptation and learning in the context of intelligent information agents. It was active from December 2000 until end of 2001, and was chaired by Pete Edwards from the university of Aberdeen, UK.
– *Working group C3 on communication, coordination, and collaboration of information agents.*
  Among others, distinguishing features of intelligent information agents are their capabilities to communicate, and coordinate their activities, and to collaborate on the distributed solution of tasks. This working group addresses topics from these three neighbouring areas, setting out from a central focus on coordination and identifying requirements for both the capabilities of individual agents but also infrastructural needs. The C3 group fosters specific thematic collaborations between academia, research groups, and industry so as to improve the current state of the art and identify novel application areas. This group has been founded in 1999 and since then is chaired by Paolo Petta from the Austrian Research Institute for Artificial Intelligence.
– *Working group AIS on agent-based information systems.*
  This group was founded in 1999 and initially focussed on topics in the domain of agent-based data and knowledge management. It was chaired by Dieter Fensel from the Free University of Amsterdam, The Netherlands, until the group significantly helped to set up and then almost completely joined the European Thematic Network 'OntoWeb' in 2000. Since 2001 the focus of the group changed to topics related to the development of agent-based information systems, and the AIS group is now chaired by Sonia Bergamaschi from the University of Bologna, Italy.

For more information on activities and regular meetings of this special interest group we encourage the reader to visit the SIG's homepage in the Web at http://www.dbgroup.unimo.it/IIA/index.html The following sections present summaries of the current state of the art in Europe in the domains of the currently active working groups of the I2A SIG: agent-based information systems, as well as communication, coordination, and collaboration. These summaries are provided by the chairs of the corresponding working groups, and hence are reflecting the AgentLink perspective. The contributions by members of all working groups to this book are then described in the final section 5.

2 R&D on Agent-Based Information Systems in Europe

With the recent advances in the development of networking infrastructures, we are witnessing today an exponential growth of distributed information for the fulfilment of user requests. Traditional approaches to manage large amounts of information that rely on distributed or federated systems [81] do not scale well to the large, diverse, growing number of information sources. Autonomous repositories distributed over the Internet that store different types of digital data with multiple formats and terminology constitute a global information infrastructure difficult to rule over. The resulting information scenario makes it impossible for users to be aware of the localisations, organisation/structure, query languages and semantics of the data in various repositories. Thus, users have more and more troubles in coping with such great amount of data to retrieve the information useful for their aims. A promising approach to this problem is to provide access to information sources by means of agent-based information systems.

An agent-based information system is a network of *information agents*, each of which provides information and expertise on a specific topic by drawing on relevant information from other information sources. More precisely, an information agent is expected to [41]:

– effectively and efficiently manage the local information it is assigned to;
– cooperate on demand with other agents for information delivery and request;
– reason about its own and other agents' capabilities, and the environment;
– actively search for relevant information on the behalf of its human users, or other agents.

On the other hand, an *agent-based information system* should guarantee a good level of balance between the independence of local information sources and the exploitation of their knowledge in a collaborative environment for the fulfilment of user requests. To this purpose languages, methods, and tools for knowledge representation and exchange have been developed. Most proposals concerns common ontologies (e.g. DAML+OIL [35,32]) and agent communication languages such as FIPA ACL [34] and KQML [33]. The key challenge of an agent-oriented information system is to be able to retrieve and integrate data from heterogeneous sources in order to deliver the required information. The extraction of the knowledge represented in information sources is thus the first step towards the information delivery process.

Information extraction usually relies on dedicated information agents named *wrappers*. Existing information sources, from databases to semi-structured and unstructured sources, can be wrapped by information agents capable of representing the source schema according to some defined common formalism. This problem has been widely investigated in literature, e.g. in [7,90,79,21]. Representing the content of heterogeneous information sources with a common formalism allows for processing and manipulating this information as a whole, trying to infer the relationships among the extracted data. The goal is creating information at higher level of abstractions with enriched semantics. To this end, a very popular approach was proposed by Wiederhold in 1992 [93]. The task of integrating information sources is entrusted to a *mediator*. The main purpose of a mediator is to gather information for information-based cooperation: it delivers on demand information by means of query processing and it actively requests information to other mediators or wrappers on behalf of its users. Most implemented mediator-based systems, such as SIMS [6], TSIMMIS [48], GARLIC [78], and MOMIS [10, 13], rely on one central mediator collaborating with multiple wrappers placed over information sources. As presented in the following, mediator-based systems designed and implemented with agent technology accounts for more flexible and distributed architectures, capable of coping with the continuously changing and growing environment of information networks.

The growing interest in information gathering and delivery based on agent technology urges the European Union to fund projects facing issues related to this field. The EU strategy - as implemented by the IST programme in the Fifth Framework Programme (FP5 1998-2002) - is to pursue the development and adoption of scientific knowledge and industrial expertise to create ubiquitous and friendly Information Society Technologies. The chosen multiple focus for this effort includes semantic-based information management, intelligent agents technologies and applications, in order to master multimedia information flows, with special reference to making the Web machine understandable. All these are coordinated through the so-called Information Access and Filtering (IAF) Action Line, which wants to “develop advanced and generic tools and techniques for the management of multimedia content to empower the user to select, receive and manipulate (in a manner that respects the user’s right to privacy) only the information required when faced with an ever increasing range of heterogeneous sources. The main focus of research is to improve the “middleware” representation, management and delivery functions of multimedia systems, for seamless content delivery, provision and access for different applications and across different media.” (see [37]). IAF also aims at “...improvements in the key functionality of large-scale multimedia asset management systems (including the evolution of the World Wide Web) to support the cost effective delivery of information services and their usage.” The activities are organised in a range of formats going from Research and Technology Development (RTD) projects, to the development of essential technologies and infrastructures, to future and emerging technologies explorations, take-up measures, and other support activities.

In the following, we briefly summarise the main FP5 IST projects related to agent-based information gathering and delivery. Some of these projects do not properly rely on agent technologies but they consider distributed organisations and address the problem of knowledge representation, sharing and interchange in this context.

KOD

The Knowledge on Demand Project, KOD (IST-1999-12503) [45], has designed and is building a multi-role personalised learning platform using collaborative software agents as the integrating components. Different technologies, based on existing Learning Technology interoperability specifications are used for other aspects of the personalisation process.

The main thrust of the KOD Project in relation to agent technologies has been the application of existing agent toolkits to a multi-role information brokerage system rather than development of agent technology itself. To this effect they have used the FIPA-compliant JADE toolkit [9] and have focussed on the development of the agent behaviour.

IBROW

The main goal of IBROW (IST-1999-19005) [12] is to develop an intelligent brokering service able to retrieve knowledge components from distributed digital libraries, according to stated user requirements. The services will go beyond simple component retrieval and will include dynamic configuration of distributed, heterogeneous applications out of pre-existing components retrieved from different libraries. The components concerned are problem solving methods (generic algorithms) and ontologies. This service will provide software-controlled access to a wide range of distributed and heterogeneous digital libraries of reusable knowledge components, at a level, which abstracts from the underlying technology. IBROW started in 2000 and will end in 2003.

Mission

The main goal of the Mission project (IST-1999-10655) [11] is to enable providers of official statistics to publish their data and to allow consumers of statistics to access these data in an informed manner with minimum effort. To this end, they adopt the World Wide Web and emerging agent-based technologies as development environment. More precisely, the reference architecture consists of five basic units. Besides the client, such an architecture provides for a library containing statistical metadata, a compute server which is a statistical processing engine for statistic delivery, a data server giving access to data and agents which perform intermediate processing and navigate the Internet to access the appropriate building blocks of the system. Mission started in 1999 and ended in 2002.

OnToKnowledge

The OnToKnowledge project (IST-1999-10132) is under way and planned to run from 1999 to 2002 [31]. The OnToKnowledge project will develop tools and methods for supporting knowledge management in large and distributed organisations. The technical backbone of On-To-Knowledge is the use of ontologies for the various tasks of information integration and mediation. Ontologies can be used to explicitly represent semantics of semi-structured information in order to enable sophisticated automatic support of acquiring, maintaining and accessing information. The goal of the OnToKnowledge project is to support efficient and effective knowledge management, mainly focusing on weakly-structured online information sources.

Onto-Logging

ONTO-LOGGING (IST-2000-28293) is under way and planned to run from 2001 to 2003 [31]. It brings together the expertise of three industrial companies in the areas of knowledge management systems, intelligent agents, and user interface and two research organisations in the areas of ontology formalisation and user modelling in order to build a set of tools on which to base the next generation distributed knowledge management systems. ONTO-LOGGING follows four main lines of action: developing a distributed ontology formalisation system, enhancing the customisation possibilities of current KM products, developing and incorporating metrics and recent developments in intelligent agent technology for user modelling and intelligent category extraction. Its main objective is the development of a set of tools to pave the way for next generation of distributed Knowledge Management systems, able to support different independent ontologies and knowledge bases.

SEWASIE

The SEWASIE project (IST-2001-34825) has just started and planned to run from 2001 to 2003 [80]. Its ambitious objective is to design and implement an advanced search engine enabling intelligent access to heterogeneous data sources on the web via semantic enrichment in order to provide the basis of structured secure web-based communication. More precisely, it will develop a distributed agent-based architecture of semantic search and communication using community-specific multilingual ontologies. SEWASIE mainly turns to small and medium enterprises (SMEs) and the developed prototypes aim at meeting their needs of enabling technology uptake and supporting stronger stand of SMEs against large competitors in a EU context. In order to meet these objectives, SEWASIE will

- develop an architecture for the proposed distributed agent-based system for semantic search (based on multilingual ontologies) and for structured secure web-based communication (for electronic negotiations);

- provide semantic enrichment processes for knowledge-based extraction of meta-information of the heterogeneous data sources;
- provide a logical layer based on Description Logics properly grounded in W3C standard such as RDF(S), XML, XML Schema;
- provide the user with efficient interfaces for formulating a query using a graphical representation and for intelligent navigation through the semantically enriched information space;
- answer user agent queries in a distributed setting that includes merging and compiling sets of queries and accessing the useful sources of information.

Silk

The Silk project (IST-1999-11135) is under way and planned to run from 2000 to 2002 [38]. The Silk project aims at integrating heterogeneous sources available in Intranet and Internet using knowledge management approaches. Integration is meant as a smoothly process, hiding the technical difficulties caused by the heterogeneous nature of data. The adopted implementation paradigms rely on logic programming and intelligent agent technology. In particular agent-oriented architecture shall be the basis of internal and external interfaces to the specific information sources.

3 R&D on Inter-agent Communication, Coordination, and Collaboration in Europe

When the working group on Communication, Coordination and Collaboration (C3) was founded in late 1998, a first challenge was to lay out the ground and define priorities within this vast territory. A straightforward constraint was to focus on complements to activities being already carried out by other working groups and other special interest groups, notably the ones on Agent Oriented Software Engineering [20] and Agent Mediated Electronic Commerce [26]. Prominent topics such as negotiation-based coordination (e.g., [62]) and their applications to e.g., agent-based distributed resources management, agent-based electronic commerce, and Virtual Enterprises Formation — as in the recent EU Projects MACIV, ForEV, or SMACE — therefore remained largely disregarded.

Selected key topics of the C3 group include advanced methods and standards for agent communication; coordination models and reference scenarios for collaborative information agents; quality of service in communication and collaboration; basic shared ontologies for communication, coordination and collaboration; and identification of infrastructural requirements and development of engineering methods. Throughout its existence, the efforts of the working group were aimed not only at identifying and fostering research on specific topics — at AgentLink meetings and via the organisation of the new workshop series, *"Engineering Societies in the Agent's World"* (ESAW) [63,64,72] and *"From Agent Theory to Agent Implementation"* (AT2AI) [69,70], but — as importantly — to induce and encourage new co-operations between researchers of different institutions, as a fundamental requirement to ensure lasting impact of the working group's activities. The selection of papers contributed to the present volume is but a short sample of evidence of the success in this latter direction, as all of these research efforts resulted from discussions and joint efforts made possible by the platform provided by the C3 working group.

Generally, there is consensus within the working group that since the publication of the reference work, "Foundations of Distributed Artificial Intelligence" [61], the state of the art in the field overall and with respect to C3 in particular has improved not so much with respect to the foundations but rather with respect to moving towards perfecting the art: slowly but surely, agents have found their way into practice and thereby have provided an essential additional source to improve the understanding of the true issues concerning coordination, communication and collaboration. This progress from theory into practice is exemplified by attempts to overcome various weaknesses diagnosed with the first attempts to provide generic agent communication infrastructures [83,47,74], witness e.g. (but certainly not only) the ongoing efforts carried out within and around FIPA and the integration of interaction and social structures into (Agent-)UML [27,14,3,8, 49]. At the same time, there is a steady progression in architectural extent from individual, "heavy" agents over multi-agent systems to open agent architectures or multi-MAS [86], complemented by research highlighting the relevance and importance of the environment in the support of phenomena such as stigmergy [97].

Opinions on the relations between communication, coordination, and collaboration differ among various communities and disciplines involved with them, including AI, CSCW, or distributed system research; e.g., whether one concept enables another (see chapters 2 and 4 in [67] and [65] for descriptions of various views on coordination, as well as chapter 8 of the present volume). One view is to understand coordination as enabling interaction, by either deciding about execution orders in programs, or empowerment of communication amongst components by interoperability (see [56]). One can also state that coordination ensures the outcome of a collaboration to be the desired result [40], thereby defining coordination as contributing to the quality of the collaboration but not necessarily enabling for communication. Although these individual models each are able to explain important characteristics of collaboration, coordination and communication, the definitions used are often in conflict.

As regards *communication*, isolated efforts such as KAoS [15] have always been placing a core emphasis on the role of dialogue protocols for the specification of agent communication, and the revised and extended current de-facto specification of KQML [46] also tried to address this topic. Even so, there still seems to be a lack of appreciation of the actual extent of the impact caused by the shift of perspective that views communication primarily as a form of interaction. The new mindset deviates from the initial research approach that was entirely dominated by references to work of computational linguists on human intention recognition, with the introduction of collaboration protocols and a high-level coordination framework on top of the ACL (see e.g. [91]).

Coordination is an interdisciplinary issue covering any area where individuals interact so as to form a society [51]. The loose definition of coordination in general covers way more than "just" a generic communication language, but encompasses in particular also coordination and cooperation. As a result of applying this view to experiences acquired in building multi-agent systems, attention has turned e.g. towards market mechanisms, with the study of properties such as fairness and truthfulness and their utility and applicability to optimise coordination among agents [26]. Overall, there has been an application-driven trend towards the establishment of coordination as a separate, fundamental component of multi-agent systems that captures distributed agent-level functionality as opposed to domain-level problem-solving capabilities [65]. Another trend is to capture knowledge about the design of coordination architectures using patterns. Some work has been performed to document coordination patterns (see [87]) and to define coordination frameworks to model institutions [24]. A systematic guide and overview of models, technologies, and applications of agent coordination in the Internet domain is provided in [65]. With the evolution of the Web from a static hypermedia system into a collection of services provided at different levels, the dearth of support for distributed applications provided by core Web technologies was put into evidence and has now become a central technological and research issue, with coordination mechanisms and agent-based architectures providing important contributions [19,87,65,88,17].

Approaches to the modeling of social capabilities of agents in multi-agent architectures traditionally differentiate between bottom-up environment-derived and top-down organisational/coordination flows of control. Macro structures studied that facilitate *collaboration* from an organisational point of view range from long-term stable organisation to flexibly formed teams and coalitions; however, the analysis of the effectiveness of different conventions for organisational structuring in different environments/domains is only just starting, as are first attempts in the direction of allowing for dynamic adaptation. [43] discusses the current state of the art in dynamic coalition formation among rational software agents in open and heterogeneous distributed environments such as the Internet and Web and identifies extant research issues in the development of a general framework of dynamic coalition formation. To exploit the potential of a given multi-agent system to the fullest, interaction should not be overly constrained at design time. To allow for such peer-to-peer scenarios, agents do have to be able to engage in negotiations at a meta level to reach agreements about the performance of tasks at hand, aligning individual private goals [29,30]. Models of persuasion [25] and justification [26,23] are contributing to the furthering of the state of the art in the area of team formation, as are acquaintance models [54] that exploit the characteristics of individuality and persistence of populations of multi-agent systems. Another example of research on collaboration, indicating the latitude of current efforts, is the work on ontological overhearing documented in [2,16]. Here, the ambitious goal is to improve collaboration between intelligent agents by intervention of third agents who act helpfully without having been explicitly involved (see also chapter 6 in this volume). The recent research area of Socionics [39,52] seeks to address the question of how to exploit models from the social world for the development of multiagent systems. It aims at identifying essential characteristics of modern societies and the sources of resilience, adaptivity and innovativity of social systems and to translate these features into intelligent technologies so as to contribute to close the gap between the "monoagent" systems (typically with an Artificial Intelligence heritage) and multi-(multi-)agent systems discussed above.

Information agents do not perform in a vacuum, nor does it make any sense to attempt to confine and reduce the environment rigorously to an external entity that is interfaced to for task-domain specific purposes only. For one, the potential and significance is clearly shown by designs capitalising on the properties of stigmergy mentioned before (see e.g. [53]). But more generally, the relevance of an environment capable of sustaining various kinds of micro- and macro-level architectures of agents is gaining more wide-spread acceptance, as also documented in the European AgentCities initiatives. As often, this brings about the danger of Balkanisation, the — intentional or not — segregation of research following different design principles. In the same way that the relationship of agent technology to underlying infrastructural support has been discussed before, cf. e.g. the discussions about KQML vs. CORBA, it is now of the utmost importance not to lose contact and to encourage and foster a continuous and lively exchange, e.g. between the agent community and grid-based approaches.

Finally, engineering of multiagent systems also requires tools to monitor and debug inter-agents aspects, such as interaction protocols, coordination policies, social norms and environment constraints. Examples of such efforts include research on tuple-based coordination infrastructures, which already delivered technologies such as TuCSoN and LuCe [66,77,22] (see also the chapter by Omicini and Ossowski in the present volume).

4 The Contributions

As mentioned in previous section, the book is structured into three parts, each of which includes contributions from members of one of the working groups of the SIG, and is edited by the respective working group chair. We first present contributions made by members of the working group on agent-based information systems (chapters 1 to 4), edited by Sonia Bergamaschi (University of Bologna, Italy), followed by those authored by members of the working group on adaptive information agents (chapters 5 to 7), edited by Pete Edwards (University of Aberdeen, UK), and finally, the presentations of joint results produced by members of the working group on communication, coordination, and collaboration (chapters 8 to 10), edited by Paolo Petta (Austrian Research Institute for Artificial Intelligence, Austria).

Please note that all contributions are offsprings or are at least strongly influenced by the intensive collaborative work between members of the SIG since its very beginning in 1998. Each chapter provides the reader with the latest advances in research and development in the domain across Europe as it is seen from the AgentLink perspective, that is, the members of the AgentLink SIG on intelligent information agents. The following sequence of contributions starts off with four contributions from the working group on agent-based information systems, followed by three chapters on work performed by members of the working group on adaptive information agents. Finally, we present three chapters provided by members of the working group on communication, coordination, and collaboration.

In chapter 1, Domenico Beneventano, Sonia Bergamaschi, Gionata Gelati, Francesco Guerra and Maurizio Vincini propose the exploitation of mobile agents as a possible approach to the scalability issue in the information integration domain. Their basic ideas take shape into the MIKS system. MIKS is an agent-based framework dealing with the integration and query of multiple and heterogeneous information sources distributed over the Web. The authors argue that software agent technology is able to meet requirements against flexibility in terms of system configuration and user needs, pro-activeness in terms of responses to a changing environment, and scalability in the number of users and reliability in terms of coping with an unstable environment. Throughout the design phase, agents are defined according to the step they come into play and the service the provide within the integration or query process. In this way, agents can be classified according to the three functionalities delineated in [41]: translation among diverse ontological domains, decomposition and execution of complex queries, and fusion of partial answers.

The issue of information integration is also the main topic of chapter 2 where Marie-Christine Rousset and Chantal Reynaud present two recent information integration agents: PICSEL and Xyleme. They follow two radically different choices concerning the expressivity of the mediated schema. The formalism adopted in PICSEL combines the expressive power of rules and classes for designing a rich mediated schema, thus enabling a fine-grained description of the contents of information sources. On the other hand, in Xyleme, the authors have chosen a simple tree structure for mediated schemas with the aim of providing a very wide-area integration of XML sources that could scale up on the Web. A comparison of the two information integration agents based on their query processing capabilities is also provided. While PICSEL is proved to be useful to model the fine-grained differences between contents of sources in order to be able to answer precise queries in an efficient way, the system architecture and the design choices about Xyleme have been motivated by "web search engine"-like performance requirements, i.e. supporting many simultaneous queries over a Web-scale XML repository.

Information extraction is the topic addressed in chapter 3. Nicholas Kushmerick and Bernd Thomas survey a variety of information extraction techniques that enable information agents to automatically gather information from heterogeneous sources. They argue that neither XML nor the Semantic Web initiative will eliminate the need for automatic information extraction since terabytes of extant data will probably never be exported in XML format and correct annotation scheme are difficult to be discovered. The chapter mainly focuses on the use of machine learning to enable adaptive information extraction systems that automatically learn extraction rules from training data in order to scale with the number of sources.

Chapter 4 includes a contribution on agent-based distributed data mining by Matthias Klusch, Stefano Lodi and Gianluca Moro. One key aspect of exploiting the huge amount of autonomous and heterogeneous data sources in the Internet is not only how to retrieve, collect and integrate relevant information but to discover previously unknown, implicit and valuable knowledge. In recent years several approaches to distributed data mining and knowledge discovery have been developed, but only a few of them make use of intelligent agents. This contribution is intended to argue for the potential added value of using agent technology in the domain of knowledge discovery. The authors briefly review and classify existing approaches to agent-based distributed data mining, propose a novel approach to distributed data clustering based on density estimation, and discuss issues of its agent-oriented implementation.

In chapter 5, Ian Dickinson, Dave Reynolds, Dave Banks, Steve Cayzer, and Poorvi Vora present a framework for the use of adaptive personal agents and user profiling for information systems designed around web services that respect the privacy of the individual. A key element of their approach in general is to consider the impact of user-profiling and autonomous agents on the user. Since personalisation and privacy have contradictory goals in disclosing personal information, the developed framework allows the user to control the trade-offs around this disclosure.

The notion of implicit culture refers to the relation between two sets of agents such that the agents of one set behave according to the culture of the other set; implicit culture can be seen as a generalisation of collaborative or social filtering. In chapter 6, Enrico Blanzieri and Paolo Giorgini present an implementation of a system for supporting implicit culture, display the results obtained by applying it to some recommendation problems, and an application to the eCulture Brokering System, a multi-agent system aimed to mediate the access to cultural information.

In chapter 7, Josep Lluis de la Rosa, Esteve del Acebo, Beatriz Lopez and Miquel Montaner argue in favour of the interchange of research results between physical agents and recommender systems. Certain properties, such as introspection and physical foundations of physical agents, are mapped into agents that model and make recommendations to users from the subjective, content-based point of view of products and services. How this real-to-virtual mapping of properties and behaviours can be performed is the main goal of the authors' work, from which they expect to obtain a general concept of recommender agents which improves the state of the art in terms of performance and maintainability. Furthermore, new features of the recommender agents could be also mapped to the physical agents discipline. Some of the features explained in this chapter have been installed in two commercial personalization products: Proto-agent and Habitat-Pro.

In chapter 8, Andrea Omicini and Sascha Ossowski provide a summary of the state of the art in agent oriented coordination research. The contributions of design-time and run-time perspectives at both the micro- and the macro-level of multi-agent system design are discussed in an integrated view that encompasses both aspects of the internals of architectures of individual agents as well as — and importantly — coordination services as elements of the agent system runtime environment. The provision of this full repertory of complementary subjective and objective models, abstractions, and technologies of coordination forms an important contribution for the improvement over the current state of the art in agent oriented design and engineering.

As discussed in the previous section, successful deployment of multi-agent systems technology in open domains is one of the biggest challenges ahead. In chapter 9, Martin Fredriksson, Rune Gustavsson and Alessandro Ricci report on their efforts to develop a coordination technology methodology capable to support progress in this direction. The authors identify the continuous loop of articulation, construction, observation, and instrumentation of system behaviour as a cornerstone of sustainable coordination of behaviour in open computational systems. They discuss the Vocs framework (Visions of open computational systems), which emphasises in particular the critical roles of observation and construction to sustain coordination in open multi-agent systems. Their presentation of a naval multi-agent system scenario also illustrates the utilisation of the methodological tools developed to support this iterative approach. In this context, Kiveat diagrams are introduced as means allowing to compare qualities of alternative technologies, in particular, those qualities that are seen to contribute to systemic properties, such as sustainability.

In the last chapter of this book, Juan Manuel Serrano, Sascha Ossowski, and Alberto Fernández present a principled approach for the extension of the standardised catalogue of performatives and protocols provided by the FIPA ACL so as to accommodate the requirements of specific given application domain contexts. They thereby tackle the practical problem of how to design an agent communication language for a particular multi-agent system, covering its communicative acts as well as its interaction protocols, while complying with both, expressiveness and interoperability requirements. Following the critique of the use of "standard ACLS" as a common language for the development of any particular MAS — given the lack of adequate support for all kinds of social interactions — put forward in [75], the authors define an integrated view of the organisation model and the agent communication language components of a multi-agent system in their RICA (Role/Interaction/Communicative Action) metamodel expressed in UML. Building upon this basis, a principled method for the design of the ACL of a particular multi-agent system is discussed and illustrated with an example from the domain of agent-based decision support for bus fleet management. The proposed design method thus makes the organisational implications of the reuse of ACL performatives and protocols explicit. In closing, the authors of this chapter also relate their work to the research agenda entry of making the transition from closed to open agent environments.

Acknowledgments. All contributions have gone through editorial review. We are indebted to the authors for their preparation of high-quality and original contributions, and the executive editor Alfred Hofmann for his support of this special volume in the series. The Austrian Research Institute for Artificial Intelligence acknowledges basic financial support by the Austrian Federal Ministry for Education, Science, and Culture.

References

1. AgentLink. European Network of Excellence for Agent-Based Computing. http://www.agentlink.org
2. M. Aiello, P. Busetta, A. Donà, and L. Serafini. Ontological overhearing. In [57], pages 175–189, 2002.
3. L. Amgoud, N. Maudet, and S. Parsons. An argumentation-based semantics for agent communication languages. In F. van Harmelen, editor, *Proceedings of the 15th European Conference on Artificial Intelligence (ECAI 2002)*, IOS Press, Amsterdam, pages 38–42, 2002.
4. E. André and T. Rist, Adding life-like characters to the Web. In M. Klusch, and L. Kerschberg, editors, *Cooperative Information Agents IV*, Proceedings of the CIA 2000 Workshop, Boston, USA, Lecture Notes in Artificial Intelligence, 1860, (Springer, 2000).
5. A. Ankolekar, M. Burstein, J. Hobbs, O. Lassila, D. Martin, S. McIllraith, S. Narayanan, M. Paolucci, T. Payne, K. Sycara, H. Zeng, DAML-S: Web Service Description for the Semantic Web. Proceedings of the International Semantic Web Conference, ISWC 2002.
6. Y. Arens and C.Y. Chee, C.A. Knoblock, Retrieving and Integrating Data from Multiple Information Sources. International Journal of Intelligent and Cooperative Information Systems, volume 2, number 2, pages 127–158, 1993.
7. N. Ashish, C.A. Knoblock, Semi-Automatic Wrapper Generation for Internet Information Sources. Proceedings of the 2nd International Conference on Cooperative Information Systems (CoopIS), pages 160–169, 1997.
8. B. Bauer, F. Bergenti, P. Massonet, and J.J. Odell. Agents and the UML: A unified notation for agents and multi-agent systems? In M.J. Wooldridge, G. Weiss, P. Ciancarini, editors, *Proceedings of the Second International Workshop on Agent-Oriented Software Engineering (AOSE-2001)* held at the 5th International Conference on Autonomous Agents, Volume 2222 of *LNCS*, Springer-Verlag Heidelberg, pages 101–118, 2002.
9. F. Bellifemine, A. Poggi, G. Rimassa, JADE: a FIPA2000 compliant agent development environment. In Proceedings of the 5th International Conference on Autonomous agents, pages 216–217, http://jade.cslet.it, 2001.
10. D. Beneventano, S. Bergamaschi, S. Castano, A. Corni, R. Guidetti, G. Malvezzi, M. Melchiori, M. Vincini, Information Integration: The MOMIS Project Demonstration. Proceedings of 26th International Conference on Very Large Data Bases (VLDB), pages 611–614, 2000.

11. V.R. Benjamins, Project Presentation MISSION: Multi-agent Integration of Shared Statistical Information Over the (inter)Net. Deliverable 1, 1999. Available at the MISSION web site http://www.epros.ed.ac.uk/mission/.
12. V.R. Benjamins, Project Presentation IBROW: An Intelligent Brokering Service for Knowledge-Component Reuse on the World-Wide Web. Deliverable 17, 2000. Available at the IBROW web site http://www.ibrow.org.
13. S. Bergamaschi, S. Castano, D. Beneventano, M. Vincini, Semantic integration of heterogenous information sources. *Journal of Data and Knowledge Engineering*, 36(3):215–249, 2001.
14. F. Bergenti and A. Poggi. Exploiting UML in the design of multi-agent systems. In [63], pages 106–113, 2000.
15. J.M. Bradshaw, S. Dutfield, P. Benoit, and J.D. Woolley. KAoS: Toward an industrial-strength open agent architecture. In J.M. Bradshaw, editor, *Software Agents*, AAAI Press/MIT Press, Cambridge/Menlo Park, pages 375–418, 1997.
16. P. Busetta, A. Donà, and M. Nori. Channeled multicast for group communications. In C. Castelfranchi and W.L. Johnson, editors, *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems*, ACM Press, New York, pages 1280–1287, 2002.
17. G. Cabri, L. Leonardi, and F. Zambonelli. Engineering mobile agent applications via context-dependent coordination. *IEEE Transactions on Software Engineering*, 28(11):1040–1056, November 2002.
18. C. Castelfranchi and Y. Lespèrance, editors, *Intelligent Agents VII – Agent Theories, Architectures, and Languages*, 7th International Workshop, ATAL 2000, Boston, MA, USA, July 2000. Proceedings, volume 1986 of *LNAI*, Springer-Verlag Berlin Heidelberg 2001.
19. P. Ciancarini and R. Tolksdorf, editors, Special Issue: Coordination Mechanisms for Web Agents, *Autonomous Agents and Multi-Agent Systems*, 2(3:September), 1999.
20. P. Ciancarini and M.J. Wooldridge, editors, *Agent-Oriented Software Engineering*, First International Workshop, AOSE 2000 Limerick, Ireland, June 10, 2000. Revised Papers, Volume 1957 of *LNCS*, Springer-Verlag Berlin Heidelberg, 2001.
21. V. Crescenzi, G. Mecca, P. Merialdo, RoadRunner: Towards Automatic Data Extraction from Large Web Sites. In Proceedings of 27rd International Conference on Very Large Data Bases (VLDB), pages 109–118, 2001.
22. E. Denti, A. Omicini, and A. Ricci. Coordination tools for MAS development and deployment. In [70], pages 721–752, 2002.
23. F. Dignum. Formal theories of negotiation. In [57], pages 321–323, 2002.
24. V. Dignum and F. Dignum. Modelling agent societies: Coordination frameworks and institutions. In P. Brazdil and A. Jorge, editors, *Progress in Artificial Intelligence*, volume 2258 of *LNAI*, Springer-Verlag Berlin Heidelberg, pages 191–204, 2001.
25. F. Dignum, B. Dunin-Keplicz, and R. Verbrugge. Agent theory for team formation by dialogue. In [18], pages 150–166, 2001.
26. F. Dignum and C. Sierra, editors, *Agent Mediated Electronic Commerce (The European AgentLink Perspective)*, volume 1991 of *LNCS*, Springer-Verlag Heidelberg, 2001.
27. R.M. van Eijk, F.S. de Boer, W. van der Hoek, and J.-J.C. Meyer. Operational semantics for agent communication languages. Technical Report UU-CS-1999-08, Department of Computer Science, Utrecht University, 1999.

28. European AgentLink Special Interest Group on Intelligent Information Agents. Since 2002: http://www.dbgroup.unimo.it/IIA/index.html; 1998–2002 http://www.dfki.de/ klusch/i2a-sig.html
29. C.B. Excelente-Toledo and N.R. Jennings. Learning to select a coordination mechanism. In C. Castelfranchi and W.L. Johnson, editors, *Proceedings of the first international joint conference on autonomous agents and multiagent systems*, ACM Press, New York, part 3 of 3, pages 1106–1113, 2002.
30. S.S. Fatima, M. Wooldridge, and N.R Jennings. Optimal negotiation strategies for agents with incomplete information. In [57], pages 377–392, 2002.
31. D. Fensel, F. van Harmelen, Project Presentation On: Content-driven Knowledge-Management Tools through Evolving Ontologies. Deliverable 33, 1999. Available at the On-To-Knowledge web site http://www.ontoknowledge.org.
32. D. Fensel, I. Horrocks, F. van Harmelen, S. Decker, M. Erdmann, M. Klein, OIL in a Nutshell. Proceedings of the 12th International Conference on Knowledge Acquisition, Modeling and Management (EKAW), pages 1–16, 2000.
33. T. Finin, Y. Labrou, J. Mayfield, KQML as an agent communication language. MIT Press, 1997.
34. FIPA, Agent Communication Language, available at http://www.fipa.org/spec/fipa97/FIPA97.html.
35. J. Hendler, D.L. McGuiness, The DARPA Agent Markup Language. *IEEE Intelligent Systems Journal*, 16(6):63–73, 2000.
36. M. Huhns and M.P. Singh, editors, *Readings in Agents*, (Morgan Kaufmann, 1998).
37. IAF, web site available at http://www.cordis.lu/ist/ka3/iaf/index.htm.
38. D. Kassai, Project Presentation SILK. Deliverable 3, 2000. Available at the SILK web site http://www.idec.gr/silk/.
39. S. Kirn. From organizational design to the design of agent systems: Towards a multiagent organization theory. In R. Trappl, editor, *Cybernetics and Systems '98: Proceedings of 14th European Meeting on Cybernetics and Systems Research*, Austrian Society for Cybernetic Studies, Vienna, pages 773–778, 1998.
40. M. Klein. Coordination science: Challenges and directions. In W. Conen and G. Neumann, editors, *Coordination technology for collaborative applications*, volume 1364 of *LNCS*, Springer-Verlag, 1998.
41. M. Klusch, editor, *Intelligent Information Agents*, (Springer, Berlin, 1999).
42. M. Klusch, Information agent technology for the Internet: A survey, *J.* Data and Knowledge Engineering, 36(3), (Elsevier Science, 2001).
43. M. Klusch, and A. Gerber. Dynamic coalition formation among rational agents. *IEEE Intelligent Systems*, Knowledge systems for coalition operations, 17(3):42–47, 2002.
44. M. Klusch, and L. Kerschberg, editors, *Cooperative Information Agents IV*, Proceedings of the CIA 2000 Workshop, Boston, USA, Lecture Notes in Artificial Intelligence, 1860, (Springer, 2000). Proceedings of CIA 1997, 1998, and 1999 are available as LNAI volumes 1202, 1453, and 1652, respectively.
45. KOD, web site available at http://kod.iti.gr.
46. Y. Labrou and T. Finin. A proposal for a new KQML specification. Technical Report CS-97-03, Department of Computer Science and Electrical Engineering, University of Maryland, Baltimore, 1997.
47. Y. Labrou, T. Finin, and Y. Peng. Agent communication languages: The current landscape. *IEEE Intelligent Systems*, Unmasking intelligent agents, 14(2):45–52, 1999.

48. C. Li, R. Yerneni, V. Vassalos, H. Garcia-Molina, Y. Papakonstantinou, J.D. Ullman, M. Valiveti, Capability Based Mediation in TSIMMIS. Proceedings of the ACM SIGMOD International Conference on Management of Data, pages 564–566, 1998.
49. J. Lind. Specifying agent interaction protocols with standard UML. In [96], pages 136–147, 2002.
50. M. Luck, V. Marik, O. Stepankova, and R. Trappl, editors, *Multi-Agent Systems and Applications.* volume 2086 of *LNAI*, Springer-Verlag Berlin Heidelberg New York, 2001.
51. T. Malone and K. Crowstone. The interdisciplinary study of coordination. *ACM Computing Surveys*, 26(1):87–119, 1994.
52. T. Malsch. Naming the unnamable: Socionics or the sociological turn of/to distributed artificial intelligence. *Autonomous Agents and Multi-Agent Systems*, 4(3):155–186, 2001.
53. M. Mamei, F. Zambonelli, and L. Leonardi. Co-fields: Towards a unifying approach to the engineering of swarm intelligent systems. In [71], 2002.
54. V. Marik, M. Pechoucek, and O. Stepankova. Social knowledge in multi-agent systems. In [50], pages 211–245, 2001.
55. M.T. Maybury and W. Wahlster, editors, *Readings in Intelligent User Interfaces.* (Morgan Kaufmann, 1998).
56. H. McEvoy. Coordinating multiset transformers. PhD. Thesis, University of Amsterdam, 1997.
57. J. Meyer and M. Tambe, editors, *Intelligent Agents VIII — Agent Theories, Architectures, and Languages*, 8th International Workshop, ATAL 2001 Seattle, WA, USA, August 1–3, 2001, Revised Papers, volume 2333 of *LNAI*, Springer-Verlag Berlin Heidelberg 2002.
58. T. Mitchell. *Machine Learning*, (McGraw-Hill, 1997).
59. M. Nodine et al., Active information gathering in InfoSleuth. *J. Cooperative Information Systems*, 9(1/2), 2000.
60. T.J. Norman and C. Reed. Delegation and responsibility. In [18],pages 136–149, 2001.
61. G.M.P. O'Hare and N.R. Jennings, editors, *Foundations of Distributed Artificial Intelligence*, Wiley Chichester London New York, 1996.
62. E. Oliveira. Agents' advanced features for negotiation and coordination. In [50], pages 173–186, 2001.
63. A. Omicini, R. Tolksdorf, and F. Zambonelli, editors, *Engineering Societies in the Agents World, First International Workshop, ESAW 2000*, Berlin, Germany, August 2000, Revised Papers, volume 1972 of *LNAI*, Springer-Verlag Berlin Heidelberg, New York, Barcelona, 2000.
64. A. Omicini, P. Petta, and R. Tolksdorf, editors, *Engineering Societies in the Agents World II, Second International Workshop, ESAW 2001*, Prague, Czech Republic, July 2001, volume 2203 of *LNAI*, Springer-Verlag Berlin Heidelberg New York, 2001.
65. A. Omicini, F. Zambonelli, M. Klusch, and R. Tolksdorf, editors, *Coordination of Internet Agents*, Springer-Verlag Berlin Heidelberg New York, 2001.
66. A. Omicini and G.A. Papadopoulos, editors, Special Issue on Coordination Models and Languages in AI, *Applied Artificial Intelligence 15(1)*, 2001.
67. S. Ossowski. *Co-ordination in Artificial Agent Societies.* volume 1535 of *LNAI*, Springer-Verlag Berlin Heidelberg New York Tokyo, 1999.
68. M.P. Papazoglou, and G. Schlageter, editors, *Cooperative Information Systems: Trends and Directions.* (Academic Press, 1998).

69. P. Petta and J.P. Müller, editors, *From Agent Theory to Agent Implementation: Best of AT2AI-1*, *Applied Artificial Intelligence*, 14(7), 2000.
70. P. Petta and J.P. Müller, editors, *Best of "From Agent Theory to Agent Implementation 3"*, *Applied Artificial Intelligence*, 16(9–10), 2002.
71. P. Petta, R. Tolksdorf, F. Zambonelli, and S. Ossowski, editors, *ESAW'02: Workshop notes of the third international workshop "Engineering societies in the agents world"*, 16–17 September 2002, Universidad Rey Juan Carlos, Madrid, Spain, Österreichisches Forschungsinstitut für Artificial Intelligence, Wien, TR-2002-35, 2002.
72. P. Petta, R. Tolksdorf, F. Zambonelli, and S. Ossowski, editors, *Engineering Societies in the Agents World III, Third International Workshop, ESAW 2002*, Madrid, Spain, September 2002, Revised Papers, volume 2577 of *LNAI*, Springer-Verlag Berlin Heidelberg, 2003.
73. R.W. Picard. *Affective Computing*. (MIT Press, Cambridge, 1997).
74. J. Pitt and A. Mamdani. Some remarks on the semantics of FIPA's Agent Communication Language. *Autonomous Agents and Multi-Agent Systems*, 2(4):333–356, 1999.
75. J. Pitt and A. Mamdani. Designing agent communication languages for multi-agent systems. In F.G. Garijo and M. Boman, editors, *Multi-Agent System Engineering*, 9th European Workshop on Modelling Autonomous Agents in a Multi-Agent World, MAAMAW'99 Valencia, Spain, June 30 – July 2, 1999 Proceedings, volume 1647 of *LNCS*, pages 102–114, 1999.
76. F. de Rosis, editor, Merging Cognition and Affect in HCI, *Applied Artificial Intelligence*, 16(7–8), 2002.
77. D. Rossi, G. Cabri, and E. Denti. Tuple-based technologies for coordination. In [65], pages 83–109, 2001.
78. M.T. Roth, M. Arya, L.M. Haas, M.J. Carey, W.F. Cody, R. Fagin, P.M. Schwarz, J. Thomas, E.L. Wimmers, The Garlic Project. Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data, pages 557, 1996.
79. M.T. Roth, Peter M. Schwarz, Don't Scrap It, Wrap It! A Wrapper Architecture for Legacy Data Sources. In Proceedings of 23rd International Conference on Very Large Data Bases (VLDB), pages 266–275, 1997.
80. SEWASIE, web site available at http://www.sewasie.com.
81. A.P. Sheth, J.A. Larson, Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases. *ACM Computing Surveys*, 22(3):183–236, 1990.
82. A. Sheth, V. Kashyap, and T. Lima, Semantic information brokering: How can a multi-agent approach help? In M. Klusch, O. Shehory, and G. Weiss, editors, *Cooperative Information Agents III*, Proceedings of the CIA 1999 Workshop, Uppsala, Sweden, Lecture Notes in Artificial Intelligence, 1652, (Springer, 1999).
83. M.P. Singh. Agent Communication Languages: Rethinking the principles. *Computer*, 31(12):40–47, 1998.
84. V.S. Subrahmanian et al., *Heterogeneous Agent Systems: Theory and Implementation*, Cambridge, USA, (MIT Press, 2000).
85. K. Sycara and D. Zeng. Coordination of multiple intelligent software agents. *J. Cooperative Information Systems*, 5(2/3):181–211, (World Scientific, 1996)
86. K. Sycara. Multi-agent infrastructure, agent discovery, middle agents for Web services and interoperation. In [50], pages 17–49, 2001.
87. R. Tolksdorf. Models of coordination. In [63], pages 78–92, 2000.
88. R. Tolksdorf. Workspaces: A Web-based workflow management system. *IEEE Internet Computing*, Global Deployment of Data Centers, 6(5):18–26, 2002.

89. R. Trappl, P. Petta, S. Payr, editors, *Emotions in Humans and Artifacts*, MIT Press, Cambridge (MA), 2003.
90. M.E. Vidal, L. Raschid, J.-R. Gruser, A Meta-Wrapper for Scaling up to Multiple Autonomous Distributed Information Sources. Proceedings of the 3rd International Conference on Cooperative Information Systems (CoopIS), pages 148–157, 1998.
91. M. Viroli and A. Omicini. An observation approach to the semantics of Agent Communication Languages. In [70], pages 775–794, 2002.
92. G. Weiss, editor, *Multiagent Systems*, (MIT Press, 1999).
93. G. Wiederhold. Mediators in the architecture of future information systems. *IEEE Computer*, 25:38–49, (IEEE Press, 1992).
94. M. Wooldridge. Semantic issues in the verification of Agent Communication Languages. *Autonomous Agents and Multi-Agent Systems*, Special Issue: Best of ICMAS'98 — Part I, 3(1):9–32, 2000.
95. M. Wooldridge and N.R. Jennings, Intelligent Agents: Theory and Practice. *Knowledge Engineering Review*, 10(2), 1995.
96. M.J. Wooldridge, G. Weiss, and P. Ciancarini, editors, *Proceedings of the Second International Workshop on Agent-Oriented Software Engineering (AOSE-2001)* held at the 5th International Conference on Autonomous Agents, Volume 2222 of *LNCS*, Springer-Verlag Berlin Heidelberg, 2002.
97. F. Zambonelli and H. Van Dyke Parunak. Signs of a revolution in computer science and software engineering. In [71], 2002.

MIKS: An Agent Framework Supporting Information Access and Integration*

Domenico Beneventano[1,2], Sonia Bergamaschi[1,2],
Gionata Gelati[1], Francesco Guerra[1], and Maurizio Vincini[1]

[1] Dipartimento Ingegneria dell'Informazione, via Vignolese 905, 41100 Modena Italy
[2] CSITE-CNR, viale Risorgimento 2, 40136 Bologna, Italy
{domenico.beneventano,sonia.bergamaschi,maurizio.vincini}@unimo.it
{gelati.gionata,guerra.francesco}@unimo.it

Abstract. Providing an integrated access to multiple heterogeneous sources is a challenging issue in global information systems for cooperation and interoperability. In the past, companies have equipped themselves with data storing systems building up informative systems containing data that are related one another, but which are often redundant, not homogeneous and not always semantically consistent. Moreover, to meet the requirements of global, Internet-based information systems, it is important that the tools developed for supporting these activities are semi-automatic and scalable as much as possible.
To face the issues related to scalability in the large-scale, in this paper we propose the exploitation of *mobile agents* in the information integration area, and, in particular, their integration in the *MOMIS* infrastructure. MOMIS (Mediator EnvirOnment for Multiple Information Sources) is a system that has been conceived as a pool of tools to provide an integrated access to heterogeneous information stored in traditional databases (for example relational, object oriented databases) or in file systems, as well as in semi-structured data sources (XML-file).
This proposal has been implemented within the MIKS (Mediator agent for Integration of Knowledge Sources) system and it is completely described in this paper.

1 Introduction

Providing an integrated access to multiple heterogeneous sources is a challenging issue in global information systems for cooperation and interoperability. In the past, companies have equipped themselves with data storing systems building up informative systems containing data that are related one another, but which are often redundant, not homogeneous and not always semantically consistent. The problems that have to be faced in this field are mainly due to both structural and application heterogeneity, as well as to the lack of a common ontology, causing semantic differences between information sources. Moreover, these semantic

* This work is partially supported by MIUR co-funded projects D2I and 'Software Agents and e-Commerce'

M. Klusch et al. (Eds.): Intelligent Information Agents, LNAI 2586, pp. 22–49, 2003.

differences can cause different kinds of conflicts, ranging from simple contradictions in name use (when different names are used by different sources to indicate the same or similar real-world concept), to structural conflicts (when different models/primitives are used to represent the same information). Complicating factors with respect to conventional view integration techniques [1] are related to the fact that semantic heterogeneity occurs on the large-scale. This heterogeneity involves terminology, structure, and domain of the sources, with respect to geographical, organizational, and functional aspects of the information use [2]. Furthermore, to meet the requirements of global, Internet-based information systems, it is important that the tools developed for supporting these activities are semi-automatic and scalable as much as possible.

To face the issues related to scalability in the large-scale, in this paper we propose the exploitation of *mobile agents* in the information integration area, and, in particular, their integration in the *MOMIS* infrastructure. *MOMIS* [3, 4] (Mediator EnvirOnment for Multiple Information Sources) is a system that has been conceived as a pool of tools to provide an integrated access to heterogeneous information stored in traditional databases (for example relational, object oriented databases) or in file systems, as well as in semi-structured data sources (XML-file). *MOMIS* focuses on capturing and reasoning about semantic aspects of schema descriptions of information sources for supporting integration and query optimization. This proposal has been implemented within the *MIKS* (Mediator agent for Integration of Knowledge Sources) system and it is completely described in this paper.

Mobile agents can significantly improve the design and the development of Internet applications thanks to their characteristics. The agency feature [5] permits them to exhibit a high degree of autonomy with regard to the users: they try to carry out their tasks in a *proactive* way, *reacting* to the changes of the environment they are hosted. The mobility feature [6] takes several advantages in a wide and unreliable environment such as the Internet. First, mobile agents can significantly save bandwidth, by moving locally to the resources they need and by carrying the code to manage them. Moreover, mobile agents can deal with non-continuous network connection and, as a consequence, they intrinsically suit mobile computing systems. All these features are particularly suitable in the information retrieval area [7].

MIKS is an agent framework for information integration that deals with the integration and query of multiple, heterogeneous information sources, containing structured and semi-structured data. This framework is a support system for semi-automatic integration of heterogeneous sources schema (relational, object, XML and semi-structured sources); it carries out integration following a semantic approach which uses Description logics-based techniques, clustering techniques and an ODM-ODMG [8] extended model to represent extracted and integrated information, ODM_{I^3}.

The *MIKS* system can be defined as an agent middleware system that integrates data belonging to different and potentially heterogeneous sources into a global virtual view and offers support for the execution of queries over the global virtual schema [9]. Middleware systems dealing in some way with a set of data sources commonly fall back on wrapper components or simply wrappers [10]. Wrappers components encapsulate existing legacy data sources and give a more presentable and understandable format according to some preferred common data model.

The outline of the paper is the following. Section 2 presents the basics related to the approach we have chosen in tackling the integration of heterogeneous data sources, section 3 reports the *MOMIS* system. Section 4 introduces the *MIKS* system, illustrating the role of the agents in a framework supporting information access and integration. Finally, section 6 discusses the related work in the area of intelligent information agents.

2 A Feasible Approach to Intelligent Information Integration Architectures

In this section we introduce the basics related to the approach we have chosen in tackling the integration of heterogeneous data sources. In the first sub-section we present the reference architecture and in the second sub-section we step through the integration process.

2.1 System Architecture

Like in other integration projects ([11,12]), we have chosen to pursue a “semantic approach” to information integration, i.e. we represent the content of information sources by means of conceptual schemas (or in other terms metadata). This allows to process data not only from a syntactical point of view but also according to their meaning in order to infer extensional and intensional relationships among them. Given the goal of our system, our reference architecture has been the I^3 architecture as specified in [13] for the Intelligent Integration of Information.

Figure 1 shows the five fundamental families of I^3 Services and the primary ways in which they interact. In particular, two salient axis are definable to emphasize the different roles of the I^3 services. The vertical axis, that spans the families (1), (3), (5), is focused on the flow and manipulation of information from raw Information Sources up to the Coordination Services. Then, an horizontal axis connects the Management family and the Coordination family. This axis emphasizes a critical aspect of the I^3 Reference Architecture, i.e. the role of the Management family in order to locate useful information sources and to exploit the local data structures.

Our system mainly exploits services belonging to the vertical axis: in particular the family (3) has been completely investigated. In previous work, we faced the issues related to the semantic integration and transformation, developing and implementing a methodology to integrate schema of heterogeneous information sources. Our research has highlighted that besides the functional elements belonging to the reference system architecture, a set of functional extensions (family 4 in the architecture) is required in order to support designers during

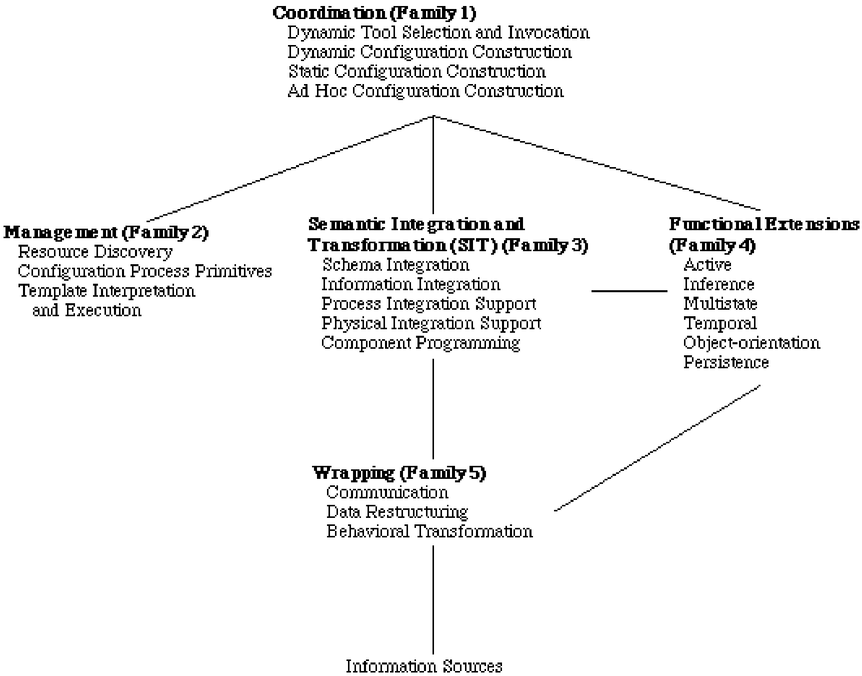

Fig. 1. Reference System Architecture

the integration process, especially when delaying with both semi-structured and structured sources. Thus, our approach has lead us to the introduction of the following services: a Common Thesaurus which has the role of a shared ontology of the source and reasoning capabilities based on Description Logics.

More precisely, the Common Thesaurus builds a set of intra and inter-schema intensional and extensional relationships, describing inter-schema knowledge about classes and attributes of sources schemas. Further, designers are left free to supply any further domain knowledge that can help refine the integration process. The Common Thesaurus provides a reference on which to base the identification of classes candidate to integration and subsequent derivation of their global representation.

As one of the goals of our integration system is revising and validating the various kinds of knowledge used for the integration, we have combined within the architecture reasoning capabilities of Description Logics with affinity-based clustering techniques.

In the following, we will consider the functional elements related to the Coordination family and propose a Multi-Agent System where coordinations take place in order to accomplish the required integration and querying functionalities.

2.2 The Integration Process

The overall information integration process we have assumed for our purposes is articulated in the following phases:

1. **Generation of a Common Thesaurus.**
   The Common Thesaurus is a set of terminological intensional and extensional relationships, describing intra and inter-schema knowledge about classes and attributes of sources schemas. In the Common Thesaurus, we express inter-schema knowledge in form of terminological and extensional relationships (synonymy, hypernymy and relationship) between classes and/or attribute names;
2. **Affinity analysis of classes.**
   Relationships in the Common Thesaurus are used to evaluate the level of affinity between classes intra and inter sources. The concept of affinity is introduced to formalize the kind of relationships that can occur between classes from the integration point of view. The affinity of two classes is established by means of affinity coefficients based on class names, class structures and relationships in Common Thesaurus.
3. **Clustering classes.**
   Classes with affinity in different sources are grouped together in clusters using hierarchical clustering techniques. The goal is to identify the classes that have to be integrated since describing the same or semantically related information.
4. **Generation of the mediated schema.**
   Starting from the output of the cluster generation, we define, for each cluster, a Global Class that represents the mediated view of all the classes of the cluster. For each global class a set of global attributes and, for each of them, the intensional mappings with the local attributes (i.e. the attributes of the local classes belonging to the cluster) are given.

2.3 Query Processing

A data integration system, based on conventional wrapper/mediator architectures, usually allows the user to pose a query and receive a unified answer without the need of: locating the sources relevant to the query, interacting with each source in isolation and combining the data coming from the different sources. In our framework, the user application interacts with the system to query the Global Schema by using the OQL_I^3 language. Using the mapping between local and global attributes, the Query Manager generates in an automatic way the reformulation/optimization of the generic OQL_I^3 query into different sub-queries, one for each involved local source (see [14]). To achieve the mediated query result, the Query Manager has to assemble each local sub-query result into a unified data set.

In a mediator architecture, the availability of several heterogeneous sources adds many novel issues to the query processing and optimization problem. From a theoretical point of view, solving a user (mediated) query, i.e. giving a single unified answer w.r.t. multiple sources, implies to face two main problems: *query reformulation/optimization* [15,16,17,18,19,20,21] and *object fusion* [22,23].

Semantic Query Optimization. In a mediator architecture the query manager usually relies on the availability of a global schema, the source schemata and a mapping between the two. On the other hand, the heterogeneity of information sources to be integrated often entails significant overlap or redundancy among them. Exploiting such an extensional knowledge is an important task in the query planning phase since most of the execution costs concerns the cost for querying remote sources, because of the high connection overhead, long computation time, financial charges and temporary unavailability. In [14] we have discussed the introduction of extensional knowledge as an additional knowledge allowing semantic optimization during the query planning phase.

In this context, besides incoherence detection and factor removal, the main achievements of semantic query optimization are related to the minimization of the number of sources to be accessed and the maximization of the selectivity of the query sent to the sources.

To illustrate the problem, let us consider a mediator integrating two sources. Suppose that the first source provides class `Student(name,email,year,tax)` and that the second source has `Professor(name,email,dept)`. Let us assume that the outcome of the integration phase is the mediated global schema with the global class `UniversityPerson(name,email,year,tax,dept)`. Let us consider the following query:

```
Q:  select email
    from University_Person
    where year = 2002
    and (dept ='cs'  or tax < 200)
```

Our semantics of a global class is based on the following hypothesis, called semantic homogeneity: the objects of `UniversityPerson` are objects of `Student` and/or of `Professor`; objects instantiated both in `Student` and `Professor` are individuate by considering the name attribute value and these objects have the same value for the common attributes. Thus, intuitively, the above query has to be reformulated on the basis of the local classes, as follows:

```
QS: select email,name,tax
     from Student
     where year = 2002

QP:  select name
      from Professor
      where dept = 'cs'
```

and then:

```
Q:   select email
     from QS
     where tax < 200
     union
     select email
     from QS, QP join on (QS.name=QP.name)
     where year = 2002
```

Now, let us suppose that the integration designer introduces the extensional knowledge that the local classes `Student` and `Professor` are disjoint. Then, in the above reformulation the answer of QS, QP join on `QS.name=QP.name` is empty and thus Q has to be reformulated only on the local class Student, as follows:

```
Q: select email
     from Student
     where year = 2002 and tax < 200
```

This example shows the main achievements of semantic query optimization of our method: (1) the minimization of the number of sources to be accessed and (2) the maximization of the selectivity of the query sent to the sources.

2.4 Managing Extensional Knowledge

A fundamental aspect of semantic query optimization is the management of the extensional knowledge. The main issue is that a set of extensional assertions which is initially verified by local sources can be violated if data are modified in one or more sources over time. Checking the consistency of the data against the extensional assertions is a very complex and challenging point. In [24] a framework to monitor and enforce distributed integrity constraints in loosely coupled heterogeneous information systems is presented. In our context, as a mediator is not the owner of the data stored in the local classes but it only provides a virtual view, we are interested only in the *verification* of integrity constraints. Nonetheless finding a solution still remains a complex activity. Thus, the main goal has to be that of reducing the transfer of data among remote heterogeneous sources. A useful optimization in a mediator environment is to be able to verify global extensional assertions by only accessing local data. Reference [56] describes a method for checking distributed constraints at a single source whenever possible. Given a global constraint and some data to be inserted in a local source, a local condition is produced in order to verify the consistency of local data. The consistency of local data is proven if (a) the local condition is (locally) satisfied and (b) before the new data were inserted, the global constraint was satisfied. this optimises the verification process as no remote data have to be accessed.

3 The *MOMIS* Project

Given the set of functionalities and how they have to work, a system can be designed following diverse approaches. A classical object-oriented implementation is feasible [25]. This approach is particularly suited to achieve client-server architectures. This way, the server acts as a centralised facility where all the computation takes place, while clients can be envisaged as lighter desktop applications and user interfaces. The server could comprise of more machines: the objects that provide the services on the server side can be distributed, typically using the CORBA [26] standard framework. Anyway, we can still maintain the server as a logically unified facility although its services are distributed over a local cluster of servers. This configuration is useful when the system is meant to operate in a limited and close search domain, i.e. search domains that are well-known as far the type and the number of sources to be integrated. A typical scenario of this sort could be represented by a small enterprise whose main goal is managing the few heterogeneous data sources belonging to the local information system. Users of the systems are usually the few analysts charged within the organisation of data monitoring and retrieval.

Within the *MOMIS* (Mediator envirOnment for Multiple Information Sources) project [28,63,4], we ourself have experienced the design and implementation of an object-oriented client-server integration system, that follows the architecture above presented. The *MOMIS* system has obviously been conceived to provide an integrated access to heterogeneous information stored in traditional databases (e.g., relational, object oriented) or file systems, as well as in semi-structured sources (*XML* files in particular). Figure 2 shows the architecture of the *MOMIS* system, based on the I^3 schema of Figure 1 : at the bottom layer we have the schema of information sources, while the layers above provide the semantic integration and the coordination management support.

As depicted in Figure 2, the system is composed by the following functional elements that communicates using the *CORBA* standard:

- Wrappers: The wrappers in *MOMIS* are the access point for the data sources. This implies that independently from the type of the underlying source (relational, object, structured and so forth), data must be presented in a standard way. Wrappers are *CORBA* objects connected to some sources and are able to describe the source structure using the ODL_i^3 language. Further, wrappers supply a way to query the source using the OQL_i^3 language;
- Mediator: It is composed of two modules: the *Global Schema Builder* (*GSB*) and the *Query Manager* (*QM*). The *GSB* module processes and integrates ODL_{I^3} descriptions received from wrappers to derive the integrated representation of the information sources. The *GSB* is the main *CORBA* object to access the *MOMIS* integration services. The *GSB* is composed by two modules:
  - *SIM* (Source Integrator Module): extracts intra-schema relationships starting from a relational, object and semi-structured source. Moreover this module performs the "semantic validation" of relationships and infers new relationships by exploiting *ODB-Tools* capabilities.

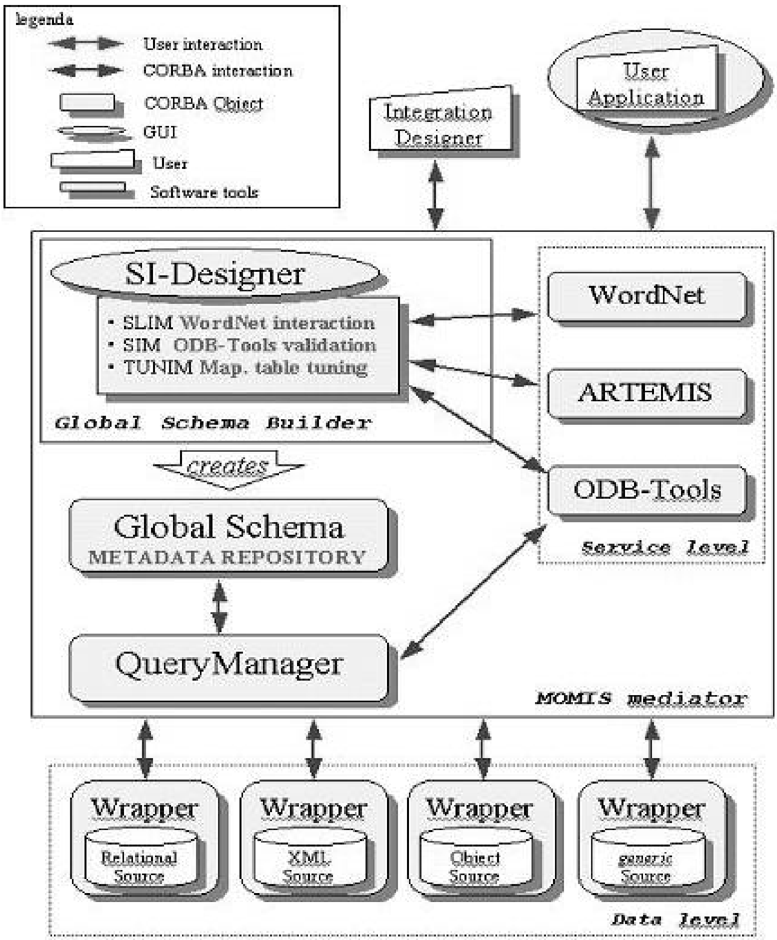

Fig. 2. The architecture of the *MOMIS* system

- *SLIM* (Sources Lexical Integrator Module) extracts inter-schema relationships between names and attributes of ODL_{I^3} classes of different sources, exploiting the *WordNet* lexical system.

All information about integration are then saved in the Global Schema which is defined as a *CORBA* object. A Global Schema object contains all information for querying the resulting global schema by a query manager object. The *QM* module performs query processing and optimization. In particular, it generates the OQL_{I^3} queries for wrappers, starting from a global OQL_{I^3} query formulated by the user on the global schema. Using Description Logics techniques, the *QM* component can generate in an automatic way the translation of the global OQL_{I^3} query into different sub-queries, one for each

involved local source. The *QM* is also responsible for synthesising the results of the sent sub-queries into one global answer to the original global query;
- The *ODB Tools* Engine ([50]), a tool based on the *OLCD* Description Logics ([29,30] which performs schema validation for the generation of the *CommonThesaurus* and query optimization ([31];
- The *ARTEMIS Tool Environment*, a tool based on affinity and clustering [32, 33], which performs classification of ODL_{I^3} classes for the synthesis of global classes.
- The *SI-Designer* module, which provides the designer with a graphical interface to interact with *SIM*, *SLIM* and *ARTEMIS* modules showing the extracted relationships and helping her/him in the Common Thesaurus construction. This way, the integration develops as a semi-automatic process where the designer can evaluate the result of the various steps and eventually try to modify the result according to his/her understanding. Once the Common Thesaurus, has been built, *SI-Designer* uses the *ARTEMIS* module to evaluate a disjoint set of structural similar classes. *SI-Designer* automatically generates a set of global attributes for each global class and a mapping table which maps each global attribute into the local attributes of the classes in the cluster.

  For a detailed description of the mappings selection and of the tool *SI-Designer* which assist the designer in this integration phase see [34].

Finally, in *MOMIS* the extensional knowledge is provided by the integration designer as a set of assertions: containment, equivalence and disjunction among local classes of the sources involved in the integration. The basic assumption is that these designer-supplied assertions are always verified over time. In the actual implementation, there is no verification tool to monitor the state of the sources against the extensional assertions.

4 The Role of the Agents: The *MIKS* System

When the application horizon gets wider, the search has to be extended to a larger and more distributed amount of data, heterogeneity related to data structures and semantics increases, the number of users can become much more sizeable and services based on a client-server architecture may not represent the most suitable and comprehensive solution. The system has to meet requirements against flexibility in terms of system configuration and user needs, pro-activeness in terms of responses to a changing environment, scalability in the number of users and reliability in terms of coping with an unstable environment. An approach that tackles these issues and that has attracted the attention of the research community especially during the last decade is that based on software agent technology [35] and multi-agent systems *(MASs)*[36]. The agent paradigm [37] proposes to design and implement systems where agents act autonomously while they can still aim at cooperating with other agents to achieve a complex task. *MASs* are those where agents are organised in societies where communication, interaction and coordination are possible. In our work we are concerned with intelligent information agents [38]. They are mainly characterised as holding intelligence (they manipulate information) and mobility (they can move through the network). The *MIKS* system hence comprises of a society of agents providing advanced information management functionalities. The advantage of the use in the *MIKS* infrastructure of intelligent and mobile software agents for the autonomous management and coordination of the integration and query processes have been introduced in [39]. Related work on the use of agents in information management systems (such as the Infosleuth project) is reported in section 6.

In the *MIKS* system, the exploitation of intelligent information agents (agents for short) improves the flexibility, as a number of system configurations are possible. As each component (i.e. agent) can be placed at whatever available host or data source and still provide the service it has been designed for, what in a client-server approach are the server side functionalities can be distributed and further moved while the system is running. On the client side, different solutions are possible in that agents allow for a number of devices to use the *MIKS* system, from desktop computers to PDAs to mobile phones. Agents can show to users services and interfaces according to the particular features of the device in use and deliver contents while users are moving wearing their portable devices.

Decomposing the system into independent and autonomous entities allow to set up configurations that respond to diverse level of granularity. According to the specific application scenario, the designer can choose whether to consider each component as a stand-alone entity or to group some components to create macro-entities whose components are joint as a whole. Throughout the design phase, we have chosen to define agents according to the step they come into play (when) and the service they provide (what) within the integration process.

Notice that, following [38], mediators agents are characterised by three capabilities:

1. translating among diverse ontological domain to build a uniform semantic view of a given domain of interest
2. decomposing and executing complex queries
3. fusing partial answers.

We can distinguish between pure integration capabilities (1) and querying capabilities (2 and 3). We have therefore split the system into two parts. The first is meant to provide the functionalities for supporting the integration process, while the second supports the querying phase. This distinction we have taken when designing the agent-based *MIKS* system. The resulting architecture comprises of two multi-agent systems (the first supporting the integration process and the other supporting the querying phase). The two multi-agent system are meant to be interoperable.

This has produced the side effect of splitting the functionalities of components (see Figure 1) according to the phase they participate in. For instance, wrappers are traditionally meant to support both the translation of the data source schema into a common data model and the execution of queries. In building the *MIKS* wrapper architecture there have been some driving goals. First, wrappers should be able to evolve over time. Wrappers should have the dynamic capability of adding new functionalities to their core behaviour. Secondly, wrappers should be flexible enough to be able to wrap sources which expose different services and capabilities, such as query processing capabilities. Thirdly, wrappers should have strong interaction capabilities with components requesting either information about the source or information contained in the source. Agents hold the properties such as pro-activeness and the ability to adjust their behaviour according to the environment that we find well-matched for designing and implementing wrappers. In our framework, these functionalities have been kept separated (namely the *Translator agent* and the *Query Support Agent*) because they come into play in realising two different services of the *MIKS* system. Defining the *MIKS* wrappers, we have found agent technology particularly useful. It defines a very flexible way of managing distributed components, from creation to migration to replication, that help shape the system configuration according to the particular operative scenario. The strong support for interoperability and conversation inherently distinguishing agents has represented an advantage as far as interactions and information exchange among the system components is concerned.

4.1 A Multi-agent System for Supporting the *MIKS* Integration Process

The MAS we have designed for integration purposes includes agents that support all of the four phases we have presented in section 2.2. Agents carry out activities for manipulating data to create information at higher levels of abstraction. Figure 3 depicts how agents are organised within the MAS.

The arrows highlight that the information can flow between the agents of different layers at either end of the arrows.

Starting from the left side of Figure 3, we find the *System Agent (SA).* It performs administrative tasks like system configuration monitoring and agent life-cycle management.

As the distribution of agents has the primary aim of distributing the workload to the *MIKS* system, the *SA* acts as the system monitor. It helps estimate the system status and workload and it is the one which decides when and where agent should migrate. This in turn realises the workload balancing mechanism. The *SA* manages the life-cycle of agents. It makes provision of an environment which is meant to host agents (code, data and status) and permits them the execution of their algorithms. The *SA* and more in general the agents, have been built using the JADE environment [40], a FIPA-compliant development tool [41].

The middle and right-side column show the agents and tools required for the integration process. Agents are grouped along three layers.

Rising up from the bottom of the middle column, in the *Data Source layer* we find the *Translator Agents (TAs).* A *TA* acts during the phase in which the source is recognised and its content has to be expressed using the ODL_I^3 language. The *TAs* have the following functionalities inside the *MIKS* system:

1. they can inherently adapt to a variety of data sources (relational DBMS, object-oriented DBMS, structured and unstructured documents)

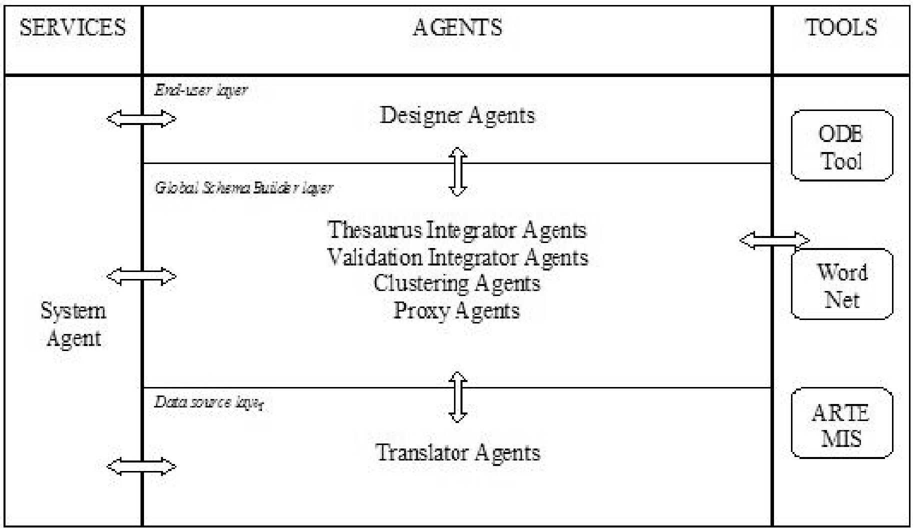

Fig. 3. Classification of agents for supporting the integration process

2. they build the ODL_{I^3} description of the underlying data source(s) and keep it up-to-date according to the committed changes. Whenever this happens, *TAs* communicate the changes to the agents belonging to the upper level, the *Global Schema Builder layer*
3. they provide a description of the services and query capabilities of the source and constantly report their status to the upper layer agents. Knowing the query capabilities of the sources is essential during the translation of the global query into the set of local queries in order to draw a query plan that is actually executable. Some interesting proposals facing the problem of querying information sources which provide different levels of query capabilities have been introduced in [27,42,12]. The aim is pruning efficiently out information sources that are unsuitable for a given query and generating executable query plans.

In the *Global Schema Builder layer* are grouped the agents that support the integration process. *Proxy agents* are information collectors. They acquire the local schemata created by the *TAs*. This knowledge base will be manipulated and enriched during the integration activities by the other *Global Schema Builder layer* agents and maintained in the corresponding *Proxy agents*.

After having acquired the set of local schemata, *Thesaurus Integrator Agents* (TIAs) and *Validation Integrator Agents* (VIAs) carry out the reasoning and inference activities required to semantically integrate data. *TIAs* are charged of extracting intensional intra/inter schema relationships. They have been subdivided into two types:

- *TIAs* that separately analyse the local ODL_{I^3} schemata in order to extract terminological and extensional relationships holding at intra-schema level
- *TIAs* that analyse the set or subsets of local ODL_{I^3} schemata in order to extract terminological relationships holding at inter-schema level.

VIAs are charged of interacting with the ODB-Tools to infer new relationships that hold at intra-schema level. Further, *VIAs* validate the whole set of relationships that have been discovered so far.

Together with designer-supplied relationships (as we will see when describing *Designer Agents*), these are used to build the *Common Thesaurus*, which represents the ontology of the given domain of integration. This completes the first two steps of the integration process. *Clustering Agents* generate the set of structural similar classes (*clusters*) and the corresponding global classes. This leads to the generation of global attributes and a mapping-table.

Notice that a common feature of the agents belonging to the *Global Schema Builder layer* is the interaction with the *MIKS* system tools. These are either knowledge bases (as the *WordNet* data base) or already existing applications (as *ARTEMIS* and *ODB-Tools*) developed in other research projects. All of the tools have sufficiently sophisticated interfaces to allow interaction with other applications. Thus, it is easy to make them interoperate with agents. This can be done by agentifying these applications (agents that are expressively being designed for exchanging data and calling functions of the tool) or by exposing the functionalities of the tools as web services.

At the end-user level, *Designer Agents (DAs)* are available. Their main functionality is providing designers a graphical user interface towards the *MIKS* system, its configuration and the data it integrates. They are much like the *MOMIS* SI-Designer module that provides the designer with a graphical user interface that shows the *Global Virtual View (GVV)*. Details can be found in [34,3]. The main difference is that *DAs* do not directly interact with the *MIKS* tools, but only with agents. A *DA* collects this information by interacting on a regular basis or on-demand with the agents presented so far. The system configuration is retrieved from the *SA* while an overview on sources, local schema, terminological and extensional relationships (inter- and intra-schema) are retrieved from the underlying layers. Further, *DAs* allow designers interacting with the other agents (and indirectly with the *MIKS* system tools), thus enabling control over the integration process (for instance, choosing the sources to be integrated and selecting the "best" clustering configuration among the proposed alternatives).

4.2 A Multi-agent System for Supporting Global Query Execution

The Global Schema gives users an integrated view over data that were previously scattered over different places and applications and had to be accessed separately both from a logical and physical viewpoints. This is a very powerful service given the initial heterogeneous information environment. Nevertheless, a comprehensive attitude towards information management includes not only displaying the whole range of data, but also a way of submitting queries in order to select particular set of data. This is very useful if we want to restrict the focus of our analysis to what is really important for our purposes. The *MIKS* system allows users to submit queries over the Global Schema (global queries). Similarly to other semantic approaches, the querying phase consists of three steps:

1. semantic optimization
2. query plan execution
3. fusion of local, partial answers.

We refer the reader to [14] for more details about the techniques deployed to realise these three steps. We have designed a MAS for supporting the whole phase of global query solution. Agents perform activities for manipulating global queries to create queries at lower level of abstraction (local queries) that are hence executable on data sources. Local answers have then to be synthesised into a global answer. Notice while the integration process is essentially a one-way bottom-up information flow starting from the source contents and ending up with the generation of a GVV, the querying phase is a two-way process: top-down when users submit queries over the GVV and bottom-up when local answer are made available and have to be merged to compose the global answer. Our MAS reflects the nature of the querying process. Figure 4 illustrates the organisation of agents. The arrows highlight that the information can flow between the agents of different layers at either end of the arrows. The right-side and left-side columns are the same we have just described above for the MAS supporting information integration. We will focus on the middle column. This time we will proceed from the top layer down.

In the end-user layer we find *User Agents* (*UAs*). This reflects the shift in the type of system users: information integration requires interaction with designers, query processing is expressively made available so that users can submit actually access data they are interested in.

In the setting of distributed services available on the network, users do not usually get access to the desired services directly at the facility that provides them. They use applications running on hosts they are using. Besides the traditional wired way of getting connected to local or wide networks or the Internet, in recent years the usage of wireless devices has become more and more recurrent. These kinds of mobile devices are characterised by strong limitations in terms of battery autonomy, processing power and bandwidth if compared to desktop and personal computers. These limitations have to be taken into account when interfacing the *MIKS* system to mobile users. Falling back on client-side applications could not be the most effective solution and sometimes it could be even impracticable to install them (especially when needs arise on the fly).

User-side applications offer an interface to some underlying capabilities that ultimately allow the access to services residing on some remote hosts on the network. The wide acceptance of the World Wide Web (simply Web) [44] as

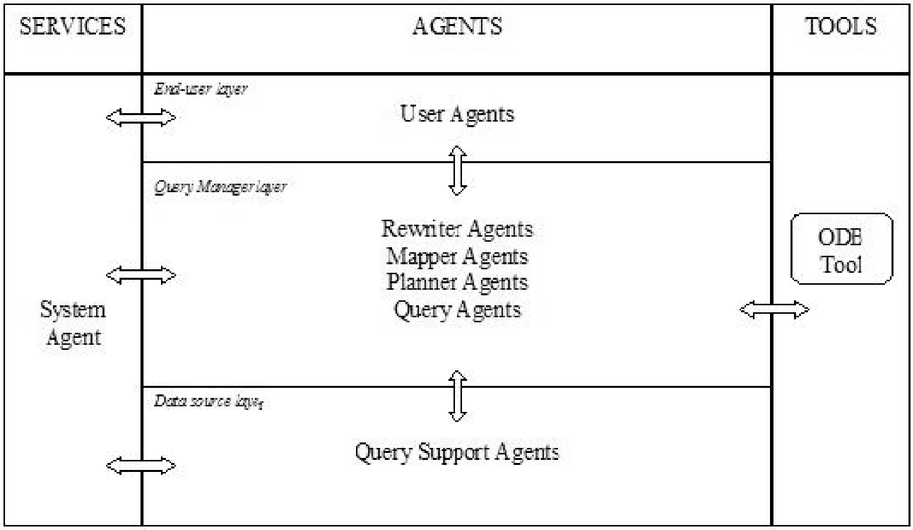

Fig. 4. Classification of agents for supporting the querying phase

main application for information delivering makes web browsers a very appealing means for supplying user interfaces whatever the underlying services are. Often a simple web browser is not sufficient to enable users the usage of more complex services. An example is the provision of web-based database connectivity that requires the usage of *JDBC* drivers [45]. A detailed discussion of this topic can be found in [46]. In general, for web-based services specific capabilities have to be added as client side applications and web browsers have to get hold of them on the fly. This is the reason modern web browsers are Java enabled. There seems to be two major approaches to the provision of complex services on the Web. The first foresees code downloading to user hosts in order to enable the access to desired services. It is the case of *Java applets* ([43]) . The second creates a process whenever a user requests a service. The process serves that specific request and then terminate. It is the case of *Java servlets* ([47]). We will not take into consideration this second choice as it refers to traditional client-server architectures. *Java applets* can be seen as a first step towards agent technology: they move to the user host on-demand and execute some task on his/her behalf. For instance, current approaches to distributed database access based on *applets* have become increasingly popular in the last decade. As Papastavrou et al. show, a drawback of the use of *applet*-based service interfaces in this setting is that not only code has to be downloaded (that is pretty much comprehensible and inescapable) but at least an additional *JDBC* driver has to be downloaded and initiated. This process results in heavy procedures that could be not feasible in the case of devices with bounded resources (especially low memory space and low bandwidth connectivity). Agent technology is a way to overcome these limitations, adapting the fruition of the provided services to the particular features and capabilities of the worn device. Defining *UAs* means in turn defining a new kind of lightweight interfaces to remote services. *UAs* are responsible for collecting user queries, user preferences and device features in order to build user profiles. User queries are collected as OQL_{I^3} statements. As future work, we are going to provide users with a more advanced and powerful interface, like the one proposed in [48]. User profiles serve in turn to know which priorities have to be deemed when assessing the relevance of the information and how they can be transmitted and displayed to end users. Further, exploiting user agents has another important facet: users have to keep connections up only for a limited amount of time, i.e. the time required to send the agent code and data and the time required to collect results. Altogether, the user agent acts as a filter against the data they are retrieved by the *MIKS* system. The other modules of the *MIKS* system cannot execute this functionality as they are meant to provide the overall outcome of the integration and querying processes (*GVV* and global answer). The filtering has to be done according to user preferences and device capabilities. User agents endow with a realistic mechanism to pledge a certain quality of service.

In the *Query Manager layer* are grouped the agents that carry out global query decomposition and partial answer merging. *Rewriter Agents (RAs)* operate on the query by exploiting the semantic optimisation techniques [49,50, 51] supported by ODB-Tools [29,52,53] in order to reduce the query access plan cost. The query is rewritten incorporating any possible restriction which is not present in the global query but is logically implied by the Global Schema (class descriptions and integrity rules).

Mapper Agents (MAs) express the rewritten global query in terms of local schemas. Thus, a set of sub-queries for the local information sources is formulated. To this end, *MAs* dialogue with *Proxy Agents* that hold the knowledge about mapping table, global and local schema. In order to obtain each local query, the mediator checks and translates every predicate in the where clause.

Planner Agents (PAs) are charged to take the set (or subsets) of local queries and produce the executable query plan. The goal of *PA* is to establish how much parallelism and workload distribution is possible. Considering that queries are assigned to *Query Agents (QAs)* that move to local sources, creating a plan means trying to balance different factors:

- how many queries have to be assigned to each single *QA*
- which sources and in which order each *QA* has to follow in order to solve the assigned queries or to fuse partial results.

The choice of the number of query agents to use can be determined by analyzing each query. In some cases, it is better to delegate the search to a single query agent, which performs a “trip” visiting each source site: it can start from the source that is supposed to reduce the further searches in the most significant way, then continue to visit source sites, performing queries on the basis of the already-found information. In other cases, sub-queries are likely to be quite independent, so it is better to delegate several query agents, one for each source site: in this way the searches are performed concurrently with a high degree of parallelism. This allow for decentralisation of the computational workload due to collecting local answers and fuse them into the final global answer to be carried to the user. Future work will take into consideration more advanced techniques as reported for instance in [54,55].

QAs move to local sources where they pass the execution of one or more queries to *Query Support Agents (QSA)*. A number of advantages are implied by moving to local sources. First, by moving locally to the source site, a query agent permits to significantly save bandwidth, because it is not necessary to transfer a large amount of data, but the search computation is performed locally where the data resides. Second, users can queries also sources that do not have continuous connections: the query agent moves to the source site when the connection is available, performs locally the search even if the connection is unstable or unavailable, and then returns as soon as the connection is available again. Finally, this fits well mobile computing, where mobile devices (which can host user applications, agents and/or sources) do not have permanent connections with the fixed network infrastructure.

QSA afford translation services between the OQL_I^3 language and the native query language of the data source. This step is required to make queries executable by local information management system.

When the local answer is available, the corresponding *QA* has to map these data (whose structure follows the local schema) to the global schema. For doing this, the *QA* interacts with *Proxy Agents* to know the set of mapping rules related to the source. For instance, given that the attribute `name` in the Global Schema maps to the attributes `lastname` and `firstname` in the local source, the *QSA* has to put together the values of the two attributes in order to obtain the value for the global attribute `name`.

4.3 Further Agent Roles

Besides supporting the integration process and the querying phase, agents are suitable within the *MIKS* system for a number of complementary and precious tasks. In particular, we highlight here the extensional assertions on the data sources and the searching for new potential data sources.

Agents managing extensional knowledge. As we have already stated, in the implementation of *MOMIS*, there is no verification tool to monitor the truth of the extensional assertions on the data sources. We have also seen that methods to check and enforce integrity constraints over a distributed set of information sources have been defined [24,56]. We claim agent technology can be efficiently deployed to develop a framework for constraint verification in a mediator environment. Intuitively we can define *Check Extensional Assertion Agents (CEAAs)* for each data source. In order to carry out its task, a *CEAA* should know the assertions concerning the classes of the source and the operations on the source data (held by *Proxy Agents*) that can potentially lead to a violation of the extensional assertions. Whenever such an operation occurs, the *CEAA* communicate and coordinate with the other *CEAAs* to check the validity of the extensional assertions. Intuitively, such activity requires some transfer of data among data sources. Our goal is to integrate existing approaches as [24,56] within our framework. We leave these new refinements of our framework as future work.

Searching for new data sources. The set of remote sources to be integrated into the *MIKS* global view can be made up by sources autonomously deciding to grant the system their knowledge. On the other hand, at any moment a need to increase the number of the sources may arise in order to enhance the quantity and quality of available data. In such a case, the system has to search for new potentially interesting sources in a certain domain (e.g. Internet). The search may exploit in this case intelligent and mobile agents. For doing so, the system has to form the list of sources to be visited. This could be done by means of a domain search according to the ontology contained into the *Common Thesaurus* and held by *Proxy Agents*. Then, the *SA* creates and coordinates a set of agents to be deployed. The *SA* allots the addresses of the list among some *Hunter Agents (HAs)*. Different policies can be enforced to control the activity of the *HAs* and to keep track of the state of the search. Once arrived at the assigned site, a *HA* has to introduce itself, clarify the goal of its visit and negotiate the availability of the source to grant access to a part or the whole of its content. If the result of the negotiation phase is positive, a *TA* has to be sent to the source.

5 Agent Interaction Patterns

As we have seen, diverse types of agents populate the two MASs of the *MIKS* system. During the design phase, the main functionalities of the system have been decomposed into a number of smaller services assigned to agents. Agents can move, thus relocating the services they provide. Figure 5 and 6 show the two extreme configurations the system can operate:

1. the former depicts how a client-server-like configuration is realisable,
2. the latter depicts a fully distributed architecture.

Arrows depict the most important interactions that occur among the system components.

The *MIKS* agents interact in order to achieve the overall goals of the *MIKS* system. Many interaction patterns are feasible as information has to flow from one agent to another. This can happen among agents either belonging to the same MAS or belonging to two different MAS. In the end, splitting the system into two MASs has served the purpose of keeping the architecture design clean avoiding waste of resources or overlapping in tasks among the agents.

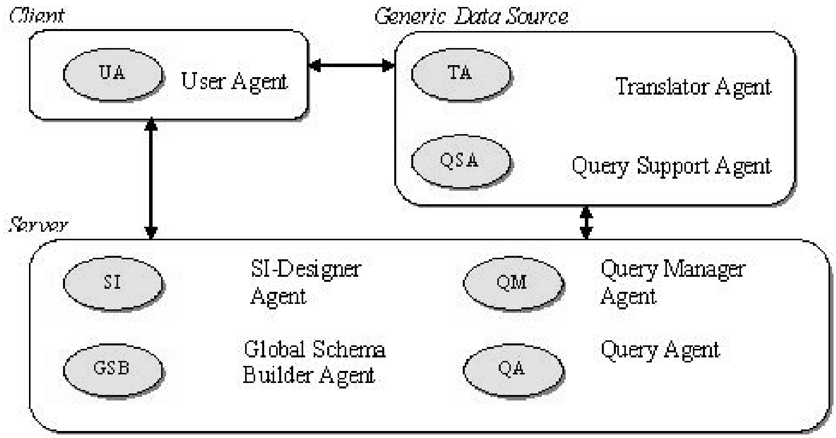

Fig. 5. The *MIKS* agents organised into a client/server configuration

In this section we will present some possible interaction patterns involving the agents we have presented so far. The purpose it to convey the reader the dynamics of agent interactions that can happen within the *MIKS* agent architecture. For the sake of clarity, we will refer to agents specifying their full name.

In a possible real scenario (Figure 7), the *System Agent* spawns the agents required for the particular activities to be carried out.

At the beginning of the integration process, it interacts with the *Designer Agent* which reports which source have to be integrated. The list of interesting data sources can be available or filled with the suggestions of *Hunter Agents*. Then, the *System Agent* spawns *Translator Agents* in order to wrap the desired data sources. Arrows are bi-directional to mean that agents engage in dialogue in order to exchange information and that interactions may happen not only during initialisation but whenever need arises and throughout the whole system life.

During the integration process there are two main interaction patterns:

1. agents communicating with *Proxy agents* for retrieving data to be manipulated and storing enriched information
2. *Designer Agent* communicating with agents belonging to the *Global Schema Builder layer* in order to interact during the various steps of the integration process.

Figure 8 show the complete set of interactions. Notice the *User Agents* can request and obtain from *Proxy Agents* the *Global Virtual View*, while *Hunter*

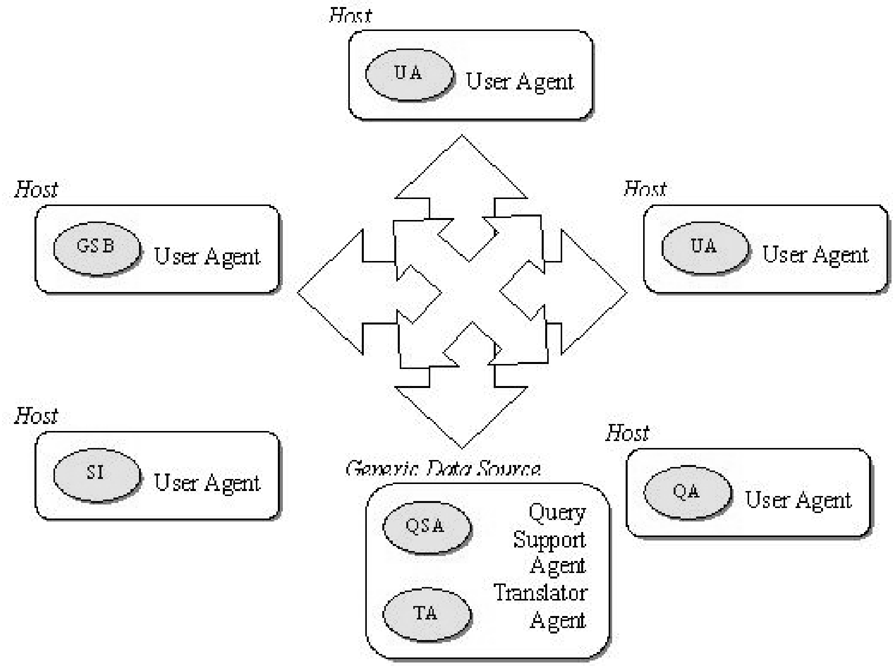

Fig. 6. The *MIKS* agents organised into a fully distributed configuration

Agents can update the reference ontology according to which the search has to be carried out.

The query solving phase foresees a more structure interaction pattern. Global queries are posed by users and then manipulated so as to be executable at the *Data Source layer*. The order of interactions follows the sequence of step required to decompose and solve the query (Figure 9):

1. *User Agents* directly transmit the submitted global queries to the *Rewriting Agents*
2. *Rewriting Agents* dialogue with *Proxy Agents* to retrieve information about the Global and Local Schema and the mapping table
3. once the queries have been rewritten, *Rewriting Agents* send it to the *Mapper Agents*
4. *Mapper Agents* decompose the global queries into a set of local queries
5. once set of local queries is available, *Mapper Agents* send it to the *Planner Agents*
6. *Planner Agents* generate a query execution plan
7. together with the *SA* organises a group of *Query Agents*
8. *Query Agents* interact with *Query Support Agents* to actually execute the queries at data source level

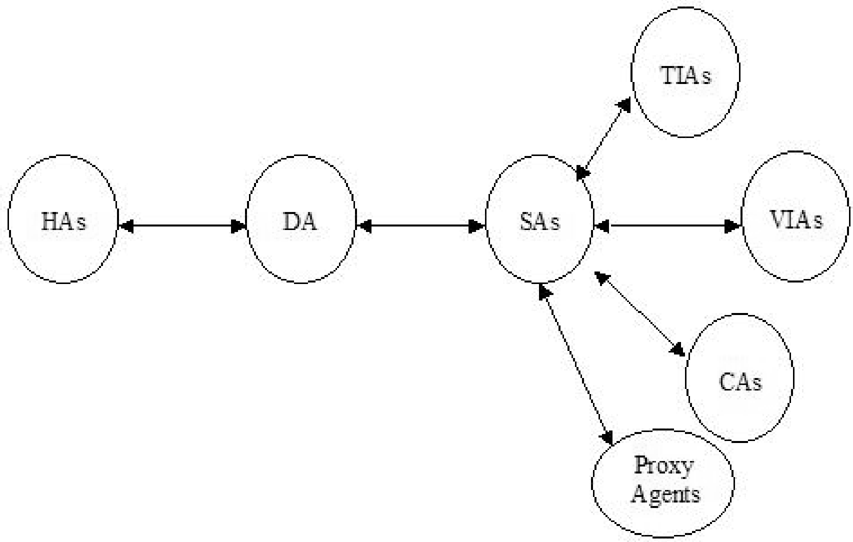

Fig. 7. System initialisation

9. *Query Agents* keep the *Planner Agents* up-to-date, exception should arise and modifications to the plan should be undertaken
10. *Query Agents* synthetise the global answer and send it to the *User Agents.*

These interaction patterns show how our MAS architecture provides for flexibility and can adapt to dynamic environment. Although we have illustrated the *MIKS* system as composed of two MASs, they have to be considered as open system in that agents belonging to either MASs can engage in interactions with all of the other agents. More in general, a system can comprise many MASs of both kinds, building a complex information infrastructure, where many agents interact in order to integrate and query different type of distributed sources and ontologies.

6 Related Work

In the area of heterogeneous information integration, many projects based on a mediator architecture have been developed. The mediator-based TSIMMIS project [57] follows a 'structural' approach and uses a self-describing model (OEM) to represent heterogeneous data sources, the MSL (Mediator Specification Language) rule to enforce source integration and pattern matching techniques to perform a predefined set of queries based on a query template. Differ-

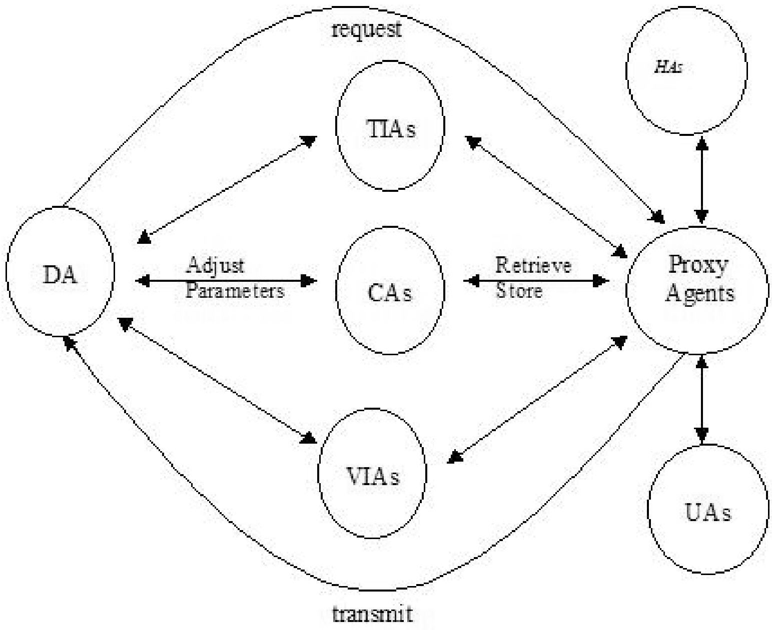

Fig. 8. A possible interaction pattern for the integration process

ently from our integration approach proposal, in TSIMMIS only the predefined queries may be executed and for each source modification a manually mediator rules rewriting must be performed.

The GARLIC project [58] builds up on a complex wrapper architecture to describe the local sources with an OO language (GDL), and on the definition of Garlic Complex Objects to manually unify the local sources to define a global schema.
The SIMS project [59] proposes to create a global schema definition by exploiting the use of Description Logics (i.e., the LOOM language) for describing information sources. The use of a global schema allows both GARLIC and SIMS projects to support every possible user queries on the schema instead of a predefined subset of them.
The Information Manifold system [16] provides a source independent and query independent mediator. The input schema of Information Manifold is a set of descriptions of the sources. Given a query, the system will create a plan for answering the query using the underlying source descriptions. Algorithms to decide the useful information sources and to generate the query plan have been implemented. The integrated schema is defined mainly manually by the designer, while in our approach it is tool-supported.
Infomaster [60] provides integrated access to multiple distributed heterogeneous

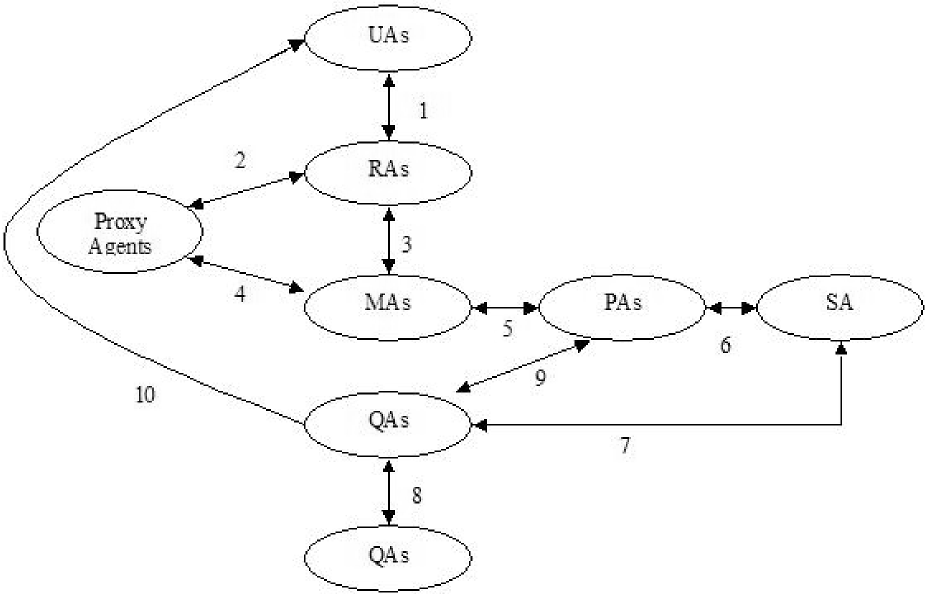

Fig. 9. A possible interaction pattern for the query solving phase

information sources giving the illusion of a centralized, homogeneous information system. It is based on a global schema, completely modelled by the user, and a core system that dynamically determines an efficient plan to answer the user's queries by using translation rules to harmonize possible heterogeneities across the sources. The main difference of these project w.r.t. our approach is the lack of a tool aid-support for the designer in the integration process.

As for the multi-agent system community, some work has been done in the direction of integration systems. For its similarities with the *MIKS* system, a particular mention deserves the Infosleuth system. Infosleuth is a system designed to actively gather information by performing diverse information management activities. In [48] the Infosleuth's agent-based architecture has been presented. InfoSleuth agents enable a loose integration of technologies allowing: (1) extraction of semantic concepts from autonomous information sources; (2) registration and integration of semantically annotated information from diverse sources; and (3) temporal monitoring, information routing, and identification of trends appearing across sources in the information network.

While addressing to the same research area, the *MIKS* system and Infosleuth system present slightly different features.

First of all, the scope of the two systems appears to be different. *MIKS* aims at building ontologies related with the integration domain, and at providing a unified view. Query are to be posed as global ones on the GVV. Infosleuth bases its data analysis on given ontologies (rather than building them) and provides visibility of data related only to the specified queries.

Secondly, *MIKS* is meant to provide a two step managing of data, i.e integration and if required also querying, while Infosleuth is devoted to directly query an information source, once an ontology has been explicitly given by humans.

In fact, the integration process differs in that *MIKS* aims at building ontologies directly from the content of the data source, inferring the relationships within the collection of concepts. Infosleuth seems to be more an ontology-driven query engine. Ontologies can be directly provided by users or designers in order to be used during the mapping process. Ontologies can be stored in some server facility for further reuse.

Thirdly, *MIKS* is characterised by strong reasoning capabilities that are meant to tackle the problem of semantic integration of concepts belonging to multiple ontologies (i.e. how we can discover that two objects belonging to different schema refer to the same real-world concept).

Further, as a consequence of these differences, the agent architectures of the two systems are quite different. Agents with common functionalities (translator agents/query support agents and resource agents, user agents, query agents) exist even though they reflect the two distinct approaches. One last remark concerns the presence of service agents in Infosleuth (in particular broker agents). The *MIKS* system provides the same services in a centralised manner with the *SA*.

Another experience is the RETSINA multi-agent infrastructure for in-context information retrieval [62]. In particular the LARKS description language [61] is defined to realize the agent matchmaking process (both at syntactic and semantic level) by using several different filters: Context, Profile, Similarity, Signature and Constraint matching.

References

1. C. Batini, M. Lenzerini, S. B. Navathe. A Comparative Analysis of Methodologies for Database Schema Integration. In ACM Computing Surveys, 18(4):323–364, 1986.
2. S. E. Madnick. From VLDB to VMLDB (Very MANY Large Data Bases): Dealing with Large-Scale Semantic Heterogenity. In F. van Harmelen, editor, Proceedings of the 15th European Conference on Artificial Intelligence (ECAI 2002), IOS Press, Amsterdam, 38–42, 2002.
3. D. Beneventano, S. Bergamaschi, F. Guerra, M. Vincini. The MOMIS approach to Information Integration. In proceedings of the Conference on Enterprise Information Systems (ICEIS01) Setúbal Portugal 2001.
4. S. Bergamaschi, S. Castano, D. Beneventano, M. Vincini. Semantic Integration of Heterogenous Information sources. Journal of Data and Knowledge Engineering, 36(3):215–249, Elsevier Science B.V. (North-Holland), Amsterdam NL, 2001.
5. N. R. Jennings, M. J. Wooldridge Applications of Intelligent Agents. In Nicholas R. Jennings and Michael J. Wooldridge(editors), Agent Technology: Foundations Applications and Market, 3–28, Springer-Verlag: Heidelberg Germany, 1998.
6. N. M. Karnik, A. R. Tripathi. IEEE Concurrency, Design Issues in Mobile-Agent Programming Systems, 6(3), 52–61, 1998.

7. G. Cabri, L. Leonardi, F. Zambonelli. Agents for Information Retrieval: Issues of Mobility and Coordination. Journal of Systems Architecture, 46(15),1419–1433, 2000.
8. R. G. G. Cattell. The Object Database Standard: ODMG93. Morgan Kaufmann Publishers, San Mateo (CA), 1994.
9. S. Bergamaschi, G. Cabri, F. Guerra, L. Leonardi, M. Vincini, F. Zambonelli. Exploiting Agents to Support Information Integration. International Journal on Cooperative Information Systems, 11(3), 2002.
10. Y. Papakonstantinou, H. Garcia-Molina, J. Widom. Object Exchange across Heterogeneous Information Sources. In P. S. Yu and A. L. P. Chen (editors) proceedings of the 11th Conference on Data Engineering, IEEE Computer Society, Taipei Taiwan, 1995.
11. Y. Arens, C. Y. Chee, C. Hsu, C. A. Knoblock. Retrieving and Integrating Data from Multiple Information Sources. International Journal of Intelligent and Cooperative Information Systems, 2(2), 127–158, 1993.
12. M. T. Roth, P. M. Schwarz. Don't Scrap It Wrap It! A Wrapper Architecture for Legacy Data Sources. In Proceeding of VLDB, 266–275, 1997.
13. R. Hull,R. King et al. ARPA I^3 Reference Architecture, 1995. Available at http://www.isse.gmu.edu/I3_Arch/index.html.
14. D. Beneventano, S. Bergamaschi, F. Mandreoli. Extensional Knowledge for semantic query optimization in a mediator based system. In Proceeding of FMII, 2001.
15. Oliver M. Duschka. Query Optimization Using Local Completeness. In proceedings of AAAI, 249–255, 1997.
16. A. Y. Levy, A. Rajaraman, J. J. Ordille. Querying Heterogeneous Information Sources Using Source Descriptions. In proceedings of VLDB, 251–262, 1996.
17. O. M. Duschka, M. R. Genesereth. Answering Recursive Queries Using Views. In proceedings of the Sixteenth ACM SIGMOD Symposium on Principles of Database Systems, 1997.
18. E. Lambrecht, S. Kambhampati, S. Gnanaprakasam. Optimizing Recursive Information-Gathering Plans. In proceedings of the International Joint Conference on Artificial Intelligence IJCAI, 1999.
19. R. Pottinger, A. Y. Levy. A Scalable Algorithm for Answering Queries Using Views. In proceedings of 26th International Conference on Very Large Data Bases September 10–14 2000 Cairo Egypt, 484–495, Morgan Kaufmann, 2000.
20. J. L. Ambite, C. A. Knoblock. Flexible and Scalable Query Planning in Distributed and Heterogeneous Environments. In proceedings of the 4th Int. Conf. on Artificial Intelligence Planning Systems AAAI, 1998.
21. C. A. Knoblock. Planning Executing Sensing and Replanning for Information Gathering. in proceedings of the Fourteenth International Joint Conference on Artificial Intelligence IJCAI, 1995.
22. Y. Papakonstantinou, S. Abiteboul, H. Garcia-Molina. Object Fusion in Mediator Systems In proceedings of the VLDB International Conference, Bombay India, 1996.
23. R. Yerneni, Y. Papakonstantinou, S. Abiteboul, H. Garcia-Molina. Fusion Queries over Internet Databases. Advances in Database Technology - EDBT'98 6th International Conference on Extending Database Technology, Lecture Notes in Computer Science, 1377, 1998.
24. S. S. Chawathe, H Garcia-Molina, J. Widom. A Toolkit for Constraint Management in Heterogeneous Information Systems. ICDE, 56–65, 1996.

25. Kim W. , F. H. Lochoisky . Object-Oriented concepts Databases and Applications. Addison-Wesley Publishing Co., ACM Press, New York NY USA, 1989.
26. Object Management Group. CORBA: Architecture and Specification. 1995.
27. C. -C. K. Chang, H. Garcia-Molina. Mind Your Vocabulary: Query Mapping Across Heterogeneous Information Sources. In proceedings of ACM SIGMOD, 335–346, 1999.
28. S. Bergamaschi, S. Castano, M. Vincini. Semantic Integration of Semistructured and Structured Data Sources. SIGMOD Records, 28(1), 1999.
29. D. Beneventano, , S. Bergamaschi , S. Lodi , C. Sartori. Consistency Checking in Complex Object Database Schemata with Integrity Constraints. IEEE Transactions on Knowledge and Data Engineering, 10, 567–598, 1998.
30. S. Bergamaschi , B. Nebel. Acquisition and Validation of Complex Object Database Schemata Supporting Multiple Inheritance. Journal of Applied Intelligence, 4, 185–203, 1994.
31. D. Beneventano , S. Bergamaschi , C. Sartori. Semantic Query Optimization by Subsumption in OODB. Flexible Query Answering Systems, Datalogiske Skrifter series, 62, Roskilde Denmark, 1996.
32. S. Castano , V. De Antonellis. Deriving Global Conceptual Views from Multiple Information Sources. In proceedings of ER'97 Preconference Symposium on Conceptual Modeling Historical Perspectives and Future Directions, 1997.
33. S. Castano , V. De Antonellis , S. De Capitani Di Vimercati. Global Viewing of Heterogeneous Data Sources. tkde, 13(2), 277–297, 2001.
34. I. Benetti , D. Beneventano , S. Bergamaschi , A. Corni, F. Guerra , G. Malvezzi. SI-Designer: a tool for intelligent integration of information. Int. Conference on System Sciences (HICSS2001), 2001.
35. Bradshaw J. (eds.). Hand of Agent Technology. AAAI/MIT Press, 2000.
36. M. Wooldridge. An Introduction to Multiagent Systems. Wiley, 2002.
37. Y. Shoham. Agent-oriented programming. Artificial Intelligence, 60, 51–92, 1993.
38. M. Klusch. Information agent technology for the Internet: A survey journal. Data and Knowledge Engineering, 36(3), 337–372, 2001.
39. S. Bergamaschi , G. Cabri , F. Guerra , L. Leonardi , M. Vincini , F. Zambonelli. Supporting Information Integration with Autonomous Agents. In proceedings of the CIA, 88–99, 2001.
40. Telecom Lab. Jade - Java Agent Development Environment.
41. FIPA. Specifications, 1997.
42. L. M. Haas , D. Kossmann , E. L. Wimmers , J. Yang. Optimizing Queries Across Diverse Data Sources. In proceedings of VLDB, 276–285, 1997.
43. Sun Microsystem. Java Applet. http://java.sun.com/applets/.
44. World Wide Web Consortium. Standards for the World Wide Web address. www.w3c.org.
45. Sun Microsystem. Java official web-site. www.java.sun.com.
46. S. Papastavrou , G. Samaras , E. Pitoura. Mobile Agents for WWW Distributed Database Access. ICDE, 228–237, 1999.
47. Sun Microsystem. Java Servlet. http://java.sun.com/servlets/.
48. M. Nodine , J. Fowler , T. Ksiezyk , B. Perry , M. Taylor , A. Unruh. Active informaiton Gathering in Infosleuth. International Journal of Cooperative Information Systems, 9(1-2), 3–27, 2000.
49. D. Beneventano , S. Bergamaschi , A. Garuti , C. Sartori , M. Vincini. ODB-QOptimizer: un Ottimizzatore Semantico di interrogazioni per OODB. Terzo Convegno Nazionale su Sistemi Evoluti per Basi di Dati - SEBD95 Salerno, 1995.

50. D. Beneventano , S. Bergamaschi , C. Sartori , M. Vincini. ODB-QOPTIMIZER: A Tool for Semantic Query Optimization in OODB. Int. Conference on Data Engineering - ICDE97, 1997.
51. S. Shenoy, M. Ozsoyoglu. Design and Implementation of a Semantic Query Optimizer. IEEE trans. Data and Knowledge Engineering, 1(3), 344–361, 1989.
52. D. Beneventano S. Bergamaschi C. Sartori and M. Vincini. ODB-Tools: A Description Logics Based Tool for Schema Validation and Semantic Query Optimization in Object Oriented Databases. in proceedings of the conference on Data Engineering ICDE'97 Birmingham UK, 1997.
53. D. Beneventano , S. Bergamaschi , C. Sartori , M. Vincini. ODB-QOptimizer: a Tool for Semantic Query Optimization in OODB. Fifth Conference of the Italian Association for Artificial Intelligence (AI*IA97) Rome 1997 LNAI 1321, 1997.
54. U. Leser. Query planning in Mediator Based Information Systems school. Tu Berlin, 2000.
55. J. L. Ambite, C. A. Knoblock. Flexible and Scalable cost-based query planning in mediators: A transformational approach. Artificial Intelligence, 118(1-2)m 115–161, 2000.
56. A. Gupta , J. Widom. Local verification of global integrity constraints in distributed databases. In proceedings pf of ACM SIGMOD, 49–58, 1993.
57. S. Chawathe , H. Garcia-Molina. Hammer, K. Irel, , Y. Papakostantinou , J. Ullman , J. Widom The TSIMMIS project: Integration of heterogeneous information sources. IPSJ Conference Tokyo Japan, 1994.
58. M. J. Carey, L. M. Haas, P. M. Schwarz, M. Arya, W. F. Cody, R. Fagin, M. Flickner, A. W. Luniewski, W. Niblack, D. Petkovic, J. Thomas, J. H. Williams, E. L. Wimmers. Object Exchange Across Heterogeneous Information Sources. Stanford University 1994.
59. Y. Arens , C. A. Knoblock , C. Hsu. Query Processing in the SIMS Information Mediator. Advanced Planning Technology, 1996.
60. M. R. Genesereth , A. M. Keller , O. Duschka. Infomaster: An Information Integration System. In proceedings of ACM SIGMOD, 1997.
61. K. P. Sycara , M. Klusch , S. Widoff , J. Lu. Dynamic Service Matchmaking Among Agents in Open Information Environments. SIGMOD Record, 28(1), 47–53, 1999.
62. K. Sycara. In-context information management through adaptative collaboration of intelligent agents. Intelligent Information Agents editor M. Klusch, 78–99, 1999.
63. D. Beneventano, S. Bergamaschi, S. Castano, A. Corni, R. Guidetti, G. Malvezzi, M. Melchiori, M. Vincini Information Integration: The MOMIS Project Demonstration. VLDB 2000, in proceedings of 26th International Conference on Very Large Data Bases, 2000, Egypt.

PICSEL and Xyleme: Two Illustrative Information Integration Agents

Marie-Christine Rousset and Chantal Reynaud

University of Paris Sud - CNRS (L.R.I) & INRIA (Futurs)
LRI, Building 490, University of Paris-Sud, F-91405, Orsay Cedex, France
{mcr,cr}@lri.fr

Abstract. An information integration agent provides a uniform query interface to a collection of distributed and heterogeneous information sources, giving users or other agents the illusion that they interrogate a centralized and homogeneous information system. In this chapter we focus on integration information agents that follow a mediator approach. A mediator is based on the specification of a single *mediated schema* describing a domain of interest, and on a set of *source descriptions* expressing how the content of each source available to the system is related to the domain of interest. These source descriptions, also called *mappings* because they model the correspondence between the mediated schema and the schemas of the data sources, play a central role in the query answering process. We present two recent information integration agents, namely PICSEL and Xyleme, which are illustrative of two radically different choices concerning the expressivity of the mediated schema.[1]

1 Introduction

The explosion of the number of information sources that are available on-line has raised the need for building information integration agents. An information integration agent provides a uniform query interface to a collection of distributed and heterogeneous information sources, giving users the illusion that they interrogate a centralized and homogeneous information system. Given a query expressing the information looked for by a user, an information integration agent takes in charge, in the place of the user, the tasks of finding the relevant information sources, interacting with each of these sources to get the useful data, and combining them to compute the appropriate final answer.

In the recent years, considerable research has been done about information integration both for structured and semistructured data sources, and several information integration systems have been implemented (e.g., [28,25,18,12,2,32, 33,5]). Most of them follow a declarative approach based on the specification of a single *mediated schema* describing a domain of interest, and on a set of *source descriptions* expressing how the content of each source available to the system is related to the domain of interest. These source descriptions, also called *mappings*

[1] This chapter is a survey and a summary of two publications [18,13].

M. Klusch et al. (Eds.): Intelligent Information Agents, LNAI 2586, pp. 50–78, 2003.

because they model the correspondence between the mediated schema and the schemas of the data sources, play a central role in the query answering process. The users queries are posed in terms of the mediated schema, and are translated by the data integration system using the source descriptions into a set of queries that can be executed against the data sources.

The general architecture of an information integration agent following a mediator approach is given in Figure 1.

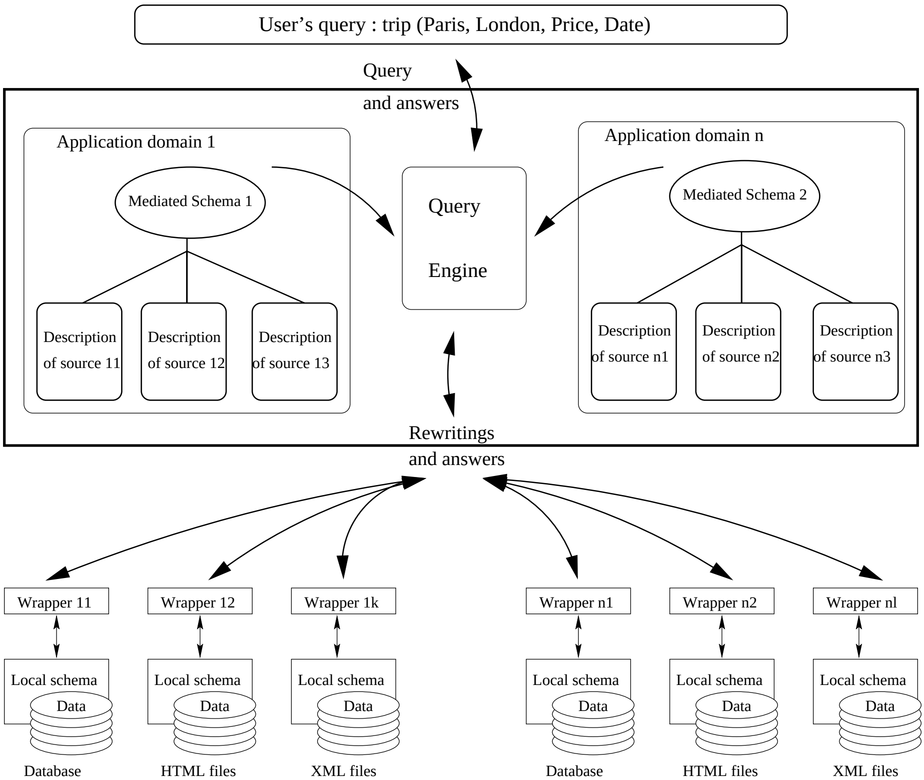

Fig. 1. Architecture of a mediator-based information integration agent

The existing mediator-based information integration agents can be distinguished according to (1) the languages used for modeling the mediated schema and the source descriptions, (2) the type of mappings between the mediated schema and the schemas of the sources.

Global as Views versus Local as Views: Concerning the last point, information integration agents can be related to two main approaches for modelling inter-schemas correspondence: *Global As Views (GAV)* and *Local As Views (LAV)* . The *GAV* approach has been the first one to be proposed and comes from the Federated Databases world. The mediated schema is defined in function of the schemas of the sources to integrate, i.e., each relation of the mediated schema is defined as a view on the relations of the sources schemas. The advantage of this approach is the simplicity of query reformulation which simply consists of replacing each atom of the query by its definition in terms of the relations of the sources schemas. Its drawback is its lack of flexibility with respect to the addition or deletion of sources to the mediator: adding (or deleting) a source to the mediator may affect the definitions of all the relations of the mediated schema. The *LAV* approach is dual and has opposite advantages and drawbacks. It consists of describing the contents of the sources in function of the mediated schema. In such an approach, adding a new source is quite straighforward because each source is defined independently of each other. It simply requires to add some formulas to model the content of the new source as queries over the mediated schema. The price to pay for this flexiblity is the difficulty of the query answering processing. Since both the users queries and the views describing the sources contents are expressed in terms of the mediated schema, the reformulation of the users queries in terms of the source relations cannot be done as a simple query unfolding as for the *GAV* approach. This requires a more complex process of *rewriting queries using views* (see [19] and [8] for more details on the problem of answering queries using extensions of views).

Relational versus object-oriented mediated schema: The most representative information integration systems of the relational approach are: *Razor* ([15]), *Internet Softbot* ([14]), *Infomaster* ([16]) et *Information Manifold* ([25]). They all follow a *LAV* approach. The *Razor* and *Internet Softbot* systems use DATALOG (without recursion) for modeling the mediated schema, the views describing the sources contents and the users queries. *Infomaster* and *Information Manifold* are based on extensions of DATALOG. *Infomaster* exploits integrity constraints in addition of DATALOG rules. *Information Manifold* extends DATALOG by allowing that some predicates used in the rules are concepts defined by using description logics constructors. This hybrid formalism is in fact a precursor of the CARIN language considered in this chapter. The *HERMES* ([30]) system is a system of federated databases that can be seen as a mediator following a *GAV* approach and based on a Prolog-like query language with annotations for handling uncertainty and time.

The most representative information integration systems of the object-oriented approach are: *TSIMMIS* ([28]), *SIMS* ([4,3]) *OBSERVER* ([27]) and *MOMIS* ([5]). *TSIMMIS* is based on the object-oriented language OEM for describing the mediated schema and the views, and on the OEM-QL query language. It follows a *GAV* approach. The *SIMS* and *OBSERVER* systems use a description logic for modeling the mediated schema, the views and the queries. They follow a *LAV* approach since the content of the source is described in function of the concepts described in the mediated schema. *SIMS* uses LOOM ([26]) while

OBSERVER is based on CLASSIC ([7]). In those two systems, the problem of rewriting queries using views is handled as a planning problem and the completeness of the rewriting algorithms is not addressed. The *MOMIS* ([5]) system is based on the use of a very expressive description logic (ODL-I3) for describing the schemas of the sources to integrate. It follows a *GAV* approach since the mediated schema is inferred from the schemas of the sources.

Orthogonally to the *LAV* versus *GAV* and relational versus object-oriented distinctions, another criteria for distinguishing the information integration agents is the expressive power chosen for modeling the mediated schema. In this chapter, we present two recent information integration agents, namely PICSEL and Xyleme, which are illustrative of two radically different choices concerning the expressivity of the mediated schema.

In PICSEL[18], it has been chosen to offer a formalism combining the expressive power of rules and classes for designing a rich mediated schema, thus enabling a fine-grained description of the contents of the sources. Our aim was to build information integration agents related to application domains for which there exists many information sources containing closely related data. In such a setting, it is of primary importance to model the fine-grained differences between contents of sources if we want to be able to answer precise queries in an efficient way. PICSEL has been used for building a mediator in the tourism domain in collaboration with France Telecom R&D and the web travel agent Degriftour [2]. This travel agent makes available on-line three databases that contain different types of touristic products (flights, tours, stays in different places), each of them having its own specificities. For example, the *BonjourFrance* database offers a large variety of touristic products but all of them are located in France. The so-called *Degriftour* database offers flights, stays and tours for a lot of destinations all over the world. Its specificity however is that all its offers correspond to a departure date which is within the next two weeks. As a counterpart, the corresponding prices are specially interesting. The *Reductour* database provides rather similar products but with a less strong constraint on the departure date: it just has to be within the next eight months. Other differences exist between the contents of those three databases. For instance, we might know that *BonjourFrance* can provide rooms in Bed&Breakfast in addition to hotel rooms, while the only housing places that are proposed in the two other sources are hotels, located exclusively out of France for *Degriftour*, and located exclusively in Europe for *Reductour*.

In Xyleme [32,33,1], the choice of a simple tree structure for mediated schemas has been guided by the goal of providing a very wide-area integration of XML sources that could scale up to the Web. The system architecture and the design choices have been motivated by "web search engine"-like performance requirements, i.e. supporting many simultaneous queries over a Web-scale XML repository. Xyleme is based on a simple data model with data trees and tree types, and a simple query language based on tree queries with boolean condi-

[2] see http://www.degriftour.fr/

tions. The main components of the data model are a mediated schema modeled by an abstract tree type, as a view of a set of tree types associated with actual data trees, called concrete tree types, and a mapping expressing the connection between the mediated schema and the concrete tree types. The simplicity of the mapping relation (correspondences between tree paths) eases automatic mapping generation and distributed storage. The query language is intended to enable end-users to express simple Query-by-Example tree queries over the mediated schema. Xyleme optimizes such tree queries to get very efficient response time.

This chapter is organized as follows. In section 2, we describe the way the expressive power of the CARIN [21] language is exploited in the PICSEL information integration agent, while maintaining the decidability of query rewriting using views and the tractability of query answering. In section 3, after describing the tree-based data model of Xyleme, we present a method for semi-automatically generating the mapping relation. For setting up semantic integration in Xyleme, the main difficulty comes from the very large number of data sources handled by Xyleme. As a consequence, the number of concrete data trees that have to be mapped to the abstract tree type makes impossible that the mapping relation is manually defined by some database administrator. We have therefore studied how to generate the mapping relation as automatically as possible, by combining syntactic, semantic and structural criteria. A prototype has been implemented and evaluated in the cultural domain. Finally, in Section 4, we draw some conclusions and we outline directions of future work about information integration in the setting of the Semantic Web.

2 PICSEL

In PICSEL, CARIN is used to represent both the domain of application and the contents of information sources that are available and relevant to that domain.

2.1 Syntax and Semantics of CARIN

A CARIN knowledge base (KB) contains two components: a set $\mathcal{R}$ of rules, and a set $\mathcal{T}$ of Description Logics statements. Description logics statements are definition or inclusion statements about concepts and roles in the domain. Concepts and roles are unary and binary relations that can also appear in the antecedents of the rules. Relations that do not appear in the description logics statements are called *ordinary relations*. Ordinary relations can be of any arity.

Description Logics component in CARIN**:** The DL component of a CARIN knowledge base in PICSEL contains *concept definitions* and some kinds of *concept inclusions*, using the $\mathcal{ALN}$ DL. $\mathcal{ALN}$ contains the DL constructors of conjunction ($C_1 \sqcap C_2$), value restriction ($\forall R.C$), number restrictions ($(\geq n\, R), (\leq n\, R)$), and negation ($\neg A$) restricted to basic concepts only.

– Concept definitions are of the form $CN := ConceptExpression$, where CN is a concept name and $ConceptExpression$ is a concept expression. We assume that a concept name appears in the left hand side of at most one concept definition. *Atomic concepts* are those which do not appear in any left hand side of a definition. A concept name CN depends on a concept name CN' if CN' appears in the definition of CN. A set of concept definitions is said *acyclic* if there is no cycle in the concept names dependency relation. In the setting of PICSEL, we consider only acyclic concept definitions. We can *unfold* an acyclic set of definitions by iteratively substituting every concept name with its definition.
– The concept inclusions that are allowed in the PICSEL setting are of the form:
  - $A \sqsubseteq ConceptExpression$, where A is a atomic concept,
  - or $A_1 \sqcap A_2 \sqsubseteq \bot$, where A_1 and A_2 are atomic concepts.

Rule component in CARIN**:** The rule component $\mathcal{R}$ of a CARIN knowledge base contains a set of rules that are logical sentences of the form:

$$\forall \bar{X}[p_1(\bar{X}_1) \wedge \ldots \wedge p_n(\bar{X}_n) \Rightarrow q(\bar{Y})]$$

where $\bar{X}_1, \ldots, \bar{X}_n, \bar{Y}$ are tuples of variables (included in $\bar{X}$) or constants. We require that the rules are safe, i.e., a variable that appears in $\bar{Y}$ must also appear in $\bar{X}_1 \cup \ldots \cup \bar{X}_n$. As a shortcut, in the following, the variable quantification will be omitted. The relations $p_1, \ldots, p_n$ may be either concept names or expressions, role names, or ordinary relations that do not appear in $\mathcal{T}$. The relation q must be an ordinary relation.

The *base* relations are those which do not appear in any consequent of rules. In particular, any concept or role appearing in the rules are base relations. We call a base atom an atom $p(\bar{X})$ where p is a base relation. We call *concept-atom* an atom $p(X)$ where p is a concept name or expression, and *role-atom* an atom $r(X, Y)$ where r is a role name.

An ordinary relation p is said to *depend* on an ordinary relation q if q appears in the antecedent of a Horn rule whose consequent is p. A set of rules is said to be *recursive* if there is a cycle in the dependency relation among ordinary relations. In the setting of PICSEL, we consider non recursive rules.

Since the rules are safe, without loss of generality, we can assume that in every rule, we have the disequality $X \neq Y$ for every pair of distinct variables appearing in the rule. For clarity, we omit these atoms in our examples and algorithms.

In addition, we allow *constraints* of the form:

$$p_1(\bar{X}_1) \wedge \ldots \wedge p_n(\bar{X}_n) \Rightarrow \bot.$$

Semantics of CARIN **KBs:** The semantics of CARIN KBs is the standard model-based semantics of first-order logic. An interpretation I contains a non-empty domain $\mathcal{O}^I$. It assigns to every constant a an object $\alpha^I(a) \in \mathcal{O}^I$, and a relation of arity n over the domain $\mathcal{O}^I$ to every relation of arity n. In particular, it assigns a unary relation C^I to every concept in $\mathcal{T}$, and a binary relation R^I over $\mathcal{O}^I \times \mathcal{O}^I$ to every role R in $\mathcal{T}$. The extensions of concept and role descriptions are given by the following equations: ($\sharp\{S\}$ denotes the cardinality of a set S):

$(C \sqcap D)^I = C^I \cap D^I$,
$(\neg A)^I = \mathcal{O}^I \setminus A^I$,
$(\forall R.C)^I = \{d \in \mathcal{O}^I \mid \forall e : (d,e) \in R^I \rightarrow e \in C^I\}$
$(\geq n\, R)^I = \{d \in \mathcal{O}^I \mid \sharp\{e \mid (d,e) \in R^I\} \geq n\}$
$(\leq n\, R)^I = \{d \in \mathcal{O}^I \mid \sharp\{e \mid (d,e) \in R^I\} \leq n\}$

An interpretation I is a model of a knowledge base $(\mathcal{T}, \mathcal{R})$ if it is a model of each of its components $\mathcal{T}$ and $\mathcal{R}$. An interpretation I is a model of $\mathcal{T}$ if $A^I \subseteq D^I$ for every inclusion $A \sqsubseteq D$ in $\mathcal{T}$, $CN^I = D^I$ for every concept definition $CN := D$. If CE and DE are two concept expressions, we say that CE *is subsumed by* DE (modulo $\mathcal{T}$), denoted $CE \preceq DE$ ($CE \preceq_{\mathcal{T}} DE$). if $CE^I \subseteq DE^I$ in every interpretation I (model of $\mathcal{T}$). A concept expression CE is satisfiable (modulo $\mathcal{T}$) iff there exists an interpretation I (model of $\mathcal{T}$) such that CE^I is not empty.

An interpretation I is a model of a rule $r : p_1(\bar{X}_1) \wedge \ldots \wedge p_n(\bar{X}_n) \Rightarrow q(\bar{Y})$ if, whenever α is a mapping from the variables [3] of r to the domain $\mathcal{O}^I$, such that $\alpha(\bar{X}_i) \in p_i^I$ for every atom of the antecedent of r, then $\alpha(\bar{Y}) \in q^I$.

An interpretation I is a model of a constraint $c : p_1(\bar{X}_1) \wedge \ldots \wedge p_n(\bar{X}_n) \Rightarrow \perp$ if there does not exist a mapping α from the variables of c to the domain $\mathcal{O}^I$ such that $\alpha(\bar{X}_i) \in p_i^I$ for every atom of the antecedent of c.

2.2 Modelling the Mediated Schema

Given a basic vocabulary denoting names of base relations that are meaningful for the application domain (e.g., tourism), Carin is used for defining new relations that are significant for the tourism domain and that can be defined over the base relations. In the setting of Carin, there are two ways of defining complex relations: by rules, or by concept expressions. It is illustrated by the following example.

Example 1. Let us suppose that our basic vocabulary contains atomic concepts such as $HousingPlace$, $Breakfast$, $CollectiveBuilding$, $PrivateBuilding$, which respectively denote the set of housing places and varied sets of services or buildings, and roles such as $AssRoom$, $AssMeal$ and $AssBuilding$, which denote binary relations between housing places and their associated buildings, room and meal services. We can define the two new concepts $Hotel$ and $Bed\&Breakfast$ by the following DL definitions:

$$Hotel := HousingPlace \sqcap (\geq\ 5\, AssRoom) \sqcap (\forall AssBuilding.CollectiveBuilding)$$

$$Bed\&Breakfast := HousingPlace \sqcap (\geq\ 1\, AssRoom) \sqcap (\geq\ 1\, AssMeal) \sqcap (\forall AssMeal.Breakfast) \sqcap (\forall AssBuilding.PrivateBuilding)$$

[3] Distinct variables are mapped to distinct elements

Rules can be used to define new n-ary relations. For instance, the following rule defines the notion of flights combined with stays as a 4-ary relation *FlightStay*. A stay combined with a flight is characterized by a departure city (denoted by the first variable *Dcity* in the consequent of the rule), an arrival city (denoted by the variable *Acity*), a departure date (*Ddate*), a return date (*Rdate*). The possible combinations of a flight to and back a given destination with a stay at that place obey some constraints that are expressed by the conditions in the antecedent of the rule.

$$\begin{aligned}&Stay(S) \wedge Flight(V) \wedge Assoc(S,H) \wedge Located(H, ACity)\\ &\wedge ArrivalCity(V, Acity) \wedge DepartureCity(V, Dcity) \wedge DepartureDate(V, Ddate)\\ &\wedge ReturnDate(V, Rdate) \wedge BeginningDate(S, Ddate) \wedge EndDate(S, Rdate)\\ &\quad \Rightarrow FlightStay(Dcity, Acity, Ddate, Rdate)\end{aligned}$$

2.3 Modelling the Contents of the Information Sources

Our approach for describing the information sources has been guided by the necessity of trading off expressive power against decidadibility of query answering. More precisely, each information source $\mathcal{S}$ is characterized by a set of source relations $\mathcal{V}_S$ (also called views), and described by a CARIN knowledge base which contains:

1. a set $\mathcal{I}_S$ of rules $v(\bar{X}) \Rightarrow p(\bar{X})$ that indicates which kind of data can be found in the source $\mathcal{S}$. The p's are domain relations and there are as many source relations v's in $\mathcal{V}_S$ (and as many implications in $\mathcal{I}_S$) as domain relations whose instances can be found in the source $\mathcal{S}$,
2. a set $\mathcal{C}_S$ of constraints on the instances of the source relations. We allow two types of constraints. *Terminological constraints* are of the form $v \sqsubseteq C$, where C is a concept expression. *Integrity constraints* are of the form $l_1(\bar{X}_1) \wedge \ldots \wedge l_n(\bar{X}_n) \Rightarrow \perp$ where the l_i's are source relations or negation of source relations, such that there is atmost one negation in each constraint.

A *source atom* (also called *view atom*) is an atom of the form $v(\bar{X})$ where v is a source relation.

We encapsulate the terminological constraints within the *definition of views* associated with such constraints.

Definition 1. *View definition Let v be a view in $\mathcal{V}$ appearing in the description $\mathcal{I}_S \cup \mathcal{C}_S$ of a source $\mathcal{S}$, and such that $v(X) \Rightarrow p(X) \in \mathcal{I}_S$. The* definition of the view *v, denoted $def(v)$, is:*

- *$def(v) = p \sqcap C$ if $v \sqsubseteq C \in \mathcal{C}_S$,*
- *$def(v) = p$ if v is not associated to any terminological constraint.*

As an example, we can have the following (partial, for this illustration) descriptions of the three information sources that we have previously mentioned (i.e., *BonjourFrance*, *Degriftour* and *Reductour*).

Example 2. The rules in the description of the three sources say that we can find instances of housing places and flights in all of them (together with instances of associated properties like their location, their departure cities and dates ...). The constraints contained in each description enable distinguishing between them: for instance, the housing places and flights that can be found in *BonjourFrance* are restricted to be located in France; the housing places that can be found in *Reductour* are necessarily hotels. Some constraints serve to express that instances of some binary relations that can be found in a given source are exclusively related to some unary relations that can be found in the same source: for instance, the constraint $v^2_{BF}(X,Y) \wedge \neg v^1_{BF}(X) \Rightarrow \perp$ in $Descr(BonjourFrance)$ expresses that the locations that can be found in the source $BonjourFrance$ (denoted by the source relation v^2_{BF} and the implication $v^2_{BF}(X,Y) \Rightarrow Located(X,Y)$) are only locations that are related to the housing places that can be found in the same source (denoted by the source relation v^1_{BF} and the implication $v^1_{BF}(X) \Rightarrow HousingPlace(X)$).

$Descr(BonjourFrance) : v^1_{BF}, \ldots, v^n_{BF}$	
mappings: $\mathcal{I}_{BonjourFrance}$	constraints: $\mathcal{C}_{BonjourFrance}$
$v^1_{BF}(X) \Rightarrow HousingPlace(X)$	$v^1_{BF} \sqsubseteq (\forall Located.France)$
$v^2_{BF}(X,Y) \Rightarrow Located(X,Y)$	$v^2_{BF}(X,Y) \wedge \neg v^1_{BF}(X) \Rightarrow \perp$
$v^3_{BF}(X) \Rightarrow Flight(X)$	$v^3_{BF} \sqsubseteq (\forall ArrivalCity.France)$
$v^4_{BF}(X,Y) \Rightarrow ArrivalCity(X,Y)$	$v^4_{BF}(X,Y) \wedge \neg v^3_{BF}(X) \Rightarrow \perp$
...	...

$Descr(Degriftour) : v^1_D, \ldots, v^n_D$	
mappings: $\mathcal{I}_D$	constraints: $\mathcal{C}_D$
$v^1_D(X) \Rightarrow Hotel(X)$	$v^1_D \sqsubseteq (\forall Located.\neg France)$
$v^2_D(X,Y) \Rightarrow Located(X,Y)$	
$v^3_D(X) \Rightarrow Flight(X)$	$v^3_D \sqsubseteq (\forall DepartureDate.NextTwoWeeks)$
$v^4_D(X,Y) \Rightarrow ArrivalCity(X,Y)$	$v^4_D(X,Y) \wedge \neg v^3_D(X) \Rightarrow \perp$
...	...

$Descr(Reductour) : v^1_R, \ldots, v^n_R$	
mappings: $\mathcal{I}_R$	constraints: $\mathcal{C}_R$
$v^1_R(X) \Rightarrow Hotel(X)$	$v^1_R \sqsubseteq (\forall Located.Europe)$
$v^2_R(X,Y) \Rightarrow Located(X,Y)$	
$v^3_R(X) \Rightarrow Flight(X)$	$v^3_R \sqsubseteq (\forall DepartureDate.NextEightMonths)$
$v^4_R(X,Y) \Rightarrow ArrivalCity(X,Y)$	$v^4_R(X,Y) \wedge \neg v^3_R(X) \Rightarrow \perp$
...	...

Here are some view definitions that we get from this modelling:
$def(v^1_{BF}) = HousingPlace \sqcap \forall Located.France$
$def(v^1_D) = Hotel \sqcap \forall Located.\neg France$
$def(v^1_R) = Hotel \sqcap \forall Located.Europe.$

2.4 Query Processing in Picsel

The queries that we consider are conjunctive queries of the form $Q(\bar{X}) : p_1(\bar{X}_1, \bar{Y}_1) \wedge \ldots \wedge p_m(\bar{X}_m, \bar{Y}_m)$, where the p_j's are domain relations, some of which may be concept expressions or role names. The variables of $\bar{X} = \bar{X}_1 \cup \ldots \cup \bar{X}_m$ are called the *distinguished* variables of the query: they represent the variables of the query which the user is interested in knowing the instances when he asks the query. The variables that are not distinguished (denoted by $\bar{Y} = \bar{Y}_1 \cup \ldots \cup \bar{Y}_m$) are called *existential* variables of the query: they are existentially quantified and their use is to constrain the distinguished variables. For example, by the query $Q(X) : Hotel(X) \wedge Located(X, Y) \wedge France(Y)$, the user specifies that the answers he is interested in, are all the possible instances of X that are hotels, and for which there exists instances of Y (their locations) that are in France.

Answering queries in PICSEL resorts to find conjunctions of source atoms (called *rewritings*) which, together with the mediated schema and the descriptions of the sources, entail the initial query.

Definition 2. *Rewriting Let $Q(\bar{X}) : p_1(\bar{X}_1, \bar{Y}_1) \wedge \ldots \wedge p_m(\bar{X}_m, \bar{Y}_m)$ be a query expressed in term of domain relations, let $Q_{\mathcal{V}}(\bar{X}, \bar{Z})$ be a conjunction of view atoms. $Q_{\mathcal{V}}(\bar{X}, \bar{Z})$ is a rewriting of $Q(\bar{X})$ iff :*
- *$\{Q_{\mathcal{V}}(\bar{X}, \bar{Z})\} \cup \mathcal{R} \cup \mathcal{T} \cup \mathcal{I}_{\mathcal{V}} \cup \mathcal{C}_{\mathcal{V}}$ is satisfiable, and*
- *$\{Q_{\mathcal{V}}(\bar{X}, \bar{Z})\} \cup \mathcal{R} \cup \mathcal{T} \cup \mathcal{I}_{\mathcal{V}} \cup \mathcal{C}_{\mathcal{V}} \models \exists \bar{Y}\, p_1(\bar{a}_1, \bar{Y}_1) \wedge \ldots \wedge p_m(\bar{a}_m, \bar{Y}_m)$*

Example 3. Coming back to Example 2, let us consider again the query:
$Q(X) : Hotel(X) \wedge Located(X, Y) \wedge France(Y)$.

The query $Q_{\mathcal{V}}(X) : v_R^1(X) \wedge v_{BF}^1(X) \wedge v_R^2(X, Y)$ is a rewriting of $Q(X)$ because, according to the definitions of v_R^1 and of v_{BF}^1, $v_R^1(X)$ entails $(Hotel \sqcap \forall Located.Europe)(X)$ and $v_{BF}^1(X)$ entails $(HousingPlace \sqcap \forall Located.France)$ (X). Then, $v_R^1(X) \wedge v_{BF}^1(X)$ entails in particular $Hotel(X) \wedge (\forall Located.France)$ (X). Therefore, according to the semantics of the $\forall$ constructor, $v_R^1(X) \wedge v_{BF}^1(X)$ $\wedge v_R^2(X, Y)$ entails that there exists Y such that $Hotel(X) \wedge Located(X, Y) \wedge France(Y)$ is true, and thus $Q_{\mathcal{V}}$ is a rewriting of Q.

Note that this rewriting shows that answering the query Q using the available sources requires to integrate the information of the two sources *Reductour* and *BonjourFrance*: hotels and their location are found in the *Reductour* source, but we have to intersect those data with housing places stored in the *BonjourFrance* source to be sure to obtain hotels that are located in France.

On the other hand, $Q'_{\mathcal{V}}(X) : v_D^1(X) \wedge v_{BF}^1(X) \wedge v_D^2(X, Y)$ is not a rewriting because it is insatisfiable. It is due to the fact that from the definitions of v_D^1 (i.e., $Hotel \sqcap \forall Located.\neg France$) and v_{BF}^1 (i.e., $HousingPlace \sqcap \forall Located.France$), it can be entailed that the location of X is on the one hand necessarily out of France, and on the other hand in France. Therefore, $v_D^1(X) \wedge v_{BF}^1(X) \wedge v_D^2(X, Y)$ entails the two contradictory facts $France(Y)$ and $\neg France(Y)$, and thus $Q'_{\mathcal{V}}$ is insatisfiable.

A rewriting of a query $Q(\bar{X})$ can then be viewed as a specialized query that can be directly executed on the sources, and which provides relevant answers to the initial query.

Query processing in PICSEL can be summarized as follows.

1. **Normalization of the query.** We exhaustively apply to the body of the query the normalization rule: $C(o) \wedge D(o) \rightarrow (C \sqcap D)(o)$, where $C(o)$ and

$D(o)$ are concept-atoms. Then, the resulting concept expressions are themselves normalized. Every concept expression is put in a normal form of a conjunction of concept expressions of the (*simple*) form: A (atomic concept), $\neg A$, $(\geq n\,R)$, $(\leq n\,R)$, or of the (*complex*) form: $\forall R_1 \forall R_2 \ldots \forall R_k.D$, where D is simple.

2. **Verification of the satisfiability of the query.** This consists of (a) checking the satisfiability modulo the terminology of each concept expression appearing in the query, and (b) applying a forward-chaining algorithm to the rules until $\perp$ is obtained or no new atom can be obtained. If the query is insatisfiable, the query processing stops and the query is transmitted to a module of query relaxation described in [6].
3. **Query unfolding.** This is an iterative process based on successive steps of unfolding using the rules. Let $p(\bar{X})$ be an ordinary atom. Let $r : p_1(\bar{X}'_1, \bar{Y}'_1) \wedge \ldots \wedge p_k(\bar{X}'_k, \bar{Y}'_k) \Rightarrow p(\bar{X}')$ be a rule of $\mathcal{R}$. Let α be the most general unifier of $p(\bar{X})$ and $p(\bar{X}')$, extended such that every variable Y'_i is assigned to a fresh variable that appears nowhere else. An unfolding of the query using the rule r is a conjunctive query obtained by replacing in the query the atom $p(\bar{X})$ by the conjunction: $p_1(\alpha(\bar{X}'_1), \alpha(\bar{Y}'_1)) \wedge \ldots \wedge p_k(\alpha(\bar{X}'_k), \alpha(\bar{Y}'_k))$.
The process is iterated on each satisfiable unfolding until getting conjunctive queries in which all the ordinary atoms are base predicates. Since the rules are non recursive, the process terminates.
4. **Computation of terminological approximations.** Depending on the variable binding within the body of each conjunctive query, it is possible that conjunctions of concept-atoms and role-atoms are entailed by a single concept-atom. It is important to determine such situations when they exist because they may lead to rewritings that could not be found otherwise.
In [17], the conjunctions of concept-atoms and atom-roles that can be entailed by a single concept-atom are characterized by the structure of the binding graph of their variables. The binding graph of a conjunction of unary and binary atoms is defined as follows: its nodes are the variables appearing in the conjunction ; there is an edge from U_1 to U_2 iff there exists a binary atom $r(U_1, U_2)$ in the conjunction. In particular, it is shown that if a conjunction $cj(X, \bar{Y})$ has a binding graph which is a tree rooted in X, we can build its $\mathcal{ALN}$ *concept approximation* which is the most general $\mathcal{ALN}$ concept description C such that $C(X) \models \exists \bar{Y}\, cj(X, \bar{Y})$.

Example 4. Consider again the query $Q(X) : Hotel(X) \wedge Located(X,Y) \wedge France(Y)$. Its body has a binding graph which is a tree rooted in X. Its concept approximation is: $Hotel \sqcap (\geq 1\, Located) \sqcap (\forall Located.France)$. The query Q can then itself be approximated by the atomic query Q' obtained by replacing the body of Q by:
$(Hotel \sqcap (\geq 1\, Located) \sqcap (\forall Located.France))(X)$.
$Q'(X) : (Hotel \sqcap (\geq 1\, Located) \sqcap (\forall Located.France))(X)$. is called a terminological approximation of Q.

Fo every satisfiable conjunctive query resulting from rule unfolding, we build their *terminological approximations*. They are obtained by replacing a con-

junction $cj(X, \bar{Y})$ of concept-atoms and role-atoms whose binding graph is a tree rooted in X by the single-concept atom $C(X)$ where C is the concept approximation of $cj(X, \bar{Y})$.
For every resulting conjunctive query, we then proceed to the rewriting of each atom of its body separately, and we conjunct the results to get the final rewritings.

5. **Rewriting of each query, atom by atom.**
Rewriting an ordinary atom or of a role-atom is trivial: the rewritings of a role-atom $\mathsf{r}(U_1, U_2)$ (or of an ordinary atom $\mathsf{p}(\bar{U})$) are the view-atoms $v(U_1, U_2)$ (ou $v(\bar{U})$) such that $v \in \mathcal{V}$ and $def(v) = \mathsf{r}$ (or $def(v) = \mathsf{p}$).
Rewriting concept-atom is much more complex. It is done by iterating the application of rewriting rules according to a strategy which guarantees termination of the rewriting process. Those rewriting rules are founded the following rules of entailment.
   - Rewriting of a concept-atom of the form $(\sqcap_{i=1}^{n} \mathsf{C}_i)(U_0)$: the rewritings of this type of atom are obtained by conjuncting the rewritings of the concept-atoms $\mathsf{C}_i(U_0)$.

$$(1)\ \bigwedge_{i=1}^{n} \mathsf{C}_i(U_0) \models (\sqcap_{i=1}^{n} \mathsf{C}_i)(U_0).$$

   - Rewriting of a concept-atom of the form $\mathsf{C}(U_0)$ (where C is simple i.e., not a conjunction) is based on the following entailment rule where $\mathsf{D} \preceq_{\mathcal{T}} \mathsf{C}$:

$$(2)\ (\forall \mathsf{r}_1\ \cdots (\forall \mathsf{r}_n\ \mathsf{D}))(U_n) \wedge \bigwedge_{i=1}^{n} \mathsf{r}_i(U_{n+1-i}, U_{n-i}) \models \mathsf{C}(U_0),$$

   This is a termination case for the rewriting algorithm.
   - Rewriting of a concept-atom of the form $(\geq n\ \mathsf{r})(U_0)$ is based on the following entailment rule:

$$(3)\ \bigwedge_{i=1}^{n} \mathsf{r}(U_0, U_i) \wedge \bigwedge_{i=1}^{n} \bigwedge_{\substack{j = i+1 \\ i \neq j}}^{n} U_i \neq U_j \models (\geq n\ \mathsf{r})(U_0).$$

   The rewriting process is iterated on the role-atoms of the left hand side of the rule.
   - Rewriting of a concept-atom of the form $(\forall \mathsf{r}\ \mathsf{C})(U_0)$ is based on the following entailment rule:

$$(4)\ (\leq n\ \mathsf{r})(U_0) \wedge \bigwedge_{i=1}^{n} (\mathsf{r}(U_0, U_i) \wedge \mathsf{C}(U_i)) \wedge \bigwedge_{i=1}^{n} \bigwedge_{\substack{j = i+1 \\ i \neq j}}^{n} U_i \neq U_j \models (\forall \mathsf{r}\ \mathsf{C})(U_0).$$

   The rewriting process is iterated on the role-atoms and on the concept-atoms of the left hand side of the rule.

Example 5. Consider the atomic query:
$Q'(X) : (Hotel \sqcap (\geq 1\,Located) \sqcap (\forall Located.France))(X)$
which is the terminological approximation of the conjunctive query:
$Q(X) : Hotel(X) \wedge Located(X, Y) \wedge France(Y)$.
Consider the views described in Example 2. Based on Rule (1), rewriting $(Hotel \sqcap (\geq 1\,Located) \sqcap (\forall Located.France))(X)$ results in the rewritings of $Hotel(X)$, $(\geq 1\,Located)(X)$ and $(\forall Located.France)(X)$.
Based on Rule (2), the rewriting of $Hotel(X)$ is terminal and we get:
$Rewritings(Hotel(X)) = \{v_D^1(X), v_R^1(X)\}$.
Based on Rule (3), we obtain the following rewritings of $(\geq 1\,Located)(X)$:
$Rewritings((\geq 1\,Located)(X)) = \{v_{BF}^2(X, Y), v_R^2(X, Y), v_D^2(X, Y)\}$.
Finally, Rule (2) is used for getting the rewriting of $(\forall Located.France)(X)$:
$Rewritings((\forall Located.France)(X)) = \{v_{BF}^1(X)\}$.
There are 6 ways of conjunctively combining the rewritings of the three atoms $Hotel(X)$, $(\geq 1\,Located)(X)$ and $(\forall Located.France)(X)$:
- $v_D^1(X) \wedge v_{BF}^2(X, Y) \wedge v_{BF}^1(X)$: not satisfiable.
- $v_D^1(X) \wedge v_R^2(X, Y) \wedge v_{BF}^1(X)$: not satisfiable.
- $v_D^1(X) \wedge v_D^2(X, Y) \wedge v_{BF}^1(X)$: not satisfiable.
- $v_R^1(X) \wedge v_{BF}^2(X, Y) \wedge v_{BF}^1(X)$: satisfiable.
- $v_R^1(X) \wedge v_R^2(X, Y) \wedge v_{BF}^1(X)$: not satisfiable.
- $v_R^1(X) \wedge v_D^2(X, Y) \wedge v_{BF}^1(X)$: satisfiable.
The rewritings of $Q'(X)$, and thus of $Q(X)$, are those which are satisfiable:
$Q_{\mathcal{V}}^1(X) : v_R^1(X) \wedge v_{BF}^2(X, Y) \wedge v_{BF}^1(X)$
$Q_{\mathcal{V}}^2(X) : v_R^1(X) \wedge v_D^2(X, Y) \wedge v_{BF}^1(X)$.

3 Xyleme

The Xyleme project deals with data integration when data sources are XML documents. All XML documents are stored in a repository. In this way, the system is efficient even when several data issued from distributed sources must be combined to answer queries. A semantic module plays the role of interface between end-users and XML documents which, by definition, come from several sources and are semantically heterogeneous. Figure 2 presents the overall architecture of the system.

– The Repository and Index Manager module is the lowest level in Xyleme. It enables to store and index XML documents.
– The Acquisition and Crawler module inspects the Web and collects all the XML documents, which are loaded in the repository by the Loader module.
– The Change Control module is responsible of specific functionalities, such as monitoring of document changes, version management and subscription of temporal queries.
– The Semantic module provides a homogeneous integrated and mediated schema on the heterogeneous XML documents stored in the repository.

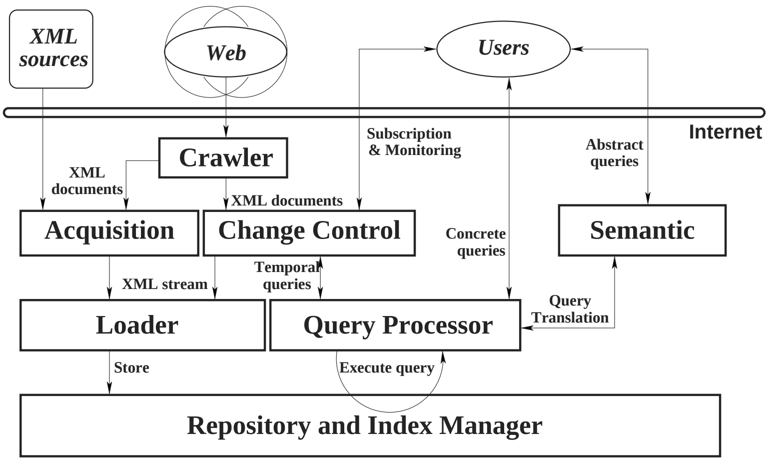

Fig. 2. Xyleme architecture

– The Query Processor module enables to query the XML repository as a database. In particular, it translates a query in terms of the semantic layer into another one computable on the stored documents.

A general presentation of Xyleme is given in [33]. In this chapter we focus on the semantic module and the translation of abstract queries. A presentation of the other modules can be found in [20,22,24].

3.1 A Uniform Tree-Based Data Model

The XML documents that are stored in the repository are instances of a DTD that defines their structure. From a data integration viewpoint, the data coming from differents sources are the XML documents, and the DTDs are the schemas of the data sources. For example, here is the partial definition of the DTD `MyDTD`:

```
<!ELEMENT MyDTD (Film*, Painting*)>
<!ELEMENT Film (Casting, Character*,Title)>
<!ELEMENT Casting (Actor*)>
<!ELEMENT Painting (Title, Author, Museum?)>
<!ELEMENT Actor (#PCDATA)>
............
```

In our model, we abstract XML documents as being *data trees*. Our modelling simplifies real XML documents in several ways. For example, the model does not distinguish between attributes and elements. We consider unordered trees. We use also a simplified version of DTDs, which we call *tree type*, where the multiplicity of an element is not considered.

Data trees and tree types are labelled unordered trees, where the labels correspond to the names of the elements and the attributes of the XML documents.

A data tree d is an instance of a tree type t if there exists an homomorphism from d onto t. For example, Figure 3 gives an example of a data tree. It is an instance of the tree type given in Figure 4.

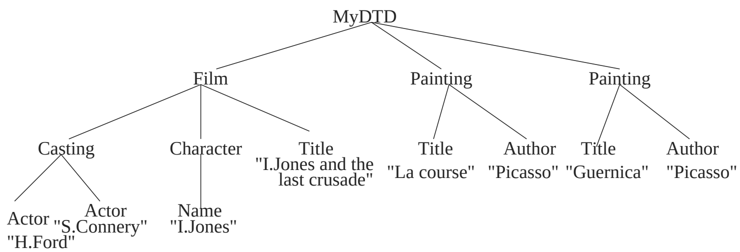

Fig. 3. A data tree

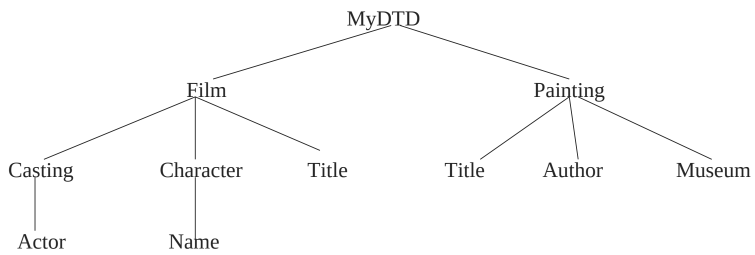

Fig. 4. Example of a concrete tree type

The aim of the semantic module is twofold: (i) to provide an homogeneous mediated schema over the data trees, in order to allow a natural expression of queries for the user without having to be aware of actual tree types that can have heterogeneous structures and labels; (ii) to define the connection between this semantic module and the tree types describing the actual data trees, in order to make possible query processing.

The semantic module provides a simple description of domains (e.g. culture, tourism) in the form of an *abstract tree type*. An abstract tree type can be considered as an abstract merger of the DTDs associated with the set of XML documents of some domain. Figure 5 shows a fragment of an abstract tree type in the cultural domain. The label of the root is the domain name (`Culture`). The internal nodes are labelled with terms that represent either concepts of the domain (`Art, Architecture, Cinema`) or properties of those concepts (`Name, Title`).

There are two main reasons for the choice of representing a mediated schema as a tree. First, it must be compatible with a real XML DTD, because it is

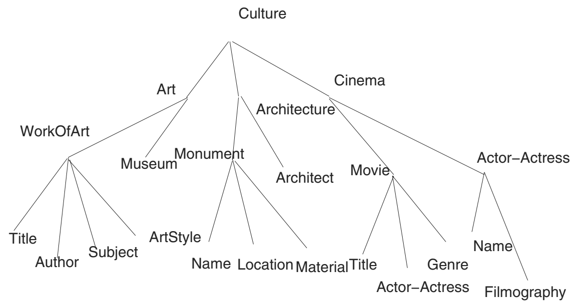

Fig. 5. Fragment of an abstract tree type on culture

intended to be the schema of all the XML documents of the domain. Next, the mediated schema must be simple enough, because it is the support of the visual query interface tool, intended to be used by non-programmer users. The links between a node and its sons in the abstract tree type have no strict semantics. They may correspond to a relation of specialisation between concepts, or to a relation of composition, or simply express different viewpoints on a concept. A link also exists between a concept and a property of it. We do not distinguish between those different types of links. In other words, a node labelled with a term E_2 being a son of a node labelled with E_1 just means that the term E_2 has to be interpreted in the scope of meaning of E_1. Thus, the different occurrences of a same term do not have the same meaning. This is obvious for terms that represent properties (e.g. `Name` can appear different times, and may represent the name of different entities), but it is also the case for terms representing concepts (e.g. `Author` under `Movie` means director, while `Author` under `Painting` means painter).

Such representations are very simple and structured. They enable end-users to build semantic queries from different viewpoints in an easy and natural way, using both the vocabulary and the structure of the abstract tree type. This provides a homogeneous and simple user interface for querying a very large amount of heterogeneous XML documents stored in the repository. Using this interface, end-users must be able to obtain relevant answers for queries.

For this, Xyleme has to identify the XML documents concerning each query issued against the semantic module. This requires the creation and maintenance of a correspondence between the labels of the abstract tree type and the elements of the concrete tree type that describe the structure of the XML documents. This way, each abstract query can be translated into a set of concrete queries over real documents, by simply replacing the abstract labels of the query with the corresponding concrete labels. In Xyleme, the correspondence is stated by a *mapping relation* between paths of the abstract tree type and paths of the tree types modeling the DTDs of the data stored in the repository.

Example 6. Here are some examples of meaningful mapping elements between the abstract tree type given in Figure 5 and the tree type given in Figure 4:

```
Culture/Cinema/Movie <-> MyDTD/Film
Culture/Cinema/Movie/ActorActress <-> MyDTD/Film/Casting/Actor
.......................
```

Because of the number of concrete data trees that have to be mapped to the abstract tree type, the mapping relation cannot be manually defined by some database administrator. Some automatic help must be provided. In Section 3.3, we describe a semi-automatic method for generating the mapping relation that has been implemented in Xyleme and experimented. By accessing the abstract tree type through a user-friendly interface, end-users query a single tree structure summarizing many DTDs. The query language is also tree-based.

The query tree on the left in Figure 6 models a query in the cultural domain asking the titles of all the films in which the actor Sean Connery is acting. The query tree on the right in Figure 6 asks the description of all impressionist works of art. These examples illustrate the basic method for building abstract queries: the user marks in the abstract tree type the nodes to be included in the result (selected nodes are marked with a `S`, e.g. `Title, WorkOfArt`) and the nodes constrained with conditions (conditional nodes are marked with a `C`, e.g. `Actor-Actress, ArtStyle`). The query itself is given by a tree query pattern built from the abstract tree type.

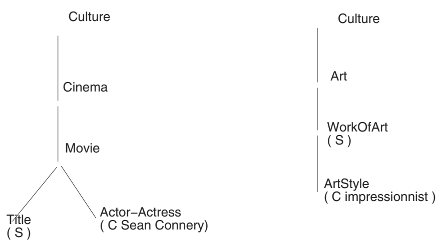

Fig. 6. Example of user queries

The Xyleme query language is a subset of *XQuery*, the W3C query language for XML data [31]. Unlike XQuery, the Xyleme query language does not allow joins or document transformation. Moreover, the language ignores ordering and random access through an ordinal number to the descendants of a tree node, features that exist in *XPath* [10], which is a part of XQuery. On the other hand, the Xyleme query language is more powerful than XPath, because it enables to extract any part of a subtree, while XPath can only extract a full subtree, identified by its access path. Notice that even if the abstract tree queries do not contain joins, the translation into concrete tree queries may produce joins based on links between concrete documents (see [11]). Notice also that the last version of the Xyleme query language, not presented here, was enriched with additional features such as joins (on values and on links) and simple document transformation.

3.2 Query Processing

The users queries are defined relatively to the mediated schema and not relatively to the actual tree types corresponding to the data stored in the repository. Therefore they cannot be directly evaluated against that data. The evaluation of an abstract query is then a two step process. First, the abstract tree query is translated into concrete tree queries by using the mapping relation. Second, each concrete tree query has to be evaluated on the database. The translation process is described in detail in [11,13]. It relies on building trees consistent with the mapping relation. In this chapter, we focus on illustrating by example the semantics of the queries and the query translation process.

Consider the following database schema composed of an abstract tree type and a concrete tree type as given in Figure 7 with the following mapping relation:

```
Movie <-> Film          Movie/ActorActress <-> Film/Casting
Movie/ActorActress/Name <-> Film/Character
Movie/ActorActress/Name <-> Film/Casting/Actor
Movie/ActorActress/Role <-> Film/Character/Name
```

An instance of the database is also given in Figure 7 with only one data tree.

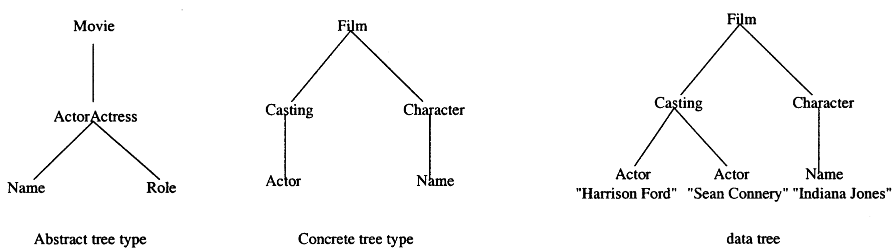

Fig. 7. An abstract tree type, a concrete tree type and a data tree

Figure 8 gives three examples of abstract tree queries that are conform to the abstract tree type of Figure 7.

The interpretation of the first one (Query Q_1) is: find all the data trees rooted at a node labelled by `Movie`, such that there exists a subtree rooted in a node labelled by `ActorActress` which contains two branches, the one rooted in a node labelled by `Name` must contain the string “Sean Connery” and the one rooted in a node labelled by `Role` must contain the string “Indiana Jones”; for those data trees returns the identity of subtrees rooted in the nodes labelled by `Name` and `Role`.

The second one (Query Q_2) has a slightly different interpretation: find all the data trees rooted at a node labelled by `Movie` such that there exists one subtree rooted in a node labelled by `ActorActress` having a branch rooted in a node

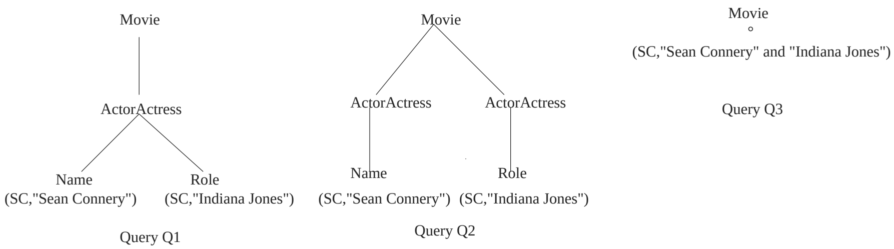

Fig. 8. Example of tree queries

labelled by `Name` which contains the string "Sean Connery" , and there exists another (possibly distinct) subtree rooted in a node labelled by `ActorActress` having a branch rooted in a node labelled by `Role` which contains the string "Indiana Jones".

The last tree query (Query Q_3) is an example where we have a non atomic boolean condition `"Sean Connery" and "Indiana Jones"`. This boolean condition expresses that the corresponding data subtree must contain the two strings `"Sean Connery"` and `"Indiana Jones"`. When talking about the interpretation of an abstract query, we must understand that these data trees exist only virtually. As explained in the database model, the real data trees are associated with concrete tree types.

For the translation process, some nodes in the tree query are crucial: they are nodes with conditions, nodes that are selected, or "join" nodes.

Definition 3. *Necessary nodes in a tree query A node in a tree query is* **necessary** *iff it is the root, a selected node, a conditional node or a node (called a join node) which has, at least, two distinct descendants that are necessary nodes. For a tree query Q, we denote $N(Q)$ the set of its necessary nodes.*

The necessary nodes can be partially ordered by the relation $\prec$. Let u and v be two necessary nodes: $u \prec v$, iff u is a descendant of v and there is no necessary node w such that $u \prec w \prec v$.

The translation of an abstract query Q into a concrete query can be summarized as follows. The concrete query has exactly the same tree structure as the translated abstract query but the nodes of the abstract tree query must be mapped by a function h onto the nodes of the concrete tree query such that the labelling of necessary nodes of the abstract tree query and the concrete tree query is conform to the mapping relation. More precisely: for two necessary nodes u and v in $N(Q)$ such that $u \prec v$, their images by the translation h must satisfy that $path(h(v))$ is a prefix of $path(h(u))$ (denoted $path(h(v)) \lhd path(h(u))$).

An abstract query can have more than one translation, depending on the mapping relation. It may also be possible that an abstract query has no translation., as it is illustrated in the following example.

Example 7. We are considering the abstract queries given in Figure 8 and the database composed of a unique data tree instance d as presented in Figure 7. Each abstract path to any necessary node of the query must be converted into a concrete path using the mapping relation. For example, if we consider the abstract tree query Q_1, its necessary nodes, u_1, u_2, u_3, and u_4, respectively labelled by `Movie`, `ActorActress`, `Name` and `Role` , are such that: $u_3 \prec u_2$ and $u_4 \prec u_2$. The corresponding abstract paths leading to the necessary nodes are: `Movie`, `Movie/ActorActress`, `Movie/ActorActress/Name`, and `Movie/ActorActress/Role`. By the mapping relation given in Example **??**, they must be mapped to the concrete paths `Film`, `Film/Casting`, `Film/Casting/ Actor` (or `Film/Character`) and `Film/Character/Name` respectively. This suggests a candidate translation based on an homomorphism h such that: $h(u_1)$ is labelled by `Film`, $h(u_2)$ is labelled by `Casting`, $h(u_3)$ is labelled by `Actor` or by `Character`, and $h(u_4)$ is labelled by `Name`. However, in order that candidate translation to be a real translation, since $u_4 \prec u_2$, $path(h(u_2))$, i.e., `Film/Casting`, should be a prefix of $path(h(u_4))$, which must be `Film/Character/ Name`. Since `Film/Casting` is not a prefix of `Film/Character/Name`, there is no translation, and thus no answer, for the query Q_1.

If we consider now the abstract query Q_2, its necessary nodes are v_1, v_2 and v_3, respectively labelled by `Movie`, `Name` and `Role`. The abstract paths leading to those necessary nodes are: `Movie`, `Movie/ActorActress/Name`, and `Movie/ActorActress/Role`. Based on the mapping relation, they can be converted respectively into `Film`, `Film/Casting/Actor` and `Film/Character/Name`. This leads to a translation Q'_2 corresponding to an homomorphism h' such that $h'(v_1)$ is labelled by `Film`, $h'(v_2)$ is labelled by `Actor` and $h'(v_3)$ is labelled by `Name`. In this case, h' satisfies the prefix property since: $v_2 \prec v_1$ and $path(h(v_1))$ (i.e., `Film`) is a prefix of $path(h(v_2))$ (i.e., `Film/Casting/Actor`), and $v_3 \prec v_1$ and $path(h(v_1))$ (i.e., `Film`) is a prefix of $path(h(v_3))$ (i.e., `Film/Character/Name`). The resulting concrete query Q'_2 is given in Figure 9.

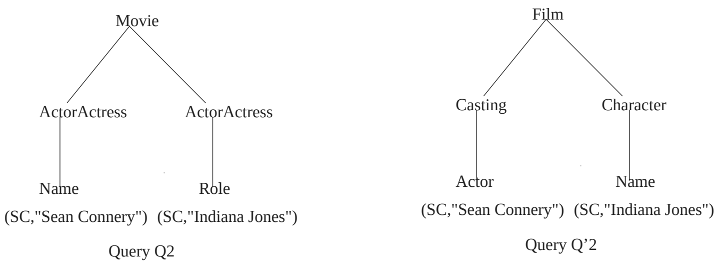

Fig. 9. The abstract query Q_2 and its translation Q'_2

It can then be evaluated against the data tree d: there exists a valuation σ_d, where the node labelled with `Actor` can be associated with the leaf of d labelled by `Actor` and valued by `"Sean Connery"` and the other node labelled by `Name`

can be associated with the leaf of d labelled by `Name` and valued by `"Indiana Jones"`.Therefore, the query Q'_2 has at least this answer.
For the abstract query Q_3 the pattern of the translation Q'_3 is reduced to one node labelled with `Film`. The whole data tree d matches also the query Q'_3 because it is rooted at a node labelled by `Film` which contains the strings `"Sean Connery"` and `"Indiana Jones"` by application of the closure valuation principle.

3.3 Automatic Generation of Mappings

We have seen previously that the mapping relationship is a crucial element of the data model for translating abstract queries into concrete ones. In this section, we address the problem of automatically finding the elements of the mapping relation which states the correspondence between paths of concrete tree types and paths of the abstract tree type. We have to handle heterogeneous names and structures because concrete tree types come from various sources designed by different persons who made personal choices on the names of the labels and on the tree structures. Therefore, automatic mapping generation have to deal with semantic and structure heterogeneity.

Below, we present the method for automatic mapping generation in Xyleme, based on two kinds of criteria: syntactic/semantic and structural. We report the first results of an experiment with a prototype, SAMAG, used to generate mappings for XML documents in the cultural domain.

Syntactic and Semantic Matching: Automatic mappings generation is based on term matching. The idea is that a mapping is generated only if the abstract and the concrete terms identified by the mapped paths are semantically related. We check two kinds of relations between terms: syntactic and semantic matching.

Syntactic matching concerns the syntactic inclusion of a term, or of a part of a term, into another one. For example, the concrete term `Actor` is syntactically similar to the abstract term `Actor-Actress`. This method also includes techniques for detecting abbreviations, such as `nb` for `Number`, etc. For more details, see [29].

Semantic matching is based on the use of extra knowledge available through existing ontologies or thesauri. In our experimentation, we chose the WordNet thesaurus [23] .

WordNet groups English nouns, verbs, adjectives and adverbs into sets of synonyms that are linked through semantic relations. Each set of synonyms (synset) represents a concept. A word may belong to several synsets, each one representing a particular sense. Our approach only exploits semantic links between nouns because we chose to use only abstract terms that are nouns. The relationships used in our prototype are listed and illustrated in Figure 10.

Dealing with Structure: Syntactic / semantic matching between words is not enough to obtain precise mappings. The meaning of a term depends on the place it occupies in the concrete tree type, which defines the interpretation context of

Semantic relationship	Linked terms
Synonymy	Movie - Film, Picture - Image Each couple of nouns belongs to the same synset
Hyponymy / Hypernymy	Painting - Portrait, because portrait is a kind of painting. Portrait is a hyponym of Painting and Painting is a hypernym of Portrait.
Meronymy / Holonymy	Movie - Scene, because a movie is composed of several scenes. Scene is a meronym of Movie and Movie is a holonym of Scene.
Hyponymy composed with hypernymy	Painting - Dance, because Art is a hypernym of both Painting and Dance

Fig. 10. Semantic relationships used in the SAMAG prototype

the word. More precisely, a term may occur several times in the same concrete tree type with different senses. Its meaning may be influenced by the meaning of its predecessors in the tree (e.g. `Name` may be the name of an artist, of a museum, etc.).

Our approach is based on a very simple hypothesis: the terms of a concrete tree type are either object names (e.g. `Painting`), or property names (e.g. `Name`). We consider that property names have a meaning only in the context of the object that they characterise. So, an object, which is characterised by a set of properties, defines the interpretation context of its properties.

In order to decide if a term is an object or a property node, we use heuristics based on the translation of a concrete tree type into a conceptual database schema. We use conceptual schemas built according to the Entity-Relationship formalism, that naturally helps to determine the classes, called entities, in a domain. An entity is a class of instances that share the same properties called attributes. In our approach, we consider that object nodes are similar to entities and that property nodes are similar to attributes. Given a link A - B in the concrete tree type, one heuristics used in the translation process is that B may refer to an entity (according to the above definition) only if B has son nodes, these ones being interpreted as B's attributes. Such a heuristics leads to consider that leaves in a concrete tree type always are property nodes (they can't refer to entities, so they can't be object nodes). Other heuristics are used to qualify internal nodes in a concrete tree type, in order to decide if they are object or property nodes. The exhaustive translation mechanism is described in [29]. However, very often, internal nodes are object nodes.

The basic idea underlying our structural constraints is that a mapping of a property node must be "compatible" with the mapping of the object node it characterizes. In other words, because the property and the object nodes in the concrete tree type are related (the object defines the context of the property), the corresponding abstract terms must be related in the same context-based manner.

Thus, if a concrete path identifying a property name is mapped to an abstract path X, this mapping will be valid only if the object characterized by that property is mapped to a predecessor of X in the abstract tree type. This introduces a context constraint into the mapping generation process, because a property name P of an object O may only be mapped to the subset of abstract terms placed below the abstract terms mapped to O, i.e. the context of the object is transmitted to the property. This also means that mappings for object names must be computed first.

For instance, in the concrete tree type MyDTD, the terms `Title`, `Author` and `Museum` are properties of the object `Painting`. The mappings:

```
Culture/Art/WorkOfArt/Title <-> MyDTD/Painting/Title
Culture/Art/WorkOfArt/Author <-> MyDTD/Painting/Author
```

are valid, because there exists a mapping between the object `MyDTD/Painting` and the abstract term `Culture/Art/WorkOfArt`.

Notice that this context-based constraint may seem too strong in some cases. For instance, the automatic generation algorithm will not find the mapping `Culture/Art/Museum <-> MyDTD/Painting/Museum` if there is no mapping from `MyDTD/Painting` to a prefix of `Culture/Art/Museum`. Imagine that the system does not semantically match `Painting` with `Art`, then the "obvious" mapping `Culture/Art/Museum <-> MyDTD/Painting/Museum` is missed. However, a more careful analysis of this case shows that the missed mapping is not so obvious, because it is not clear if `Culture/Art/Museum` is a museum with paintings.

The structural, context-based constraints for automatic mapping generation significantly improve precision if mappings are correctly found for object nodes. On the other hand, an error at an object node will propagate to all its property nodes.

Experiment: A prototype, SAMAG (Semi-Automatic MApping Generator), has been implemented in Java. We report here the first results of this work.

The corpus of concrete tree types used in our experiment is related to the cultural domain. An abstract tree type on this domain has been written and mappings have been established manually between this abstract tree type and all the concrete tree types associated with real DTDs of the corpus. We used these manual mappings as a reference set for evaluating the results of the SAMAG system.

SAMAG is a semi-automatic tool. User-validation is necessary each time a syntactic relationship between an abstract term and a concrete term is detected. Indeed, two syntactically similar terms may refer to different concepts and may not have the same meaning. Only a human is actually able to guarantee the semantic consistency of such a mapping.

SAMAG has been implemented as a modular system that can be parameterised. This makes the evaluation of our method easier because results obtained with various parameters from the same inputs can be compared. SAMAG's parameters are:

1. The semantic relationships used by the system to detect a semantic link between abstract and concrete terms, such as synonymy, hypernymy, holonymy, etc.
2. The structural constraints to apply on terms of concrete tree types. One can decide to apply or not these constraints. Furthermore, they can be applied only on property nodes in order to consider that property names have a meaning only in the context of the object that they characterise. But they can also be applied on all the nodes of a concrete tree type considering that the meaning of object names is also influenced by the meaning of their predecessors in the tree.

Notice that in all the cases, the system tests the existence of a syntactic relationship between an abstract term and a concrete term.

The results of the automatic mappings generation process are analysed according to (1) the number of relevant mappings among those automatically generated, and (2) the number of mappings manually created that are retrieved by the system. In particular, we study the following categories of mappings:

1. Mappings that are both manually and automatically generated, such as `Culture/Art/Artist <-> School/Painters_list/Painter`[4]. Such a mapping is generated thanks to a synonymy relationship between `Painter` and `Artist`,
2. Mappings manually created that are not retrieved by the SAMAG system, such as `Culture/Art/Artist/ArtMovement <-> School/Name`. Here, there is no semantic relationship in WordNet between `ArtMovement` and `Name`, and the detection of a similarity relationship between an abstract term and a concrete term is a precondition to generate a mapping,
3. Irrelevant mappings, that are automatically but not manually generated, such as `Culture/Cinema/Movie/Picture <-> Painting/Image`. Here, there is a synonymy relationship between `Picture` and `Image`, but `Picture` is used in a cinema context and `Image` is used in a painting context. SAMAG is unable to detect such context differences, because `Image` is not a property node. Other irrelevant mappings may be generated because of the polysemy of terms (a word may have different meanings), not handled by SAMAG.
4. Mappings automatically generated, that may be considered relevant, even if they have not been found manually, such as `Culture/Art <-> Painting`. The above discussion about the mapping `Culture/Art/Museum <-> MyDTD/Painting/Museum` demonstrates that several meanings are possible for a term in this context, and that sometimes the semantics is a matter of choice. We also considered as relevant mappings that "have a sense", such as `Culture/Art/Artist <-> School/Painters_list`, but that were considered as useless at manual mappings creation because there is a tag `Painter` below `School/Painters_list`.

The table in Figure 11 gathers some significant statistical results about the mappings generated by the SAMAG system.

[4] The abstract tree type in Figure 5 is just a part of the one used in this experiment.

Semantic relationships	All	Synonymy	All
Structural constraints	None	All	Property nodes
Generated mappings	683	32	267
Relevant mappings	145	28	115
Percentage of relevant mappings	21%	87.5%	43%
Manual mappings retrieved	74 / 111	18 / 111	52 / 111

Fig. 11. Behavior of the SAMAG prototype

Results analysis: The first column in the table describes results that are obtained when all the semantic relationships in WordNet are used and no structural constraints are applied on terms of the concrete tree types. With this configuration, one obtains the maximum number of mappings that SAMAG can generate. The only condition to generate a mapping is the existence of a similarity relationship (semantic or syntactic) between abstract and concrete terms. In this configuration a lot of mappings are not relevant because the system does not consider the interpretation context of terms.

The second column in the table describes the results when SAMAG is highly constrained. We consider that the meaning of all terms in the tree is influenced by the meaning of its predecessor. Moreover, only one semantic relationship, synonymy, is checked between terms. It is the relation that introduces the lowest distance between terms. In this configuration, almost all the generated mappings are relevant. However, few mappings are generated, so few manual mappings are retrieved.

The last column in the table seems a good trade-off between the number of relevant mappings and the number of mappings automatically retrieved by SAMAG among manual mappings. In this configuration, all the semantic relationships in WordNet are used and only property names are considered to have a meaning in the context of their predecessor in the tree.

The relatively high number of mappings not found by SAMAG (33%) is explained by the fact that many manually created mappings are based on corpus-specific knowledge, e.g. `Culture/Art/WorkOfArt/Artist <-> XML_DOC/Document/Title`, because a document `XML_DOC` is about a painter whose name is contained in the document's title `XML_DOC/Document/Title`. Manually created mappings take advantage of such knowledge, while SAMAG cannot. Other mappings are missed because the concrete terms are non-standard abbreviations instead of significant nouns. The last category of manual mappings missed by the system are mappings that connect an object to some identifying property, e.g. a director and his name, such as in `Culture/Cinema/Director/Name <-> List/Movie/Directed_by/Director`.

The major issue explored in this experiment is the way the interpretation context of terms involved in mappings should be considered in an automated process. We proposed a solution based on structural constraints and the analysis of our results must be viewed as a contribution to better understand how to take context into account.

First, we conclude that structural constraints are necessary to limit the number of generated mappings and to produce relevant ones. The number of irrelevant mappings is too high when no structural constraint is applied (145 relevant mappings among 683 when the configuration in column three gives 115 relevant mappings among 267).

The results also show that applying contextual or structural constraints only on property nodes is not enough restrictive. In the last column of the table in Figure 11, more than 50% of automatically generated mappings are not relevant. An analysis of the irrelevant mappings shows that most of terms involved in bad mappings are objects nodes. Thus, structural constraints must also be applied on object nodes. However, if we apply structural constraints on all the nodes (see results in the second column), the system becomes too constrained and does not generate enough mappings.

The question is then how to define the interpretation context for object nodes. In many cases, the context of an object node is not only given by its predecessor node within the associated path. For an object node, the number of nodes within the path that are significant to define its context of interpretation depends on it, so this number is variable from an object node to another. For instance, the context of interpretation of the object node `Actor` in `Culture/Cinema/Movie/Actor` is `Movie` whereas the context of interpretation of the object node `Museum` in `Culture/Painting/Oil/Museum` is `Painting, Oil`.

The results also depend on the extra knowledge used to establish semantic relationships between terms. WordNet was used as a resource to look for semantic relationships among terms. This choice has been principally motivated by the fact that data in WordNet are easily accessible for automatic applications. It offers a large coverage of general lexicon in English. However, because a term in natural language has often many meanings, WordNet returns synsets for all the senses of a given term. However, we don't know how to select automatically the right sense of a term, because it directly depends of its interpretation context. So when the system, for instance, asks to WordNet the set of synonyms of a given term, this one, being not domain-specific, takes synonyms of all the senses of the term.

An important remark concerns the influence of mapping generation on query translation. Of course, if a relevant mapping is missed by SAMAG, the query processor will miss the results concerning that mapping. However, additional irrelevant mappings do not necessarily produce irrelevant answers because query translation need all the mappings in the translation of the query tree to be compatible (i.e. to form a tree that respects the descendant relationship among terms). Yet, reducing the number of irrelevant mappings is important for storage saving reasons, which are critical for web-scale domains.

4 Conclusion and Perspectives

PICSEL and Xyleme are two mediator-based information integration agents that have been designed in very different settings. As a result, different choices have been done for designing the mediated schema and the semantic mappings between the mediated schema and the schemas of the sources.

In PICSEL, a logical formalism combining the expressive powers of (non recursive) DATALOG rules and the $\mathcal{ALN}$ description logic has been used for modeling the semantic relationships between source schemas through a rich mediated schema. It has proved very useful for expressing fine-grained differences between sources containing closely related data. This choice is appropriate for building specialized information servers over a reasonable number of data sources.

Xyleme is representative of information integration agents that have to integrate a huge number of sources covering very broad topics. In such a setting, the mediated schema and the mappings must be simple because the challenging issue is to obtain them as automatically as possible. In particular, the number of mappings is too huge to build them manually.

We have outlined the impact of the choice of the knowledge representation formalism on the query reformulation problem, which is the core algorithmic problem for answering queries in an information integration system. Clearly, as the languages for describing data sources, the mediated schema, or the users' queries become more expressive, the query reformulation problem becomes harder. The key challenge is then to identify formalisms offering a reasonable tradeoff between expressive power and good computational properties for the accompanying reformulation algorithm.

Despite their differences, PICSEL and Xyleme both illustrate a centralized approach of mediation based on a single mediated schema. For scaling up to the Web, this centralized approach of mediation is probably not flexible enough, and *distributed systems of mediation* are more appropriate.

The Semantic Web envisions a world-wide distributed architecture where data and computational resources will easily interoperate to coordinate complex tasks such as answering queries or global computing. Semantic marking up of web resources using *ontologies* is expected to provide the necessary glue for making this vision work. I think that the de-centralized nature of the Web makes inevitable that communities of users or software developers will use their own ontologies to describe their data or services. In this vision of the Semantic Web based on distributed ontologies, the key point is the mediation between data, services and users, using mappings between ontologies. Complex mappings and reasoning about those mappings are necessary for comparing and combining ontologies, and for integrating data or services described using different ontologies. For an easy deployment of distributed peer to peer data management systems at the scale of the Web, it will be essential to use expressive and declarative languages for describing semantic relationships between ontologies serving as schemas of distributed data or services. CARIN has the potential to be a useful modeling tool for expressing and reasoning on complex mappings between ontologies.

Acknowledgments. We would like to thank the PICSEL and Xyleme teams for many fruitful discussions and particularly Serge Abiteboul, Bernd Aman, Sophie Cluet, Claude Delobel, Christine Froidevaux, Irini Fundulaki, Francois Goasdoue, Tova Milo, Brigitte Safar, Jean-Pierre Sirot, Pierangelo Veltri and Dan Vodislav.

References

[1] V. Aguiléra, S. Cluet, P. Veltri, D. Vodislav, and F. Wattez. Querying the XML Documents on the Web. In *Proc. of the ACMSIGIR Workshop on XML and I. R., Athens*, July 28, 2000.

[2] B. Amann, I. Fundulaki, and M. Scholl. Integrating Ontologies and Thesauri for RDF Schema Creation and Medata Querying. *International Journal of Digital Libraries*, 2000.

[3] Yigal Arens, Chung-Nan Hsu, and Craig A. Knoblock. Query processing in the SIMS information mediator. In Austin Tate, editor, *Advanced Planning Technology*, pages 61–69. AAAI Press, Menlo Park, California, 1996.

[4] Yigal Arens and Craig A. Knoblock. SIMS: Retrieving and integrating information from multiple sources. In Peter Buneman and Sushil Jajodia, editors, *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 562–563, Washington, D.C., 26–28 May 1993.

[5] Domenico Beneventano, Sonia Bergamaschi, Silvana Castano, Alberto Corni, R. Guidetti, G. Malvezzi, Michele Melchiori, and Maurizio Vincini. Information integration: The MOMIS project demonstration. In *VLDB 2000, Proceedings of 26th International Conference on Very Large Data Bases, September 10–14, 2000, Cairo, Egypt*, pages 611–614.

[6] A. Bidault, C. Froidevaux, and B. Safar. Repairing queries in a mediator approach. In *14th European Conference on Artificial Intelligence*, pages 406–410, Berlin, 2000.

[7] A. Borgida, R. J. Brachman, D. L. McGuinness, and L. Alperin Resnick. CLASSIC: a structural data model for objects. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, 18(2):58–67, June 1989.

[8] Diego Calvanese, Giuseppe De Giacomo, G. D., and Maurizio Lenzerini. Answering queries using views in description logics. In *Proceedings of AAAI 2000.*

[9] Sudarshan Chawathe, Hector Garcia-Molina, Joachim Hammer, Kelly Ireland, Yannis Papakonstantinou, Jeffrey D. Ullman, and Jennifer Widom. The TSIMMIS project: Integration of heterogeneous information sources. In *16th Meeting of the Information Processing Society of Japan*, pages 7–18, Tokyo, Japan, 1994.

[10] J. Clark and S. DeRose (eds). *XML Path Language (XPath).* W3C Recommandation, 1999.
`http://www.w3c.org/TR/xpath`.

[11] Sophie Cluet, Pierangelo Veltri, and Dan Vodislav. Views in Large Scale XML Repository. In *Proc. Int. Conf. on Very Large Data Bases*, 2001.

[12] C-web project.
`http://cweb.inria.fr`.

[13] Claude Delobel, M-Christine Rousset, Chantal Reynaud, J-Pierre Sirot and Dan Vodislav. Semantic Integration in Xyleme: a Uniform Tree-based Approach. In *Journal on Data & Knowledge Engineering*, to appear.

[14] Oren Etzioni and Daniel Weld. A Softbot-Based Interface to the Internet. *Communications of the ACM*, 37(7):72–76, 1994.

[15] Marc Friedman and Daniel S. Weld. Efficiently executing information-gathering plans. In *15th International Joint Conference on Artificial Intelligence*, pages 785–791, Nagoya, Japan, 1997.

[16] Michael R. Genesereth, Arthur M. Keller, and Oliver M. Duschka. Infomaster: an information integration system. In *Proceedings of SIGMOD 97* pages 539–542, New York, May 1997.

[17] François Goasdoué and Marie-Christine Rousset. Compilation and Approximation of Conjunctive Queries by Concept Descriptions. in *Proceedings of ECAI 2002*.

[18] François Goasdoué, Véronique Lattes, and Marie-Christine Rousset. The Use of CARIN Language and Algorithms for Information Integration: The PICSEL System. *International Journal of Cooperative Information Systems*, 9(4):383–401, 2000.

[19] Alon Halevy. Answering queries using views: a survey. *VLDB Journal*, 2001.

[20] Carl-Christian Kanne and Guido Moerkotte. Efficient storage of xml data. Technical report, University of Mannheim, http://pi3.informatik.uni-mannheim.de/, 1999.

[21] Alon Levy and Marie-Christine Rousset. Combining Horn Rules and Description Logics in CARIN. *Artificial Intelligence Journal*, 104, pp 165–209, 1998.

[22] Laurent Mignet, Mihai Preda, Serge Abiteboul, Sebastien Ailleret, Bernd Amann, and Amelie Marian. Acquiring xml pages for a webhouse. In *BDA'00*, 2000.

[23] G. Miller. Wordnet: A lexical database for english. *Communications of the ACM*, 38(11), 1995.

[24] Benjamin Nguyen, Serge Abiteboul, Gregory Cobena, and Mihai Preda. Monitoring xml data on the web. In *VLDB'01*, 2001.

[25] J. Ordille, A. Levy, and A. Rajaraman. Querying Heteregeneous Information Sources Using Source Descriptions. In *Proc. Int. Conf. on Very Large Data Bases*, pages 251–262, 1996.

[26] R. MacGregor and R. Bates. The LOOM knowledge representation language. Technical Report ISI/RS-87-188, Information Science Institute, University of Southern California, Marina del Rey (CA), USA, 1987.

[27] E. Mena, Vipul Kashyap, Amit Sheth, and A. Illarramendi. OBSERVER: An approach for query processing in global information systems based on interoperation across pre-existing ontologies. In *4th Int. Conf. on Cooperative Information Systems*, pages 14–25, Bruessels, Belgium, 1996.

[28] Y. Papakonstantinou, H. Garcia-molina, and J. Widom. Object Exchange Across Heteregeneous Information Sources. In *ICDE Conf. on Management of Data*, 1995.

[29] Jean-Pierre Sirot. Documents xml et serveurs d'information: approche, techniques et outil d'indexation bases sur l'utilisation d'une ontologie. Technical report, Rapport de stage de DEA I3, University of Paris-Sud, 2000.

[30] V. S. Subrahmanian, Sibel Adali, Anne Brink, Ross Emery, James J. Lu, Adil Rajput, Timothy J. Rogers, Robert Ross, and Charles Ward. HERMES: A heterogeneous reasoning and mediator system. Technical report, University of Maryland, 1995.

[31] XQuery 1.0: An XML Query Language. `http://www.w3.org/TR/xquery/`.

[32] Xyleme. `http://www.xyleme.com`.

[33] Lucie Xyleme. A dynamic warehouse for xml data of the web. *IEEE Data Engineering Bulletin*, 2001.

Adaptive Information Extraction: Core Technologies for Information Agents

Nicholas Kushmerick[1] and Bernd Thomas[2]

[1] Computer Science Department, University College Dublin; nick@ucd.ie
[2] Institut für Informatik, Universität Koblenz-Landau; bthomas@uni-koblenz.de

1 Introduction

For the purposes of this chapter, an information agent can be described as a distributed system that receives a goal through its user interface, gathers information relevant to this goal from a variety of sources, processes this content as appropriate, and delivers the results to the users. We focus on the second stage in this generic architecture. We survey a variety of information extraction techniques that enable information agents to automatically gather information from heterogeneous sources.

For example, consider an agent that mediates package-delivery requests. To satisfy such requests, the agent might need to retrieve address information from geographic services, ask an advertising service for freight forwarders that serve the destination, request quotes from the relevant freight forwarders, retrieve duties and legal constraints from government sites, get weather information to estimate transportation delays, etc.

Information extraction (IE) is a form of shallow document processing that involves populating a database with values automatically extracted from documents. Over the past decade, researchers have developed a rich family of generic IE techniques that are suitable for a wide variety of sources, from rigidly formatted documents such as HTML generated automatically from a template, to natural-language documents such as newspaper articles or email messages.

In this chapter, we view information extraction as a core enabling technology for a variety of information agents. We therefore focus specifically on information extraction, rather than tangential (albeit important) issues, such as how agents can discover relevant sources or verify the authenticity of the retrieved content, or caching policies that minimize communication while ensuring freshness.

Before proceeding, we observe that neither XML nor the Semantic Web initiative will eliminate the need for automatic information extraction. First, there are terabytes of content available from numerous legacy services that will probably never export their data in XML. Second, it is impossible to determine "the" correct annotation scheme, and applications will have their own idiosyncratic needs ("should the unit of currency be included when extracting prices?", "should people's names be split into first and surname?", "should dates such as 'Sun. May 14, 67' be canonicalized to 14/05/1967?"). For these reasons we expect that automatic information extraction will continue to be essential for many years.

Scalability is the key challenge to automatic information extraction. There are two relevant dimensions. The first dimension is the ability to rapidly process large document

M. Klusch et al. (Eds.): Intelligent Information Agents, LNAI 2586, pp. 79–103, 2003.

collections. IE systems generally scale well in this regard because they rely on simple shallow extraction rules, rather than sophisticated (and therefore slow) natural language processing.

The second and more problematic dimension is the number of distinct sources. For example, a package-delivery agent might need to request quotes from a thousand different freight forwarders, weather information from dozens of forecast services, etc. IE is challenging in this scenario because each source might format its content differently, and therefore each source could require a customized set of extraction rules.

Machine learning is the only domain-independent approach to scaling along this second dimension. This chapter focuses on the use of machine learning to enable *adaptive information extraction* systems that automatically learn extraction rules from training data in order to scale with the number of sources.

The general idea behind adaptive information extraction is that a human expert annotates a small corpus of training documents with the fragments that should be extracted, and then the learning system generalizes from these examples to produce some form of knowledge or rules that reliably extract "similar" content from other documents. While human-annotated training data can be expensive, the assumption underlying adaptive IE is that it is easier to annotate documents than to write extraction rules, since the latter requires some degree of programming expertise. Furthermore, we will describe techniques aimed at minimizing the amount of training data required for generalization, or even eliminating the need for manual annotation entirely.

The adaptive information extraction research community has developed a wide variety of techniques and approaches, each tailored to particular extraction tasks and document types. We organize our survey of this research in terms of two distinct approaches. First, we describe *finite-state* approaches that learn extraction knowledge that is equivalent to (possibly stochastic) finite-state automata (Section 2). Second, we describe *relational* approaches that learn extraction knowledge that is essentially in the form of Prolog-like logic programs (Section 3). For the sake of brevity we can not describe these techniques in detail, but see [26] for more information about some of these ideas.

2 Finite-State Techniques

Many approaches to Web information extraction can be categorized as finite-state approaches, in that the learned extraction knowledge structures are formally equivalent to (possibly stochastic) regular grammars or automata. In this section we survey several prominent examples, as well as some additional research that relates to the entire wrapper "life-cycle" beyond the core learning task.

2.1 Wrapper Induction

Kushmerick first formalized adaptive Web information extraction with his work on *wrapper induction* [21,17,19]. Kushmerick identified a family of six wrapper classes, and demonstrated that the wrappers were both relatively expressive (they can learn wrappers for numerous real-world Web sites), and also relatively efficient (only a handful of training examples, and a few CPU seconds per example, are needed for learning).

To illustrate Kushmerick's wrapper induction work, consider the example Web page shown in Figure 1(a), its HTML encoding (b), and the content to be extracted (c). This example is clearly extremely simple, but it exhibits all the features that are salient for our discussion.

Kushmerick's wrappers consist of a sequence of delimiters strings for finding the desired content. In the simplest case (shown in Figure 1(d–e)), the content is arranged in a tabular format with K columns, and the wrapper scans for a pair of delimiters for each column, for a total of $2K$ delimiters. The notation "ℓ_k" indicates the left-hand delimiter for the k'th column, and "r_k" is the k'th column's right-hand delimiter. In this case of the country-code wrapper $\texttt{ccwrap}_{\mathrm{LR}}$, we have $K = 2$.

To execute the wrapper, procedure $\texttt{ccwrap}_{\mathrm{LR}}$ (Figure 1(d)) scans for the string ℓ_1 =`<B>` from the beginning of the document, and then scans ahead until the next occurrence of r_1 =`</B>`. The procedure then extracts the text between these positions as the value of the first column of the first row. The procedure then scans for ℓ_2 =`<I>` and then for r_2 =`</I>`, and extracts the text between these positions as the value of the second column of the first row. This process then starts over again with ℓ_1; extraction terminates when ℓ_1 is missing (indicating the end of the document).

Figure 1(e) formalizes these ideas as the *Left-Right* (LR) wrapper class. An LR wrapper w_{LR} consists of a set $\{\langle\ell_1, r_1\rangle, \ldots, \langle\ell_K, r_K\rangle\}$ of $2K$ delimiters, one pair for each column to be extracted, and the "operational semantics" of LR are provided by the $\texttt{exec}_{\mathrm{LR}}$ procedure (Figure 1(e)). This procedure scans for ℓ_1 from the beginning of the document, and then scans ahead until the next occurrence of r_1. The procedure then extracts the text between these positions as the value of the first column of the first row. $\texttt{ccwrap}_{\mathrm{LR}}$ then scans for ℓ_2 and then for r_2, and extracts the text between these positions as the value of the second column of the first row. This process is repeated for all K columns. After searching for r_K, the procedure starts over again with ℓ_1; extraction terminates when ℓ_1 is missing (indicating the end of the document).

Given this definition, the LR machine learning task is to automatically construct an LR wrapper, given a set of training documents. LR learning is relatively efficient, because the $2K$ delimiters can all be learned independently. The key insight is that whether a particular candidate is valid for some delimiter has no impact on the other delimiters. Based on this observation, Kushmerick describes a quadratic-time algorithm for learning LR wrappers. The algorithm simply enumerates over potential values for each delimiter, selecting the first that satisfies a constraint that guarantees that the wrapper will work correctly on the training data. Kushmerick demonstrates (both empirically, and theoretically under the PAC model) that this algorithm requires a modest training sample to converge to the correct wrapper.

Of course, just because an efficient learning algorithm exists does not mean that the wrappers are useful! Below, we discuss the limitations of the LR class and show that it can not handle documents with more complicated formatting. However, even the very simple LR class was able to successfully wrap 53% of Web sites, according to a survey. While LR is by no means a definitive solution to Web information extraction, it clearly demonstrates that simple techniques can be remarkably effective.

LR is effective for simple pages, but even minor complications to the formatting can render LR ineffective. For example, consider ℓ_1. The LR class requires a value for

(a)

(b)

```
<HTML><TITLE>Some Country Codes</TITLE><BODY>
<B>Congo</B> <I>242</I><BR>
<B>Egypt</B> <I>20</I><BR>
<B>Belize</B> <I>501</I><BR>
<B>Spain</B> <I>34</I><BR>
</BODY></HTML>
```

(c)

$$\left\{\begin{array}{c} \langle\text{`Congo'}, \text{`242'}\rangle, \\ \langle\text{`Egypt'}, \text{`20'}\rangle, \\ \langle\text{`Belize'}, \text{`501'}\rangle, \\ \langle\text{`Spain'}, \text{`34'}\rangle \end{array}\right\}$$

(d)

procedure ccwrap$_{\mathrm{LR}}$(page P)
  while there are more occurrences in P of '<B>'
    for each $\langle \ell_k, r_k\rangle \in \{\langle$'<B>', '</B>'$\rangle$, $\langle$'<I>', '</I>'$\rangle\}$
      scan in P to next occurrence of ℓ_k; save position as start of k'th attribute
      scan in P to next occurrence of r_k; save position as end of k'th attribute
  return extracted $\{\ldots, \langle$country, code$\rangle, \ldots\}$ pairs

(e)

procedure exec$_{\mathrm{LR}}$(wrapper $w_{\mathrm{LR}} = \{\langle \ell_1, r_1\rangle, \ldots, \langle \ell_K, r_K\rangle\}$, page P)
  $m \leftarrow 0$
  while there are more occurrences in P of ℓ_1
    $m \leftarrow m + 1$
    for each $\langle \ell_k, r_k\rangle \in \{\langle \ell_1, r_1\rangle, \ldots, \langle \ell_K, r_K\rangle\}$
      scan in P to the next occurrence of ℓ_k; save position as $b_{m,k}$
      scan in P to the next occurrence of r_k; save position as $e_{m,k}$
  return label $\{\ldots, \langle\langle b_{m,1}, e_{m,1}\rangle, \ldots, \langle b_{m,K}, e_{m,K}\rangle\rangle, \ldots\}$

Fig. 1. A fictitious Internet site providing information about countries and their telephone country codes: (a) an example Web page; (b) the HTML document corresponding to (a); (c) the content to be extracted; (d) the ccwrap$_{\mathrm{LR}}$ procedure, which generates (c) from (b); and (e) the exec$_{\mathrm{LR}}$ procedure, a generalization of ccwrap$_{\mathrm{LR}}$.

ℓ_1 that reliably indicates the beginning of the first attribute. However, there may be no such delimiter. For example, suppose that Figure 1(b) was modified to include a heading `<B>Country code list</B>` at the top of the document. In this case the delimiter $\ell_1 =$<B> used by ccwrap$_{\mathrm{LR}}$ would not work correctly. Indeed, it is possible to show that there is no legal value for ℓ_1 and hence no LR wrapper for documents modified in this manner.

Kushmerick tackled these issues by extending LR to a family of five additional wrapper classes. First, the *Head-Left-Right-Tail* (HLRT) class uses two additional delimiters to skip over potentially-confusing text in either the head (top) or tail (bottom) of the page. In the example above, a head delimiter h (such as h =`list`) could be used to skip over the initial `<B>` at the top of the document, enabling ℓ_1 =`<B>` to work correctly. Alternatively, the *Open-Close-Left-Right* (OCLR) class uses two additional delimiters to identify an entire tuple in the document, and then uses the regular LR strategy within this mini-document to extract each attribute in turn. These two ideas can be combined in fourth wrapper class, the *Head-Open-Close-Left-Right-Tail* (HOCLRT) class.

Finally, Kushmerick explored two simple wrappers for data that is not formatted in a simple tabular fashion. The *Nested-Left-Right* (NLR) class can be used to extract hierarchically-organized data, such as a book's table of contents. NLR operates like LR except that, after processing r_k, there are $k+1$ possibilities (start at level $k+1$, continue level k, return to level $k-1, \ldots,$ return to level 1) instead of just one (proceed to attribute $k+1$). The *Nested-Head-Left-Right-Tail* (NHLRT) class combines NLR and HLRT.

Kushmerick developed specialized learning algorithms for each of these five classes. He demonstrated, both empirically and using complexity theory, that there is a trade-off between the expressive power of the wrapper classes and the extent to which they can be efficiently learned. For example, even though the six classes can successfully wrap 70% of surveyed sites, the algorithms for learning NLR and NHLRT wrappers take time that grows exponentially in the number of attributes, and a PAC analysis reveals that HOCLRT requires substantially more training examples to converge compared to the other classes.

2.2 More Expressive Wrapper Classes

Following Kushmerick's initial investigation of the LR family of wrappers, there has been substantial research effort at elaborating various alternative wrapper classes, and deriving more efficient learning algorithms. Even when Kushmerick's various extended wrapper classes are taken into consideration, there are numerous limitations. Muslea et al [27], Hsu and Dung [14], and others have developed various wrapper-learning algorithms that address the following shortcomings:

Missing attributes. Complicated pages may involve missing or null attribute values. If the corresponding delimiters are missing, then a simple wrapper will not process the remainder of the page correctly. For example, a French e-commerce site might only specify the country in addresses outside France.

Multi-valued attributes. The simple wrapper classes discussed so far assume a simple relational model in which each attribute has a single value, but non-relational structures such as multi-valued attributes are natural in many scenarios. For example, a hotel guide might explicitly list the cities served by a particular chain, rather than use a wasteful binary encoding of all possible cities.

Multiple attribute orderings. The wrappers described so far assume that the attributes (and therefore the delimiters) will occur in one fixed ordering, but variant orderings abound in complicated documents. For example, a movie site might list the release date before the title for movies prior to 1999, but after the title for recent movies.

Disjunctive delimiters. The wrappers discussed above assume a single delimiter for each attribute, but complicated sites might use multiple delimiters. For example, an e-commerce site might list prices with a bold face, except that sale prices are rendered in red.

Nonexistent delimiters. The wrappers described earlier assume that some irrelevant background tokens separate the content to be extracted, but this assumption may be violated in some cases. For example, how can the department code be separated from the course number in strings such as “COMP4016” or “GEOL2001”. This problem is also relevant for many Asian languages in which words are not tokenized by spaces.

Typographical errors and exceptions. Real-world documents may contain errors, and if these errors occur in the formatting that drives extraction, then a simplistic wrapper may fail on the entire page even if just a small portion is badly formatted.

Sequential delimiters. So far, the wrapper classes above assumed a single delimiter per attribute, but the simplest way to develop an accurate wrapper might be to scan for several delimiters in sequence. For example, to extract the name of a restaurant from a review it might simpler to scan for `<B>`, then to scan for `<BIG>` from that position, and finally to scan for `<FONT>`, rather than to force the wrapper to scan the document for a single delimiter that reliably indicates the extracted content.

Hierarchically organized data. Kushmerick’s nested classes are a first step at handling non-tabular data, but his results are largely negative. In complicated scenarios there is a need extraction according to a nested or embedded structure.

Hsu and Dung [14] addresses the problem of learning wrappers that correspond to an expressive class of deterministic finite-state transducers. This formalism handles all but the last two requirements just mentioned. The transducer processes the document to extract a single tuple; after extraction control returns to the start state and the second tuple is extracted, etc. Each extracted attribute is represented as a pair of states: one state to identify the start of the attribute value and the second to identify the end.

Since a general automaton model is used, states can be connected in an arbitrary manner, permitting missing attributes (skipped states), multi-valued attributes (cycles) and multiple attribute orderings (multiple paths from the start to end state). Furthermore, state-transitions are governed by an expressive rule language that allows disjunctive delimiters. A limited form of exception-processing is permitted, allowing the system to recover from formatting errors and exceptions. Crucially, Hsu and Dung describe an algorithm for efficiently learning their wrapper transducers from training data. Empirically, the report that their wrapper classes handles the 30% sites that could not be wrapped by Kushmerick’s wrapper classes.

Muslea et al [27] identify a class of wrappers that, unlike Hsu and Dung, tackle the last two issues mentioned above. The main distinguishing feature of Muslea et al’s wrappers is the use of multiple delimiters that they call landmarks. Rather than insisting that there exist a single delimiter that exactly identifies the relevant position deep inside some document, landmark-based wrappers use a sequence of delimiters to jump to the appropriate position in a series of simple steps. These simple steps are usually easier to learn, and enable more robust extraction. A second major feature of Muslea et al's work is that their "embedded catalog" formalization of nested data is more expressive than the simple hierarchical approach used by Kushmerick.

2.3 Extraction from Natural Text

The techniques described so far are aimed at highly regular documents, such as machine-generated HTML emitted by CGI programs. However, most research on information extraction has focused on natural free-text documents, such as email messages, newspaper articles, resumes, etc. Are the "wrapper" results relevant to these less structured domains. Several recent investigations have shown promising results.

Freitag and Kushmerick [11] explore "boosted wrapper induction". They define a class of extraction patterns that is essentially the LR class, for the case when there is exactly $K = 1$ attributes. They then enrich this class by permitting delimiters to contain wild-cards over token types (eg, <Num> rather than specific instances such as 23).

For example, for a corpus of email seminar announcements, the algorithm learns the following rule for extracting the starting time: {([time :],[<Num>]), ([], [- <Num> : <*> <Alpha>])}, which matches a document such as "... Time: 2:00 - 3:00 pm ...", where the fragment to be extracted has been underlined. This rule basically says "to find the start of the time, look for 'time:' followed by any number; then find the end of the time by looking for a dash, another number, a colon, any token at all, and finally an alphanumeric token".

This simple rule language is by itself not very useful for extraction from free text. Freitag and Kushmerick improve the performance by using boosting (a general technique for improving the accuracy of a weak learning algorithm) to learn many such rules. Each individual rule has high precision, but low recall; when combined, the rule set has both high precision and high recall. The result is an accurate extraction algorithm that is competitive with other state-of-the-art approaches in a variety of free-text domains, and superior in many. For example, boosted wrapper induction performs essentially perfectly at the task of extracting seminar announcement times, and better than most competitors at other attributes such as the speaker name and seminar location.

Soderland [32] describes a related approach to using finite-state techniques for information extraction from free text. Soderland's extraction rules correspond to a restricted class of regular expressions. These regular expressions serve two purposes: they can be both contextual pattern for determining whether a particular fragment should be extracted, or delimiters for determining the precise boundaries of the target fragment. Soderland's language is important because it is designed to work for documents that span the spectrum from unstructured natural text through to highly structured Web pages. Depending on the degree of structure in the training documents, the learning algorithm automatically creates appropriate patterns. For example, if simple delimiter-based extraction is sufficiently accurate then the learning algorithm will not bother to add additional contextual constraints.

For example, consider extracting the price and number of bedrooms from apartment listing documents such as "Capitol Hill- 1 br twnhme. D/W W/D. Pkg incl $675. 3BR upper flr no gar. $995. (206) 999-9999". Soderland's system learns rules such as "* (<Digit>) 'BR' * '$' (<Numb>)", where the parenthesized portions of the regular expression indicate the values to be extracted. This rule would extract the content {(1, 675), (3, 995)} from the example document.

2.4 Hidden Markov Models

The work of Freitag and Kushmerick [11] and Soderland [32] are two instances of generalizing finite-state approaches from rigidly structured HTML documents to less structured documents such as email and newspaper articles. However, these approaches are still brittle because they do not have any facility for evaluating the strength of the evidence that guides extraction decisions. For example, suppose the phrase `will be held in` often precedes a seminar location, but a new document contains the typographical error `will held in`. The techniques described so far make binary decisions and thus have no way to use this uncertain evidence.

Hidden Markov Models are a principled and efficient approach to handling this sort of inherent uncertainty. A Hidden Markov Model (HMM) is a stochastic finite-state automaton. States emit tokens according to a fixed and state-specific distribution, and transitions between states occur according to a fixed distribution. HMMs are an attractive computational device because there are efficient algorithms for both learning the model's distribution parameters, and for inferring the most-likely state sequence given some observed token sequence.

To use HMMs for information extraction, states are associated with the tokens to be extracted. For example, with the email seminar announcement corpus, the HMM would contain a state for the start time tokens, the end time tokens, the speaker name tokens, and the location tokens. Optionally, there may be additional states that generate "background" tokens. To perform extraction, the standard HMM Viterbi decoding algorithm is used to determine the most-likely state-sequence to have generated the observed document, and then the extracted fragments can simply by read off this most-likely path.

Hidden Markov models have been used successfully by numerous researchers in a variety of extraction scenarios (eg, [3,22]). They key challenge is that there is no efficient general-purpose algorithm for determining an appropriate state topology (ie, which state-state distribution probabilities should be forced to be zero and which should be permitted to be positive). Initial work has generally used a hand-crafted topology, in which the states are connected manually in a "reasonable" way after evaluating the training corpus.

More recently, there have been several attempts to automatically learn an appropriate topology. The general approach is to greedily search the space of possible topologies for one that maximizes some objective function. Seymore et al [31] attempt to maximize the probability of the training data given the topology. This approach is reasonably efficient but potentially misguided: the goal of using an HMM is not to model the training data per se, but to perform accurate extraction. Freitag and McCallum [12] therefore use as the objective function the actual accuracy of the proposed topology for extraction from a held-out validation corpus. While this approach is significantly slower it can result in a more compact topology and better generalization.

2.5 Wrapper Maintenance

All of the wrapper-learning work described earlier ignores an important complication. Information agents generally have no control over the sources from which they receive data. As described above, the agent's wrappers tend to be relatively brittle, as the invariably rely on idiosyncratic formatting details observed during the learning process. Unfortunately, if the source modifies its formatting (for example, to "remodel" its user interface) then the observed regularities will no longer hold and the wrapper will fail. As a concrete example, Figure 2 and Figure 3 show the Altavista search engine, before and after a site redesign.

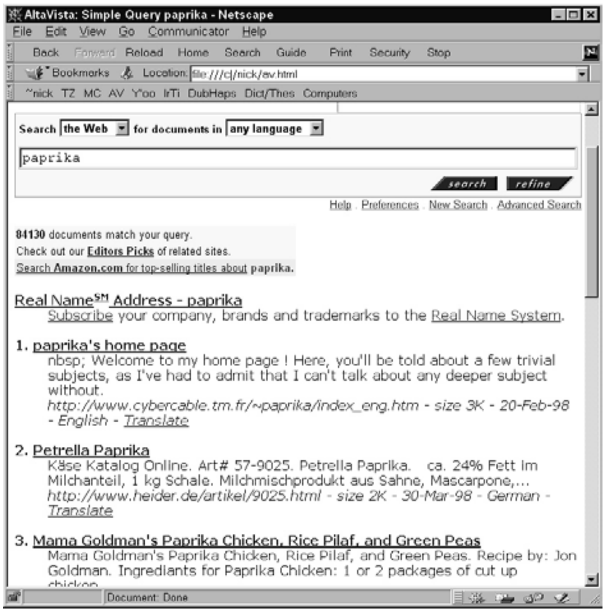

Fig. 2. Altavista snapshot before redesign

The two key challenges to wrapper maintenance are *wrapper verification* (determining whether the wrapper is still operating correctly), and *wrapper re-induction* (learning a revised wrapper). The second challenge is considerably more difficult, although even wrapper verification is non-trivial. The difficulty is that at most web sites, either the content to be extracted, or the formatting regularities, or both, may have changed, and the verification algorithm must distinguish the two. For example, suppose that the change in the Microsoft stock price is checked three times at a stock-quote server, and the extracted values are +3.10, -0.61 and <B><IMG src= advert.gif>. Intuitively our verification algorithm should realize that the relatively the first two values are "similar" and do not indicate trouble, but the third value is an outlier and probably indicates a defective wrapper.

Kushmerick [18,20] describes a simple and accurate algorithm for wrapper verification. The algorithm first learns a probabilistic model of the data extracted by the wrapper during a training period when it is known to be operating correctly. This model captures various properties of the training data such as the length or the fraction of numeric characters of the extracted data. To verify the wrapper after the training period, the extracted data is evaluated against the learned model to estimate the probability that wrapper is operating correctly. The algorithm is domain independent and is not tied to any particular wrapper class or learning algorithm, but rather treats the wrapper as a black-box and inspects only its output. The algorithm handles (acyclic) XML data, not just relational data, so it is applicable to all of the wrapper classes described above.

Wrapper re-induction has also received some attention. Lerman et al [23] learn a probabilistic model of the extracted data that is similar to (though substantially more expressive than) that used by Kushmerick. This more sensitive model enables wrapper re-induction as follows. After a wrapper is deemed to be broken, the learned model is used to identify probable target fragments in the (new and unannotated) documents. This training data is then post processed to (heuristically) remove noise, and the data is given to a wrapper induction algorithm. Lerman et al demonstrate empirically that this semi-supervised approach is highly accurate in many real-world extraction scenarios.

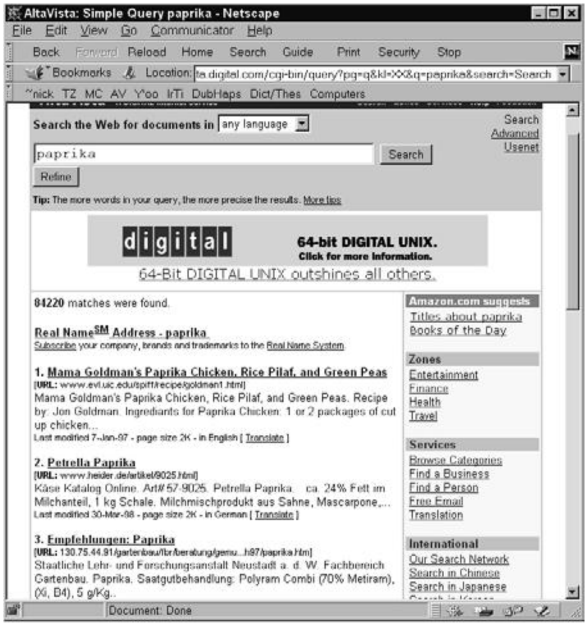

Fig. 3. Altavista snapshot after redesign

2.6 Post-processing Extracted Content

The work described so far is highly simplified in that the task is assumed to involve simply processing a given document to extract particular target fragments. However, in many extraction scenarios, the information to be extracted is actually distributed across multiple documents, or an attribute value is given only once on a page but is relevant to several extracted objects. For example, Figure 4 shows a simple scenario in which some attribute values are "re-used" across multiple extracted objects, and other values must be harvested from a collection of hyperlinked documents.

Some of these issues are handled by the wrapper classes defined earlier. For example, Muslea et al's embedded catalog formalism [27] permits an extracted fragment to be "shared" across multiple objects. Furthermore, the information extraction community has long investigated the issue of cross-document references. However, these approaches require considerable linguistic processing and are not applicable to the example shown in Figure 4.

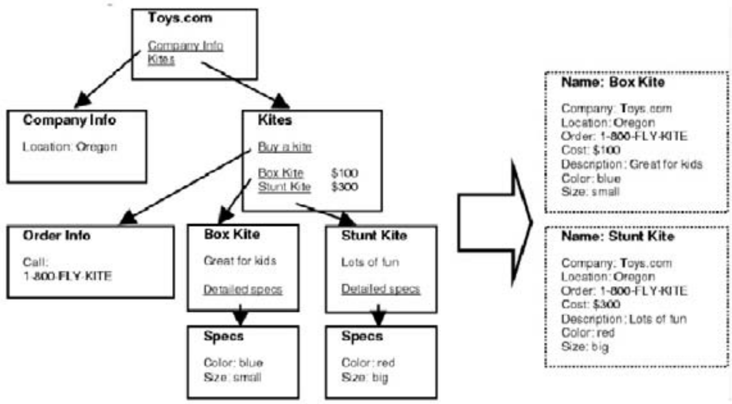

Fig. 4. A complicated extraction task in which attribute values are both distributed across multiple documents, and reused across objects. (Adapted from [15].)

Jensen and Cohen [15] address these problems by proposing a language for specifying how the extracted data should be post-processed. Rules express how the raw extracted data should be grouped into larger composite objects. Jensen and Cohen argue that their language is sufficiently expressive to handle the data extracted from 500 web sites exporting job and product advertisements. Furthermore, they suggest (though do not implement) an algorithm for automatically learning such rules from examples of grouped data.

2.7 Beyond Supervision

The key bottleneck with adaptive information extraction is obtaining the labeled training data. The use of machine learning is motivated by the fact that the cost of labeling documents is usually considerably less than the cost of writing the wrapper's extraction rules by hand. Nevertheless, labeling documents can require considerable domain expertise, and is generally tedious and error-prone. The approaches described so far simply assumes that an adequate training corpus exists, but considerable research effort has investigated so-called "active learning" methods for minimizing the amount of training data required to achieve a satisfactory level of generalization.

The basic idea of active learning is to start with a small amount of training data, run the learning algorithm, and then used the learned wrapper to predict which of the remaining unlabeled documents is most informative, in the sense of helping the learning system generalize most with the one additional training document. As a trivial example, if the corpus contains duplicate documents, then the learner should not suggest that the same document be annotated twice.

As one example of the use of active learning in the context of wrapper induction, consider Muslea et al [28]. The basic idea of this approach is that every information extraction task has a "dual", and correlations between the original task and its dual can help the system identify useful unlabeled documents.

Recall that Muslea et al's wrapper learning algorithm learns a sequence of landmarks for scanning from the beginning of the document to the start of a fragment to be extracted. An alternative way of finding the same position is to scan backwards from the end of the document for a (different!) set of landmarks. Muslea's active-learning extensions solves both learning tasks in parallel on the available training data. The two resulting wrappers are then applied to all the unlabeled documents. The system then asks the user to label one of the documents for which the two wrappers give different answers. Intuitively, if the two wrappers agree for a given unlabeled document, then the document is unlikely to be useful for subsequent learning.

Muslea et al demonstrate that the active-learning version of their algorithm requires significantly less training data to obtain the same level of generalization. For example, averaged across a variety of challenging extraction tasks, the error of the learned wrapper is about 50% less when the user annotates ten training documents chosen in this intelligent manner, compared to ten documents chosen randomly.

Brin [4] explores a different sort of extraction task, in which the user gives the system examples of some concept. For example, to learn to extract book title/author pairs, the user would supply a small sample of pairs, such as {(`Isaac Asimov`, `The Robots of Dawn`), (`Charles Dickens`, `Great Expectations`), ...}. The job of the extraction system is then to flesh out this list with as many additional instances as possible.

Brin's algorithm iteratively searches the Web for the seed pairs. When it finds a document that contains a pair, it learns an information extraction pattern for that particular pair, and then applies this pattern to the remainder of the page. The resulting extracted pairs are added to the seeds and the process iterates. There is no guarantee that this process will converge or even that the extracted pairs are correct. Nevertheless, preliminary experiments demonstrated promising results.

Finally, Crescenzi et al [9] focus on an even bigger challenge: wrapper learning without any supervision (labeled training data) at all. Consider the pages from some online bookstore that would be returned by two queries, for Dickens and for Asimov. In most cases, these pages would be formatted the same way, with the only difference being the content to be extracted. The intuition behind Crescenzi et al's approach is that a wrapper can be learned by comparing these two pages and finding similarities and differences. The similarities correspond to common formatting and structural elements; the differences correspond to data to be extracted. By repeatedly replacing the differences with wild-cards and noting repetitive structures, their algorithm can learn a wrapper that corresponds to a regular grammar, without the need for any manually labeled training data. Crescenzi et al report that their algorithm works well in a variety of real-world domains.

3 Relational Learning Techniques

In this section we introduce adaptive IE systems that use relational learning techniques. We present a short introduction to common relational rule induction algorithms and how they are used as a basis in several information extraction systems. We do not focus on the formal foundations of inductive logic programming [25,1]; our goal is to provide a summary of relational rule induction approaches as they have been used for adaptive IE over the past decade. Before introducing the basic concepts of rule induction let us give a short motivation for using relational techniques for learning wrappers. Several existing techniques for IE (like HMM's presented in Section 2) are based on the assumption to determine relevant text parts to be extracted by statistical means. These finite-state techniques can be seen as some sort of rule learning, ie the learning of production (grammar) rules constrained by certain probability measures. When we talk about rule learning in the context of relational rule learning we have logical rules in mind, in the sense of learning rules of first order predicate logic or at least subsets of first order rules like Horn rules or Prolog programs. As we will see, while the different rules learned by the various IE systems vary in their representation and signature, in general they can all be rewritten in a uniform predicate logic representation.

Talking about learning logical rules in combination with IE only makes sense if we abstract from the pure lexical representation of documents. Thus a first step to use relational learning techniques is to find a suitable document representation suited to the formal framework of logic, literals and logic rules. The most common representation used in relational IE is to interpret a document as a sequence of feature terms or tokens having several attributes describing features of grouped symbols from the document. How the tokenization is done is subject a) to the type of extractions needed, and b) the learning methods used. For example, the relevant features may be its type (integer, char, HTML tag), whether it is upper or lower case, its length, linguistic knowledge about the word category, its genus or even additional semantic knowledge drawn from a rich taxonomy.

For example Thomas [34] uses a document transformation into feature terms, in which a fragment like `<b>Pentium 90</b>` is written as a list of tokens: *[token(type=html, tag=b), token(type=word, txt='Pentium'), token(type=int, val=90), token(type=html_end, tag=b)]*. If we replace a feature value like *b* in *token(type=html, tag=b)* with a variable *token(type=html, tag=X)*, we can a) describe the class of all tokens of type *html*, and b) use unification methods to find all non-closing HTML tags in a tokenized document.

It then becomes obvious how more complex patterns can be defined by the use of first order predicate rules. For example, the rule *link(Description, Url) :- pos(P, token(type=html, tag=a, href=Url)), sequence(P, E, TokenSeq), not in(token(type=html_end, tag=a), TokenSeq), next(E, token(type=html_end, tag=a))* extracts tuples of the form *<Description, URL>* from a HTML document. For further details of how logic programs can be used for information extraction, see [35]. In the last decade various representations have been developed, some influenced largely by logic programming [16,34], and other slot-oriented approaches motivated by natural language processing. In essence they all can be represented without much effort in a first order predicate logic syntax.

Additional representations may be used to capture the documents layout. For example, a parse tree of the HTML structure can give information on paragraphs, titles, subsections, enumerations, tables. For information extraction tasks tailored for HTML or XML documents, the Document Object Model (DOM) can provide additional information into the learning and the later extraction process. Systems like that of Cohen [8] and the wrapper toolkit of the MIA system (Section 3.3) make use of such representations. In general the document representation can not be considered to be independent from the extraction task. For example, if someone wants to extract larger paragraphs from free natural language documents she probably will use a document representation and therefore relational representation reflecting larger document blocks, compared to someone interested in prices from online catalogues. Nevertheless if relational methods are to be used no matter which representation is chosen it must be presented in terms of relations.

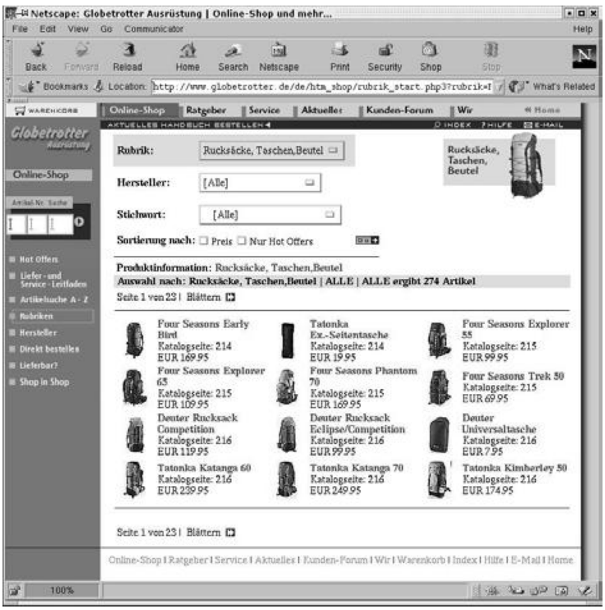

Fig. 5. An online catalogue

Section 2 already presented several state of the art adaptive IE algorithms, so what might be the shortcoming of these systems and what might be the advantages of relational rule based IE systems? One answer is that of human readability. A learned rule in the relational approach has a clear conclusion and conjunctive set of premises, which are all understandable because they refer to certain easily recognizable features of the document representation. This of course only holds if the document representation itself is clear and understandable. Thus such a learned rule will also give a clear explanation why it is used for extraction. Another strong argument is that of extending relational learning approaches. Having sets of first order rules in mind, it becomes apparent that adding additional background knowledge like ontologies or additional domain knowledge is easy to do. And even more important this additional knowledge can also be incorporated into the rule construction algorithm, and the inductive rule learning calculus can be extended by reasoning components from automated deduction systems. For example, some IE systems make use of additional semantic knowledge derived from a taxonomy [5,33].

By now we can think of relational learning as a core algorithm that expects a set of examples in the form of relation instances, some additional knowledge (the background theory) and a set of predicates from which rules can be build. The algorithm tries to construct rules using the background theory and these special predicates such that the new rules explain (cover) the presented positive examples and exclude (if given) the negative ones. Applying this approach to the tasks of IE involves the use of a appropriate relational document representation, text examples as grounded facts, and additional predicates plus a background theory used to check certain features of text fragments and tokens. In theory the core algorithm is not affected by the representation: by choosing the right representation, a standard relational rule induction algorithm can be used as core learning algorithm for adaptive IE. In practice it turns out that due to complexity issues modification and tailored approaches are needed, but the important point is the theoretical framework of learning logical rules [1] provides a well understood and formal basis for adaptive information extraction.

3.1 Rule Learning

Because IE involves extracting certain fragments from a document (where the key idea is that the extracted fragments are rule variable bindings), we will not discuss propositional rule learning. Since we adopt the general approach from Section 2 that extraction rules consist of delimiters and slots (extraction variables), we are confronted with the problem of inducing left and right delimiters. Additionally many approaches also try to induce some information about the extractions itself, by recognizing certain specific features from the provided example fragments. For example they generalize starting from certain linguistic or semantic features. So far almost all existing relational approaches are using one of the following techniques.

One-shot learning. Given a set of positive examples and perhaps negative examples plus additional information needed (eg the documents the examples are drawn from), these approaches try to learn a rule in one step. One shot learning approaches do no refinement and evaluation at each step during the rule building process, instead they assume their applied learning operators are good enough or they pass the evaluation and further refinement of learned rules to the user. The Autoslog [30] system and the T-Wrapper system [34] use one shot learning approaches.

Sequential covering. In contrast to One Shot Learning a sequential covering involves an iterative process of refinement and testing. The general Sequential Covering algorithm is shown in Figure 6. This algorithm learns a disjunctive set of rules depending on a threshold with regard to the performance of a learned rule. In other words, the learned rule set may still cover some negative examples. The algorithm repeatedly tries to learn one rule meeting the threshold condition. As long as the threshold is not met the example set for the next iteration is built by removing the positive and negative examples covered by the previously learned rule. Sequential Covering serves as a basis for many inductive algorithms. The crucial point of this algorithm is the function Learn-One-Rule. Clark and Niblet [7] provide a K-Beam Search based algorithm, which is by now one standard approach for learning one rule.

```
Sequential-Covering(Target_Attribute,Attributes,Examples,Threshold)
    Learned_Rules ← {}
    Rule ← Learn-One-Rule(Target_Attribute,Attributes,Examples)
    while Performance(Rule,Examples) > Threshold, do
        Learned_Rules ← Learned_Rules + Rule
        Examples ← Examples - {examples correctly classified by Rule}
        Rule ← Learn-One-Rule(Target_Attribute,Attributes,Examples)
    Learned_Rules ← sort Learned_Rules according to Performance over Examples
    return Learned_Rules
```

Fig. 6. Sequential covering algorithm. (Adapted from [24].)

```
FOIL(Target_Predicate, Predicates, Examples)
    Pos ← Examples for which Target_Predicate is true
    Neg ← Examples for which Target_Predicate is false
    Learned_Rules ← {}
    while Pos, do
        NewRule ← rule that predicts Target_Predicate with no preconditions
        NewRuleNeg ← Neg
            while NewRuleNeg, do
            Candidate_Literals ← generate new body literals for NewRule, based on Predicates
            Best_Literal ← argmax l ∈ Candidate_Literals FOIL_GAIN(l,NewRule)
            add Best_Literal to preconditions of NewRule
            NewRuleNeg ← subset of NewRuleNeg that satisfies NewRule preconditions
        Learned_Rules ← Learned_Rules + NewRule
        Pos ← Pos - {members of Pos covered by NewRules}
    return Learned_Rules
```

Fig. 7. The FOIL algorithm. (Adapted from [24].)

FOIL: Learning first order rules. Though the Sequential Covering algorithm builds the basis for many inductive rule learning algorithms it is in combination with CN2 a propositional rule learner. Nevertheless it builds a basis for many first order rule learning approaches. A widely used first order rule learning algorithm is the top-down procedure called FOIL [29]. Modified versions of FOIL are the basis for most adaptive IE systems using relational learning techniques. For example, the SRV system [10] uses a FOIL based core algorithm. FOIL tries to find a description of a target predicate, given a set of examples and some background knowledge. In general the background theory and examples are function-free ground facts, where the examples are positive and negative instances of the target predicate. Here negative instances means explicitly stating for which instantiations the target predicate shall not be true. FOIL uses the closed world assumption during rule learning: every instance not declared positive is assumed to be negative.

Using FOIL for wrapper induction. Assume we chose a document representation where a document D is mapped to a set of facts $T(D)$ of the form *word(Pos,Token)*, where *Pos* is the starting position of the text fragment described by the token *Token*. *Token* is a feature term in the previously discussed sense. Further we have a set of dedicated predicates, which FOIL uses for rule construction. In general these predicates are of two different types: one to test for certain token features and the other to set tokens into relation to each other. For example, *next(P1, P2)* describes all tokens at position *P1* and its direct successor token referred by *P2*. The reader can think of many different relational predicates like: *fragment(P1, Length, F)* or *near_by(P1,P2)*. Examples for predicates checking certain features may be *has_feature(type, html, Token)*, which checks if a token *Token* has the attribute *type* and if its value is *html*. An additional background theory provides the definition for these predicates. At this point the advantage of relational and first order rule learning becomes apparent. Suppose our document representation is modified so that it contains relational information in the sense of a document object model (DOM). Then to learn rules taking advantage of additional document layout features, we need only extend the background theory with additional predicates for traversal and retrieval of DOM nodes. Let us return to the FOIL algorithm. What is left to demonstrate FOIL are examples. Assuming we want to learn a rule *extract(X,Y)*. Note this is a multi slot rule which itself defines a problem class in the IE context. Because examples have to contain the target predicate our examples will be of the form *extract("Tatonka Kimberley 50","EUR 174.95")*.

Now that we know how to define positive examples let us see how to represent negative ones. In comparison to some finite state techniques the need for negative examples may be a shortcoming. While FOIL needs negative examples, we do not necessarily have to provide them by tedious annotating them by hand. Some assumptions can be taken, such that negative examples can be generated automatically. For example we can take a closed world assumption: negative examples are all those text tuples not explicitly stated to be positive ones. Besides the huge amount of possible tuples that can be generated, this has another serious shortcoming: if a document contains more relevant fragments than annotated, we will run into problems if we rely on the closed world assumption. So either we must exhaustively enumerate all relevant fragments of the document, or else we must use heuristics to construct negative examples automatically. Examples of such heuristics include text fragments consisting of positive examples but extended to the left and right or permutation of positive example arguments. Freitag [10] uses a heuristic that constructs negative examples from all those fragments with more, or fewer, tokens than the positive examples.

Let us start FOIL's top-down learning with the most general rule *extract(X,Y)* $\leftarrow$ *true*. As long as the current rule covers one of the negative examples—for example *extract("Tatonka","EUR")* (see Figure 5)—the body of the rule is extended by the best constraining literal. Most applications of FOIL in the context of IE use a similar function to that of Quinlan's *FOIL_GAIN* which characterizes the information contained in the ratio of positive and negative examples in a set of examples. The SRV system uses following gain function: $I(S) = -log_2(P(S)/(P(S) + N(S)))$, where $P(S)$ and $N(S)$ are the positive and negative number of examples of the example set S. $GAIN(A) = P(S_A)(I(S) - I(S_A))$ with S_A is the subset of S covered by the rule after adding the literal A to it. The computation of the best literal is thus very expensive depending on the document sets and the complexity of the predicates. The following rules illustrate some hypothetical intermediate steps during a possible rule induction process:

extract(X,Y) ← fragment(P1,2,X), fragment(P2,2,Y).
extract(X,Y) ← fragment(P1,2,X), near_by(P1,P2,Y), fragment(P2,2,Y).
extract(X,Y) ← word(P0,Token1), has_feature(type,html,Token1),
has_feature(color,blue,Token1), next(P0,P1), fragment(P1,2,X),
near_by(P1,P2,Y), fragment(P2,2,Y), in(Token2,Y),
has_feature(type,float,Token2).

3.2 Relational Learning Techniques in Practice

In the following we will introduce some adaptive IE systems that use relational learning techniques or are very strongly connected to it.

One-shot learning. AutoSlog [30] learns a set of extraction patterns by specializing a set of general syntactic patterns. Such patterns are called concept nodes and consist of certain attributes like rule trigger, constraints, constant slots or enabling condition.

AutoSlog needs semantic pre-processing which involves a user to tag the example documents with certain semantic labels. The system expects a part of speech (POS) or linguistically tagged input document. The key idea is to figure out a *trigger* word and matching predefined linguistic patterns. The algorithm then tries to modify these patterns by several heuristics and the use of a taxonomy. The AutoSlog algorithm does not contain any inductive step, it is left to the user to accept or reject a learned rule. It is mentioned in the context of relational learning techniques, because concept nodes may be easily rewritten in the form of logic rules and thus specializing among those is similar to relational learning.

Fig. 8. The T-Wrapper web interface

Another one shot learning system by Thomas is T-Wrappers [34]. Similar to Kushmerick he defines several wrapper classes assuming that the essential part for wrapper construction is it to learn left and right delimiters, which he calls anchors. Thomas describes wrapper in a Prolog-like language [13]. His approach can be summarized into three steps: 1) for each argument of each example tuple collect the left and right text fragments wrt. to a given length 2) generalize on all left and on all right fragments, such that for a 3-tuple for example

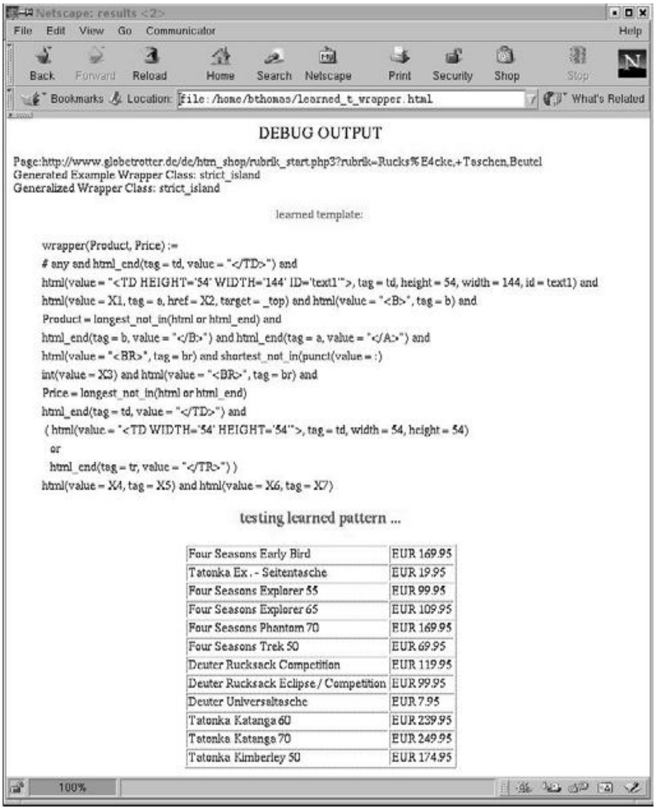

Fig. 9. Learned T-Wrapper rule and extractions.

this generalization process results in 6 anchor patterns. 3) depending on the wrapper class [13] use a pre-defined structural rule layout where the learned anchors are inserted. Thomas uses a feature term representation for text fragments and anti-unification (LGG) based techniques for generalization. In contrast to Riloff he uses a bottom-up strategy, starting with the surrounding fragments (the most specific ones) of each argument of each example. The T-Wrapper system learns from positive examples only, does not use any linguistic knowledge (though this can be easily encoded into the feature term representation) and it learns multi slot extraction rules. Only a handful of examples is needed to produce wrappers for HTML documents without the need of post processing the learned wrapper. Figure 9 shows such a learned rule and extractions from the web page shown in Figure 5.

FOIL-based systems: Top-down learning. One of the most successful ILP and top-down based learning system used for IE is the SRV system by Freitag [10]. SRV strongly follows the idea of the standard FOIL algorithm as described in Section 3.1. The system is capable of learning single slot wrappers from natural and HTML texts. It also follows the key idea in learning left and right delimiters. Freitag also extended SRV's feature predicates and document representation by linguistic information. Surprisingly this had little effect on the performance of SRV. Though Freitag uses a standard FOIL algorithm his representation of rules does not follow the strict concept of first order rules or standard Prolog rules. Junker et al [16] illustrates how to use a top-down sequential covering algorithm to learn Prolog rules. Junker et al focuses on single slot extraction and text documents. Their algorithm can be seen as a special ILP learner based on a set of operators used for rule refinement. Each of these operators can be understood as transformations or inference rules, replacing or introducing new or modified literals to the body of a rule.

Bottom-up learning. Bottom-up algorithms start with the most specific rules or example instances and stepwise generalize these rules until a certain stop criterion is reached (eg none of the negative examples are covered and most of the positive ones). Soderland presents the Crystal system [33], which uses a bottom-up sequential covering algorithm. Crystal and the later implemented system WebFoot (for HTML documents) use two different generalization operations: a unification-based operation for syntactic generalization, and a second based on knowledge drawn from a taxonomy to generalize on the slot fillers. For example, assume a semantic constraint allows as slot fillers instances of the class $< PERSON >$ and a word *he* is an instance of class $< GENERICPERSON >$ which is a subclass of $< PERSON >$. Then the semantic constraint for the word *he* is met because of the implication. Furthermore in Crystal whole sentences are linguistically pre-processed and annotated to serve as examples. The system is able to learn multi slot extraction rules given positive and negative examples.

Hybrid relational learning techniques. A serious problem using top-down learning algorithms for IE is to guide the search for good body literals. As long as the search space can be kept small, a pure non-biased top-down approach may be tractable for IE. But in general the great number of negative examples and the use of more sophisticated predicates (eg background theory) for rule refinement blows up the search space. Thus a standard top-down algorithm that exhaustively checks all possible rule refinements is infeasible. On the other hand using a standard bottom-up algorithm often leads to overly specific rules with very low recall.

The Rapier [5] system combines top-down and bottom-up techniques to learn single slot extraction rules from POS-tagged documents. In fact it uses a similar approach adapted from the ILP system CHILLIN to learn three patterns: pre-filler patterns (left delimiter), post-filler-pattern (right delimiter), and a patterns for the slot-filler itself. Like most of the other systems it uses a linguistic pre-processing step to annotate the document with part of speech information. For the generalization step Rapier uses a modified LGG operator, that provides disjunctions of two patterns to be generalized. Rapier begins by generalizing two filler patterns. This generalized pattern is used to initiate a top-down learning step. Elements to be added to the rule are created by generalizing the appropriate portions of the pre-fillers or post-fillers of the pair of rules from which the new rule is generalized. The LGG operator is also modified such that it uses an additional semantic hierarchy to derive super-classes covering instances in the initial constraints.

Another system combining several learning techniques is that of Ciravegna [6] called Pinocchio. He uses a sequential covering algorithm. The system needs as input a col-

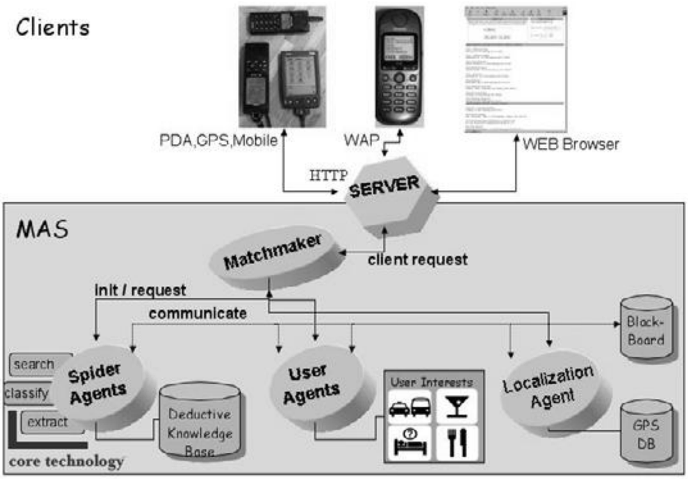

Fig. 10. Architecture of the MIA system.

lection of texts pre-processed with a POS tagger. Pinocchio is unique in that it does not try to learn an extraction rule for a complete slot. Instead it learns rules to recognize the left delimiter independently from the right one. The algorithm can be separated into three steps: 1) A bottom-up based rule construction from tag surrounding text fragments, where the k best rules for each instance are retained in a best rule pool. 2) To raise the overall recall rate, some rules not contained in the best rule pool are considered for further refinement. The idea is to find a set of rules that are reliable at least in restricted areas. Ciravegna illustrates a method which he calls contextual tagging. 3) Finally, correction rules are learned. These rules shift wrongly positioned tags to their correct position. They are learned with the same algorithm like the tagging rules but also match the tags produced during the previous steps. Pinocchio can be used to extract multiple slots. Results presented by Ciravegna are very promising.

3.3 Application Example

As shown in Figure 10, *MIA* [2] is a multi-agent based information system focusing on the retrieval of short and precise facts from the web. Making the immense amount of information on the web available for *ubiquitous computing* in daily life is a great challenge. Besides hardware issues for wireless ubiquitous computing, like wireless communication, blue-tooth technologies, wearable computing units, integration of GPS, PDA and telecommunication devices, one major problem is that of intelligent information extraction from the WWW.

Instead of overwhelming the mobile user with documents found on the web, the MIA system offers the user a short precise piece of information she is really interested in with fast query response times. *MIA* monitors the position of the mobile users and autonomously updates the subject of search whenever necessary. Changes may occur when the user travels to a different location, or when she changes her search interests.

Currently MIA supports three different user types: stationary Web Browser, mobile phone with WAP support and PDA with GPS device. The search domains are freely configurable by the user. So far *MIA* is capable of automatic (multi slot) address extraction; extraction of time-tables and event descriptions is also planned.

Using pre-learned wrappers. One heuristic used in the MIA system is that to use a set of pre-selected web sites as entry points for the retrieval. This assures to the user that at least some results are presented. While some known information resources are used, the problem remains of extracting this information. The MIA administrator uses MIA's wrapper toolkit to learn wrappers for certain domains. Whenever the extraction agents visit one of these domains during their search they use these pre-learned wrappers to extract information from one of the web pages. Currently the wrapper toolkit uses a one shot learning strategy [34] extended with a special document representation. Instead of assuming the document to be a linear sequence of tokens the DOM model of the document is used. The general strategy to learn left and right anchors (delimiters) is kept, but extended to path learning in the DOM of the document. Additionally the general intention is now to learn one wrapper for a whole document class (eg found at one web domain) instead learning one wrapper for one document. This is in contrast to [34] and the following method used by MIA.

Learning wrappers during search and retrieval. The major problem someone is confronted with in the context of an autonomous multiagent system like *MIA* is the lack of available examples for learning wrappers online. Because *MIA*'s web page classifier provides unknown web pages to the system, we can not assume the existence of the training data needed for learning. On the one hand, the classifier is good enough to determine if an address is contained on a web page; on the other hand, arbitrary web pages vary too much for a single general purpose address wrapper to be effective. While the use of a large address database and a name recognizer (POS-tagger or named-entity recognizer) might work better for this particular problem, but MIA is aiming at a generic approach with no hard-wired solutions.

To overcome this problem MIA uses a learning algorithm that derives its examples by means of *knowledge representation* (*KR*) techniques. That is, we model our knowledge about addresses with a logic KR language in advance and are able to query this knowledge base to derive example patterns. These example patterns can then be used to construct examples as input to a modified learner. This allows MIA to learn wrappers even for unknown pages. This approach called *learning from meta examples* shows very promising results for the automatic construction of address wrappers (Figure 11). Various experiments showed that the knowledge base can also be used to verify the extracted information on a still limited level, but nevertheless this idea can serve as basis for some kind of self-supervision for autonomous information extraction agents.

4 Summary

Information extraction is a core enabling technology for a wide variety of information-gathering and -management agents. The central challenge to information extraction is the ability to scale with the number and variety of information sources. We have

Test Setting (meta examples) pages: 461 tuples: 3158 KB: 5 rules, 6 patterns						
wrapper class	pages cov.	pos	neg	recall	precision	coverage
semi, no html	319	785	85	0,69	0,9	0,27
weak, no html	382	1712	904	0,83	0,65	0,54
semi, look ahead	339	895	156	0,73	0,85	0,28

self-supervision:
semi: extractions must be matched by derived pattern from KB
weak: extractions must be matched by generalized patterns from KB
wrapper wlass
no html: extraction are not allowed to contain HTML
look ahead: extractions are not allowed to contain tokens of right delimiter

Fig. 11. Performance of MIA's IE component.

described a variety of adaptive information extraction approaches that use machine learning techniques to automatically learn extraction rules or knowledge from training data.

Due to the highly practical nature of the IE task, all of the approaches described in this chapter have been tested on various real world examples. That means the gap between pure research and practical usage in agent systems is smaller than it might seem at first glance. For example, the IE component of the MIA multi-agent system described in Section 3.3 is based directly on techniques described in this chapter.

We have segmented the field of adaptive information extraction roughly into two areas: *finite state* techniques that learn extraction knowledge corresponding to regular grammars or automata, and the *relational rule learning* techniques that learn first-order Prolog-like extraction rules. In addition to the core adaptive IE techniques, we also briefly discussed several issues related to the entire information extraction "lifecycle".

The finite-state approaches are generally simpler, and their learning algorithms are generally faster and require fewer training examples. On the other hand, relational representations are substantially more expressive, which can be crucial for natural language domains such as newspaper articles that exhibit substantial variability. Adaptive information extraction thus exhibits a familiar expressiveness–complexity tradeoff.

Perhaps the most fundamental open issue in adaptive information extraction is a method for determining which technique is best suited to any particular extraction task. Today, this complicated judgment requires considerable expertise and experimentation. Ultimately, we foresee a semi-automated methodology in which the heterogeneity of the documents could be measured in various dimensions, in order to predict the simplest approach that will deliver satisfactory performance.

Acknowledgments. Kushmerick was supported by grant N00014-00-1-0021 from the US Office of Naval Research, and grant SFI/01/F.1/C015 from Science Foundation Ireland. Thomas and the MIA project was supported by grant from the state of Rhineland-Palatinate, Germany.

References

1. F. Bergadano and D. Gunetti. *Inductive Logic Programming*. MIT Press, 1996.
2. G. Beuster, B. Thomas, and C. Wolff. MIA – A Ubiquitous Multi-Agent Web Information System. In *Proceedings of International ICSC Symposium on Multi-Agents and Mobile Agents in Virtual Organizations and E-Commerce (MAMA'2000)*, December 2000.
3. D. Bikel, S. Miller, R. Schwartz, and R. Weischedel. Nymble: A high-performance learning name-finder. In *Proc. Conf. on Applied Natural Language Processing*, 1997.
4. S. Brin. Extracting patterns and relations from the World Wide Web. In *Proc. SIGMOD Workshop on Databases and the Web*, 1998.
5. M. E. Califf. *Relational Learning Techniques for Natural Language Information Extraction*. PhD thesis, University of Texas at Austin, August 1998.
6. F. Ciravegna. Learning to Tag for Information Extraction from Text. In *Workshop Machine Learning for Information Extraction, European Conference on Artifical Intelligence ECCAI*, August 2000. Berlin, Germany.
7. P. Clark and T. Niblett. The CN2 induction algorithm. *Machine Learning*, 3:261–283, 1989.
8. W. Cohen and L. Jensen. A structured wrapper induction system for extracting information from semi-structured documents.
9. V. Crescenzi, G. Mecca, and P. Merialdo. Roadrunner: Towards automatic data extraction from large web sites. In *The VLDB Journal*, pages 109–118, 2001.
10. D. Freitag. *Machine Learning for Information Extraction in Informal Domains*. PhD thesis, Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, November 1998.
11. D. Freitag and N. Kushmerick. Boosted Wrapper Induction. In *Proceedings of the Seventh National Conference on Artificial*, pages 577–583, July 30 - August 3 2000. Austin, Texas.
12. D. Freitag and A. McCallum. Information Extraction with HMM structures learned by stochastic optimization. In *Proceedings of the Seventh National Conference on Artificial*, July 30 - August 3 2000. Austin, Texas.
13. G. Grieser, K. P. Jantke, S. Lange, and B. Thomas. A Unifying Approach to HTML Wrapper Representation and Learning. In *Proceedings of the Third International Conference on Discovery Science*, December 2000. Kyoto, Japan.
14. C. Hsu and M. Dung. Generating finite-state transducers for semistructured data extraction from the web. *J. Information Systems*, 23(8):521–538, 1998.
15. L. Jensen and W. Cohen. Grouping extracted fields. In *Proc. IJCAI-01 Workshop on Adaptive Text Extraction and Mining*, 2001.
16. M. Junker, M. Sintek, and M. Rinck. Learning for Text Categorization and Information Extraction with ILP. In *Proc. Workshop on Learning Language in Logic*, June 1999. Bled, Slovenia.
17. N. Kushmerick. *Wrapper Induction for Information Extraction*. PhD thesis, University of Washington, 1997.
18. N. Kushmerick. Regression testing for wrapper maintenance. In *Proc. National Conference on Artificial Intelligence*, pages 74–79, 1999.
19. N. Kushmerick. Wrapper induction: Efficiency and expressiveness. *Artificial Intelligence*, 118(1–2):15–68, 2000.
20. N. Kushmerick. Wrapper verification. *World Wide Web Journal*, 3(2):79–94, 2000.
21. N. Kushmerick, D. S. Weld, and R. Doorenbos. Wrapper Induction for Information Extraction. In M. E. Pollack, editor, *Fifteenth International Joint Conference on Artificial Intelligence*, volume 1, pages 729–735, August 1997. Japan.
22. T. Leek. Information extraction using hidden Markov models. Master's thesis, University of California, San Diego, 1997.

23. K. Lerman and S. Minton. Learning the common structure of data. In *Proc. National Conference on Artificial Intelligence*, 2000.
24. T. M. Mitchell. *Machine Learning*. McGraw-Hill, 1997.
25. S. Muggleton and L. D. Raedt. Inductive logic programming: Theory and methods. *Journal of Logic Programming*, 19(20):629–679, 1994.
26. I. Muslea. Extraction patterns for information extraction tasks: A survey. In *Proc. AAAI-99 Workshop on Machine Learning for Information Extraction*, 1999.
27. I. Muslea, S. Minton, and C. Knoblock. A hierarchical approach to wrapper induction. In *Proc. Third International Conference on Autonomous Agents*, pages 190–197, 1999.
28. I. Muslea, S. Minton, and C. Knoblock. Selective sampling with redundant views. In *Proc. National Conference on Artificial Intelligence*, 2000.
29. J. R. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5:239–266, 1990.
30. E. M. Riloff. *Information Extraction as a Basis for Portable Text Classification Systems*. PhD thesis, University of Massachusetts Amherst, 1994.
31. K. Seymore, A. McCallum, and R. Rosenfeld. Learning hidden Markov model structure for information extraction. In *Proc. AAAI-99 Workshop on Machine Learning for Information Extraction*, 1999.
32. S. Soderland. Learning information extraction rules for semi-structured and free text. *Machine Learning*, 34(1-3):233–272, 1999.
33. S. G. Soderland. *Learning Text Analysis Rules for Domain-Specific Natural Language Processing*. PhD thesis, University of Massachusetts Amherst, 1997.
34. B. Thomas. Anti-Unification Based Learning of T-Wrappers for Information Extraction. In *Proc. AAAI-99 Workshop on Machine Learning for Information Extraction*, 1999.
35. B. Thomas. Token-Templates and Logic Programs for Intelligent Web Search. *Intelligent Information Systems*, 14(2/3):241–261, March-June 2000. Special Issue: Methodologies for Intelligent Information Systems.

Agent-Based Distributed Data Mining: The KDEC Scheme

Matthias Klusch[1], Stefano Lodi[2,3], and Gianluca Moro[2,4]

[1] Deduction and Multiagent Systems
German Research Centre for Artificial Intelligence
Stuhlsatzenhausweg 3, 66123 Saarbruecken, Germany
klusch@dfki.de
[2] Department of Electronics, Computer Science and Systems, University of Bologna
[3] CSITE-CNR, Viale Risorgimento, 2, I-40136 Bologna, Italy
[4] Via Rasi e Spinelli, 176, I-47023 Cesena (FC), Italy
{slodi,gmoro}@deis.unibo.it

Abstract. One key aspect of exploiting the huge amount of autonomous and heterogeneous data sources in the Internet is not only how to retrieve, collect and integrate relevant information but to discover previously unknown, implicit and valuable knowledge. In recent years several approaches to distributed data mining and knowledge discovery have been developed, but only a few of them make use of intelligent agents. This paper is intended to argue for the potential added value of using agent technology in the domain of knowledge discovery. We briefly review and classify existing approaches to agent-based distributed data mining, propose a novel approach to distributed data clustering based on density estimation, and discuss issues of its agent-oriented implementation.

1 Introduction

Mining information and knowledge from huge data sources such as weather databases[1], financial data portals[2], or emerging disease information systems[3] has been recognized by industrial companies as an important area with an opportunity of major revenues from applications such as business data warehousing, process control, and personalised on-line customer services over the Internet and Web. *Knowledge discovery* (KD) is a process aiming at the extraction of previously unknown and implicit knowledge out of large databases which may potentially be of added value for some given application [1]. Among the steps of the overall KD process including preparation of the data to be analysed as well as evaluation and visualisation of the discovered knowledge, *data mining* (DM) which is devoted to automated extraction of unknown patterns from given data is a central element. The large variety of DM techniques which have been developed over the past decade includes methods for pattern-based similarity search,

[1] http://www.noaa.gov
[2] http://www.nasdaq.com
[3] http://www.cdc.gov

M. Klusch et al. (Eds.): Intelligent Information Agents, LNAI 2586, pp. 104–122, 2003.

clustering analysis, decision-tree based classification, generalization taking the data cube or attribute-oriented induction approach, and mining of association rules [2]

Most of the existing DM techniques were originally developed for centralized data and need to be modified for handling the distributed case. The increasing demand to scale up to massive data sets inherently distributed over a network with limited bandwidth and computational resources available motivated the development of methods for parallel (PKD) and distributed knowledge discovery (DKD) [3]. The related pattern extraction problem in DKD is referred to as *distributed data mining* (DDM). DDM is expected to perform partial analysis of data at individual sites and then to send the outcome as partial result to other sites where it is sometimes required to be aggregated to the global result. In fact, quite a number of DDM solutions are available using various techniques such as distributed association rules, distributed clustering, Bayesian learning, classification (regression), and compression, but only a few of them make use of intelligent agents at all.

One of the most widely used approach to DDM in business applications is to apply DM techniques to data which has been retrieved from different sources and stored in a central data warehouse. A *data warehouse* is a collection of integrated data from distributed data sources in a single repository [4]. However, despite its commercial success, the centralised application of data warehousing technology for DDM may be impractical or even impossible for some business settings in distributed environments. The main problems any approach to DDM is challenged to cope with concern issues of autonomy, privacy, and scalability. For example, when data can be viewed at the data warehouse from many different perspectives and at different levels of abstraction, it may threaten the goal of protecting individual data and guarding against invasion of privacy. Requirements to respect strict or a certain degree of autonomy of given data sources as well as privacy restrictions on individual data may make monolithic DM infeasible. Though DDM techniques are commonly considered as an important step towards reconciliation the opposing privacy concerns of centralised DM, that is protecting individual data and reducing unwanted privacy intrusions, research in the domain is still far from having resolved this problem of secure DDM in general.

Another problem arises with the need to scale up to massive data sets which are distributed over a large number of sites. For example, the NASA Earth Observing System[4] (EOS) is a data collector for satellites producing 1450 data sets of about 350GB per day and pair of satellites at a very high rate which are stored and managed by different systems geographically located all over the USA. Any online mining of such huge and distributed data sets in a central data warehouses may be prohibitively expensive in terms of costs of both communication and computation. To date, most work on DDM and PDM use distributed processing and the decomposability of data mining problems to scale up to large data sources. One lesson from the recent research work on DDM is that cooper-

[4] http://eospso.gsfc.nasa.gov/

ation among distributed DM processes may allow effective mining even without centralised control [5].

This in turn leads us to the question whether there is any real added value of using concepts from agent technology [6,7] for the development of advanced DDM systems. In general, the inherent feature of software agents of being autonomous, capable of adaptive and deliberative reasoning seems to fit quite well with the requirements of coping with the above mentioned problems and challenges of DDM. Autonomous data mining agents as a special kind of information agents [6] may perform various kinds of mining operations on behalf of its user(s) or in collaboration with other agents. Systems of cooperative information agents for data mining tasks in distributed, hetergoeneous and massive data environments appear to be quite a natural vision for the near future to be realised.

In this paper we briefly review and classify existing DDM systems and frameworks according to some criteria in section 2. This is followed by a brief discussion on the benefits of using agents for DDM in section 3, and the proposal of a novel scheme, named KDEC, for agent-based distributed data clustering in sections 4 and 5. We compare our approach to other related work in section 6, and conclude the paper in section 7 with an outline of ongoing and future research work.

2 State of the Art

In this section we provide a brief review of the most prominent and representative agent-based DDM systems to date, that are BODHI, PADMA, JAM, and Papyrus, according to (a) the kind, type, and used means for security of data processed; (b) used DM techniques, implementation of the system and agents; and (c) the architecture with respect to the main coordination and control, execution of data processing, and transmission of agents, data, and models in due course of the DM tasks to be pursued by the system.

BODHI [3] has been designed according to a framework for collective DM tasks on heterogeneous data sites such as supervised inductive distributed function learning and regression. This framework guarantees correct local and global data model with low network communication load. BODHI is implemented in Java; it offers message exchange and runtime environments (agent stations) for the execution of mobile agents at each local site. The mining process is distributed to the local agent stations and agents that are moving between them on demand each carrying its state, data and knowledge. A central facilitator agent is responsible for initializing and coordinating DM tasks to be pursued within the system by the agents and agent stations, as well as the communication and control flow between the agents.

PADMA [8] deals with the problem of DDM from homogeneous data sites. Partial data cluster models are first computed by stationary agents locally at different sites. All local models are collected to a central site that peforms a second-level clustering algorithm to generate the global cluster model. Individual agents perform hierarchical clustering in text document classification, and web based information visualization.

JAM [9] is a Java-based multi-agent system designed to be used for meta-learning DDM. Different learning classifiers such as Ripper, CART, ID3, C4.5, Bayes, and WEPBLS can be executed on heterogeneous (relational) databases by any JAM agent that is either residing on one site or is being imported from other peer sites in the system. Each site agent builds a classification model and different agents build classifiers using different techniques. JAM also provides a set of meta-learning agents for combining multiple models learnt at different sites into a meta-classifier that in many cases improves the overall predictive accuracy. Once the combined classifiers are computed, the central JAM system coordinates the execution of these modules to classify data sets of interest at all data sites simultaneously and independently.

Papyrus [10] is a Java-based system addressing wide-area DDM over clusters of hetereogeneous data sites and meta-clusters. It supports different task and predictive model strategies including C4.5. Mobile DM agents move data, intermediate results, and models between clusters to perform all computation locally and reduce network load, or from local sites to a central root which produces the final result. Each cluster has one distinguished node which acts as its cluster access and control point for the agents. Coordination of the overall clustering task is either done by a central root site or distributed to the (peer-to-peer) network of cluster access points. Papyrus supports various methods for combining and exchanging the locally mined predictive models and metadata required to describe them by using a special markup language.

Common to all approaches is that they aim at integrating the knowledge which is discovered out of data at different geographically distributed network sites with a minimum amount of network communication, and maximum of local computation. Agent-based DDM solutions have been applied to both heterogeneous and homogeneous data sites in both multi-database and distributed database environments

3 Why Agents for DDM?

Looking at the state of the art of agent-based DDM systems presented in the previous section we may identify following arguments in favor or against the use of intelligent agents for distributed data mining.

Autonomy of data sources. A DM agent may be considered as a modular extension of a data management system to deliberatively handle the access to the underlying data source in accordance with given constraints on the required autonomy of the system, data and model. This is in full compliance with the paradigm of cooperative information systems [11].

Interactive DDM. Pro-actively assisting agents may drastically limit the amount a human user has to supervise and interfere with the running data mining process [12]. For example, DM agents may anticipate the individual limits of the potentially large search space and proper intermediate results particularly driven by their individual users' preferences with respect to the particular type of DM task at hand.

Dynamic selection of sources and data gathering. One challenge for intelligent DM agents acting in open distributed data environments in which, for example, the DM tasks to pursue, the availability of data sites and their content may change at any time, is to discover and select relevant sources. In such settings DM agents may be applied to adaptively select data sources according to given criterias such as the expected amount, type and quality of data at the considered source, actual network and DM server load [13]. Such DM agents may be used, for example, to dynamically control and manage the process of data gathering to support any OLAP (online analytical processing) and business data warehouse application.

Scalability of DM to massive distributed data. One option to reduce network and DM application server load may be to let DM agents migrate to each of the local data sites in a DDM system on which they may perform mining tasks locally, and then either return with or send relevant pre-selected data to their originating server for further processing. Experiments in using mobile information filtering agents in distributed data environments are encouraging [14].

Multi-strategy DDM. For some complex application settings an appropriate combination of multiple data mining technique may be more beneficial than applying just one particular one. DM agents may learn in due course of their deliberative actions which one to choose depending on the type of data retrieved from different sites and mining tasks to be pursued. The learning of multi-strategy selection of DM methods is similar to the adaptive selection of coordination strategies in a multi-agent system as proposed, for example, in [15].

Collaborative DM. DM agents may operate independently on data they have gathered at local sites, and then combine their respective models. Or they may agree to share potential knowledge as it is discovered, in order to benefit from the additional opinions of other DM agents. Meta-learning techniques may be used to perform mining homogeneous, distributed data. However, naive approaches to local data analysis may produce an ambiguous and incorrect global data model if different hetereogeneous data sites are involved which store data for different sets of features, possibly with some common features among the sites. Collaborative DM agents may negotiate among each other and jointly plan a solution for the above mentioned problems at hand. The need for DM agents to collaborate is prominent, for example, in cases where credit card frauds have to be detected by scanning, analysing, and partially integrating worldwidely distributed data records in different, autonomous sources. Other applications of potential added value include the pro-active re-collection of geographically distributed patient records and mining of the corresponding data space on demand to infer implicit knowledge to support an advanced treatment of patients no matter into which and how many hospitals they have been taken into in the past. However, frameworks for agent-based collective data mining such as BODHI are still more than rare to date.

Security and trustworthiness. In fact, this may be an argument against the use of agents for DDM. Of course, any agent-based DDM system has to cope with the problem of ensuring data security and privacy. However, any failure to implement least privilege at a data source, that means endowing subjects with only enough permissions to discharge their duties, could give any mining agent unsolicited access to sensitive data. Moreover, any mining operation performed by agents of a DDM system lacking a sound security architecture could be subject to eavesdropping, data tampering, or denial of service attacks. Agent code and data integrity is a crucial issue in secure DDM: Subverting or hijacking a DM agent places a trusted piece of (mobile) software - thus any sensitive data carried or transmitted by the agent - under the control of an intruder. In cases were DM agents are even allowed to migrate to remote computing environments of the distributed data sites of the DDM system methods to ensure confidentiality and integrity of a mobile agent have to be applied. Regarding agent availability there is certainly no way to prevent malicious hosts from simply blocking or destroying the temporarily residing DM agents but selective replication in a fault tolerant DDM agent architecture may help. In addition, data integration or aggregation in a DDM process introduces concern regarding inference attacks as a potential security threat. Data mining agents may infer sensitive information even from partial integration to a certain extent and with some probability. This problem, known as the so called inference problem, occurs especially in settings where agents may access data sources across trust boundaries which enable them to integrate implicit knowledge from different sources using commonly held rules of thumb. Not any of the existing DDM systems, agent-based or not, is capable of coping with this inference problem in the domain of secure DDM.

In the following sections we are investigating how agents may be used to perform a special kind of distributed data mining, that is clustering of data at different homogeneous data sites. For this purpose, we are presenting an approach to cluster analysis based on density estimation, adopting it to the distributed case, and then briefly discuss issues of implementing the resulting scheme for distributed data clustering in an agent-based DDM system.

4 Data Clustering

4.1 The Cluster Analysis Problem

Cluster analysis is a a descriptive data mining task which aims at decomposing or partitioning a usually multivariate data set into groups such that the data objects in one group are similar to each other and are different as possible from those in other groups. Clustering techniques inherently hinge on the notion of distance between data objects to be grouped, and all we need to know is the set of interobject distances but not the values of any of the data object variables. Several techniques for data clustering are available but must be matched by the developer to the objectives of the considered clustering task [16]. In partition-based clustering, for example, the task is to partition a given data set into multiple disjoint sets of data objects such that the objects within each set are as homogeneous as possible. Homogeneity here is captured by an appropriate cluster scoring function. Another option bases on the intuition that homogeneity is expected to be high in densely populated regions of the given data set. Consequently, searching for clusters may be reduced to searching for dense regions of the data space which are more likely to be populated by data objects. That leads us to the approach of density estimation based clustering.

4.2 Density Estimation Based Clustering

In *density estimation* (DE) based clustering the search for densely populated regions is accomplished by estimating a so-called probability density or cumulative distribution function from which the given data set is assumed to have arisen. Many techniques for DE-based clustering are available from the vast KDD literature [17,18,19,20] and statistics [21]. In both areas, the proposed clustering methods require the computation of a non-parametric estimation of the density function from the data. One important family of non-parametric estimates is known as *kernel estimators.* The idea is to estimate a density function by defining the density at any data object as being proportional to a weighted sum of all objects in the data set, where the weights are defined by an appropriately chosen kernel function. In the following we introduce our approach to kernel-based density estimation.

Let us assume a set $S = \{\boldsymbol{x}_i \mid i = 1, \ldots, N\} \subseteq \mathbb{R}^n$ of data points or objects. Kernel estimators originate from the intuition that the higher the number of neighbouring data objects $\boldsymbol{x}_i$ of some given object $\boldsymbol{x} \in \mathbb{R}^n$, the higher the density at this object $\boldsymbol{x}$. However, there can be many ways of capturing and weighting the influence of data objects. When given the distance between one data object $\boldsymbol{x}$ and another $\boldsymbol{x}_i$ as an argument, the influence of $\boldsymbol{x}_i$ may be quantified by using a so called kernel function. A *kernel function* $K(x)$ is a real-valued, non-negative function on $\mathbb{R}$ which has finite integral over $\mathbb{R}$. When computing a kernel-based density estimation of the data set S, any element $\boldsymbol{x}_i$ in S is regarded as to exert more influence on some $\boldsymbol{x} \in \mathbb{R}^n$ than elements which are farther from $\boldsymbol{x}$ than the element. Accordingly, kernel functions are often non-increasing with $|x|$. Prominent examples of kernel functions are the square pulse function $\frac{1}{4}(sign(x+1) - sign(x-1))$, and the Gaussian function $\frac{1}{\sqrt{2\pi}} \exp(-\frac{1}{2}x^2)$.

A *kernel-based density estimate* $\hat{\varphi}_{K,h}[S](\cdot) \colon \mathbb{R}^n \to \mathbb{R}_+$ is defined, modulo a normalization factor, as the sum over all data objects $\boldsymbol{x}_i$ in S of the distances between $\boldsymbol{x}_i$ and $\boldsymbol{x}$, scaled by a factor h, called *window width*, and weighted by the kernel function K:

$$\hat{\varphi}_{K,h}[S](\boldsymbol{x}) = \sum_{i=1}^{N} K\left(\frac{d(\boldsymbol{x}, \boldsymbol{x}_i)}{h}\right). \tag{1}$$

The influence of data objects and the smoothness of the estimate is controlled by both the window width h and the shape of kernel K: h controls the smoothness of the estimate, whereas K determines the decay of the influence of a data object according to the distance. Even if the number N of data objects is very large, in practice it is not necessary to compute N distances for calculating the kernel density estimate at a given object $\boldsymbol{x}$. In fact, the value of commonly used kernel

Algorithm 1 DE-cluster algorithm: Clustering based on density estimation

funct $\Phi(\boldsymbol{x}, S, K, h, I) \equiv \sum_{\boldsymbol{x}_i \in I(\boldsymbol{x})} K\left(\frac{d(\boldsymbol{x}, \boldsymbol{x}_i)}{h}\right)$.

funct $\mathit{Uphill}(\boldsymbol{x}, S, K, h, I) \equiv \boldsymbol{x} + \delta \frac{\nabla\Phi(\boldsymbol{x}, S, K, h, I)}{\|\nabla\Phi(\boldsymbol{x}, S, K, h, I)\|}$.

proc *DensityCluster*$(S[\,], K, h, I, \mathit{Cluster}[\,]) \equiv$
  for $i := 1$ **to** *Len*(S) **do**
    $\boldsymbol{x} := \mathit{ApproxFixedPoint}\big(\mathit{Uphill}(\boldsymbol{x}_i, S, K, h, I), \epsilon\big)$
    $\mathit{Cluster}[i] := \mathit{Nearest}(\boldsymbol{x})$
  od.

functions is negligible for distances larger than a few h units; it may even be zero if the kernel has bounded support, as it is the case, for example, for the square pulse. Using kernel-based density estimation, it is straightforward to decompose the clustering problem into three phases as follows.

A. Choose a window width h and a kernel function K.
B. Compute the kernel-based density estimate $\hat{\varphi}_{K,h}[S](\boldsymbol{x})$ from the given data set.
C. Detect the regions of the data space for which the value of the computed estimate exceeds a given threshold and group all data objects of these regions into corresponding clusters.

In the literature, many different definitions of cluster have been proposed formalizing the clusters referred to in step C above. A *density-based* cluster [18] collects all data objects included in a region where density exceeds a threshold. *Center-defined* clusters [19] are based on the idea that every local maximum of $\hat{\varphi}$ corresponds to a cluster including all data objects which can be connected to the maximum by a continuous, uphill path in the graph of $\hat{\varphi}$. Finally, an *arbitrary-shape* cluster [19] is the union of center-defined clusters having their maxima connected by a continuous path whose density exceeds a threshold.

The DE-cluster Algorithm 1 essentially adopted from Hinneburg and Keim [19] implements the computation of center-defined clusters by an uphill climbing procedure driven by the density estimate. Function Φ approximates $\hat{\varphi}$ by restricting the set of data objects considered in the summation to a neighbourhood of $\boldsymbol{x}$. A function I specifies the neighbourhood. *Uphill* returns an object which is δ units away from $\boldsymbol{x}$ in the steepest direction of Φ, which is given by its gradient. The approximate fixed point function *ApproxFixedPoint* stops the computation of *Uphill* when the difference between two consecutive objects returned by *Uphill* is smaller than ϵ. Finally, the cluster index of object $\boldsymbol{x}_i$ is set as the index of the nearest neighbour in the data set S of the fixed object.

The complexity of the DE-cluster algorithm is that of computing the approximate fixed point of the hill-climbing function, and executing N times a nearest neighbour query. An efficient implementation approximates the gradient by $\sum_{\boldsymbol{x}_i \in I(\boldsymbol{x})} (\boldsymbol{x}_i - \boldsymbol{x}) K\left(\frac{d(\boldsymbol{x}_i, \boldsymbol{x})}{h}\right)$ (cf. [19]) and takes the nearest data object at every step of the climbing path. Then searching for a fixed object can be stopped whenever an already clustered data object is reached, since objects in the entire path can be labeled as this clustered object. The number of visited data objects is therefore N, and for each visit a k-nearest neighbour query is executed. The function $I(\cdot)$ can be implemented as such a query to a spatial access method like KD- or MVP-tree. In summary, the total cost is then $O(Nq(N))$, where $q(N)$ is the cost of a nearest neighbour query. In many practical cases, $q(N)$ is very close to $\log N$.

5 Distributed Data Clustering

The body of work on applications of data clustering in distributed environments, the problem of so called *distributed data clustering* (DDC), is comparatively small. In this section we adopt the kernel density estimation based clustering approach presented above for the distributed case assuming homogeneous data, which means that a data object cannot be split across two sites.

5.1 The DDC Problem

Let $\mathcal{A}(\cdot)$ be a clustering algorithm mapping any data set S to a *clustering* of S, that is, a collection of pairwise disjoint subsets of S. We define the problem of *homogeneous distributed data clustering* for clustering algorithm $\mathcal{A}$ as follows. Let $S = \{\boldsymbol{x}_i \mid i = 1, \ldots, N\} \subseteq \mathbb{R}^n$ be a data set of objects. Let $L_j, j = 1, \ldots, M$, be a finite set of *sites*. Each site L_j stores one data set D_j, and it will be assumed that $S = \bigcup_{j=1}^{M} D_j$. The DDC problem then is to find for $j = 1, \ldots, M$, a site clustering $\mathcal{C}_j$ residing in the data space of L_j, such that

(*i*). $\mathcal{C}_j = \{C \cap D_j : C \in \mathcal{A}(S)\}$ (*correctness requirement*)
(*ii*). Time and communications costs are minimized (*efficiency requirement*)
(*iii*). At the end of the computation, the size of the subset of S which has been transferred out of the data space of any site L_j is minimized (*privacy requirement*).

The traditional solution to the homogeneous distributed data clustering problem is to simply collect all the distributed data sets D_j into one centralized repository where their union S is computed, and the clustering $\mathcal{C}$ of the union S is computed and transmitted to the sites. Such approach, however, does not satisfy our problem's requirements both in terms of privacy and efficiency. We therefore propose a different approach yielding a kernel density estimation based clustering scheme, called KDEC, which may be implemented by appropriately designed DM agents of an agent-based DDM system.

5.2 The KDEC Scheme for DDC

The key idea of the KDEC scheme is based on the observation that the density estimate computed on each local data set conceals the details of the objects of the data set. Moreover, the local density estimate can be coded to provide a more compact representation of the data set for the purpose of transmission. In the sequel, we tacitly assume that all sites L_j agree on using a global kernel function K and a global window width h. We will therefore omit K and h from our notation, and write $\hat{\varphi}[S](\boldsymbol{x})$ for $\hat{\varphi}_{K,h}[S](\boldsymbol{x})$.

Density estimates in the form of Equation (1) are additive, i.e. the global density estimate $\hat{\varphi}[S](\boldsymbol{x})$ can be decomposed into the sum of the site density estimates, one estimate for every data set D_j:

$$\hat{\varphi}[S](\boldsymbol{x}) = \sum_{j=1}^{M} \sum_{\boldsymbol{x}_i \in D_j} K\left(\frac{d(\boldsymbol{x}, \boldsymbol{x}_i)}{h}\right) = \sum_{j=1}^{M} \hat{\varphi}[D_j](\boldsymbol{x}). \tag{2}$$

Thus, the local density estimates can be transmitted to and summed up at a distinguished *helper site* yielding the global estimate which can be returned to all sites. Each site L_j may then apply, in its local data space, the hill-climbing technique of the DE-cluster Algorithm 1 to assign clusters to the local data objects. There is nevertheless a weakness in such a plan: definition of a density estimate explicitly refers to all the data objects $\boldsymbol{x}_i$. Hence, knowing how to manipulate the estimate entails knowing the data objects, which contradicts the privacy requirement. However, only an intensional, algebraic definition of the estimate includes knowledge of the data objects. Multidimensional sampling provides an alternative extensional representation of the estimate which makes no explicit reference to the data objects.

Let $Diag[\boldsymbol{u}]$, $\boldsymbol{u} = [u_1, \ldots, u_n]^T \in \mathbb{R}^n$, denote the $n \times n$ diagonal matrix having diagonal $\boldsymbol{u}$, that is, the matrix $\{a_{ij}\}$ defined by

$$a_{ij} = \begin{cases} u_i & \text{if } i = j, \\ 0 & \text{otherwise.} \end{cases}$$

Further let $\tau = [\tau_1, \ldots, \tau_n]^T \in \mathbb{R}^n$ a vector of *sampling periods*. The *sampled form* of $\hat{\varphi}[S](\boldsymbol{x})$ at intervals $\tau = [\tau_1, \ldots, \tau_n]^T$ is the sequence $\{(x[S])_z\}$, $z \in \mathbb{Z}^n$, defined by

$$(x[S])_z = \hat{\varphi}[S](Diag[z] \cdot \tau), \tag{3}$$

where $\cdot$ is the inner product between vectors. Therefore, $(x[S])_z$ is the sequence of the values of $\hat{\varphi}[S](\boldsymbol{x})$ computed at all the real, n-dimensional vectors whose ith coordinates are spaced by a multiple of the ith sampling period τ_i, $i = 1, \ldots, n$. The sampled forms of the local density estimates are defined in a similar way by

$$(x[D_j])_z = \hat{\varphi}[D_j](Diag[z] \cdot \tau) \qquad j = 1, \ldots, M. \tag{4}$$

It is immediate to see by (2) that addivity holds for the sampled forms:

$$(x[S])_z = \sum_{j=1}^{M} (x[D_j])_z \qquad j = 1, \ldots, M. \tag{5}$$

Therefore, after receiving the sampled forms $(x[D_j])_z$ of the M density estimates, the helper site, can compute by (5) the sampled form of the overall estimate and transmit it to the sites L_j. Sites L_j can then apply the hill-climbing procedure of the DE-cluster Algorithm 1 to cluster local data with respect to the overall density estimate, using the interpolation formula

$$\sum_{z \in \mathbb{Z}^n} (x[S])_z \; sinc(Diag[\tau]^{-1} \cdot (\boldsymbol{x} - Diag[z] \cdot \tau)) \tag{6}$$

where

$$sinc(\boldsymbol{x}) = \prod_{i=1}^{n} \frac{\sin x_i}{x_i}, \tag{7}$$

to compute the values of the overall estimate that are needed in the hill-climbing function.

We briefly discuss the extent to which function (6), which follows from Shannon's theorem, represents $\hat{\varphi}[S](\boldsymbol{x})$. Shannon's theorem asserts that sampling a function $g(\boldsymbol{x})$ is an invertible transformation if, for every coordinate $i = 1, \ldots, n$, there is a frequency f_i such that the absolute value of the Fourier transform (the *amplitude spectrum*) of g differs from zero only in the interval $[-f_i, f_i]$ and the samples are computed with a period not greater than $\tau_i = \frac{1}{2f_i}$. Under these assumptions, the value of the interpolation formula computed at $\boldsymbol{x}$ equals $g(\boldsymbol{x})$. Unfortunately, most popular kernel functions do not satisfy this constraint since there is no upper bound to the set of frequencies at which the amplitude spectrum is not zero. That, in turn, means that any summation of such kernels does not satisfy Shannon's theorem hypothesis either. Consequently Shannon's theorem does not apply to density estimates and sampling in such a case is a lossy transformation. However, it can be shown that the kernel density estimate's amplitude spectrum is negligible outside a neighbourhood of the origin of radius not greater than a few $\frac{1}{h}$ units. Therefore, the global density estimate can be reconstructed from its samples by (6) introducing only a small error. Besides, if the used kernel function has a bounded support the same goes with the density estimate. That means there are only finitely many values in the sampled form, thus the number of terms in summation (6) is finite. If, however, the kernel function has unbounded support, like the Gaussian kernel, then the density estimate can be approximated by setting its value to zero where it is less than a threshold $\epsilon_{\hat{\varphi}}$.

According to this approach, we propose the following algorithmic KDEC scheme for computing the kernel density estimation based clusters for local data spaces at M distributed data sites L_j (see Algorithm 2). Every local site runs the procedure *DataOwner* whereas the helper site runs *Helper*. *DataOwner* is passed a reference to the helper L and the local data set $D[\,]$, and returns a clustering vector *Cluster*$[\,]$. *Helper* is passed a list of references $L[\,]$ to the local sites. Procedure *Negotiate* carries out a negotiation with the other local sites through the helper site to determine the sampling periods τ, the boundaries of the sampling rectangle $z_1, z_2 \in \mathbb{Z}^n$, the kernel K and the window width h.

Algorithm 2 KDEC: distributed clustering based on density estimation

funct $\Phi(\boldsymbol{x}, S, K, h, I) \equiv \sum_{\boldsymbol{x}_i \in I(\boldsymbol{x})} K\left(\frac{d(\boldsymbol{x}, \boldsymbol{x}_i)}{h}\right)$**.**
funct $Sample(D, \tau, z_1, z_2, K, h, I) \equiv$
  $\{\Phi(Diag[z] \cdot \tau, D, K, h, I) : z_1 \leq z \leq z_2\}$**.**
funct $Reconstruct(\boldsymbol{x}, \tau, z_1, z_2, Sam) \equiv$
  for $z := z_1$ **to** z_2 **do**
    $r := r + Sam[z]\, Sinc(Diag[\tau]^{-1} \cdot (\boldsymbol{x} - Diag[z] \cdot \tau))$
  od;
  r**.**
funct $Uphill(\boldsymbol{x}, \tau, z_1, z_2, Sam) \equiv$
  $\boldsymbol{x} + \delta \frac{\nabla Reconstruct(\boldsymbol{x}, \tau, z_1, z_2, Sam)}{\|\nabla Reconstruct(\boldsymbol{x}, \tau, z_1, z_2, Sam)\|}$**.**
proc $DataOwner(D[\,], L, Cluster[\,]) \equiv$
  $Negotiate(L, \tau, z_1, z_2, K, h, I));$
  $Send(Sample(D, \tau, z_1, z_2, K, h, I), L);$
  $Sam := Receive(L);$
  for $i := 1$ **to** $Len(D)$ **do**
    $\boldsymbol{x} := ApproxFixedPoint\big(Uphill(\boldsymbol{x}, \tau, z_1, z_2, Sam), \epsilon\big);$
    $Cluster[i] := Nearest(\boldsymbol{x});$
  od.
proc $Helper(L[\,]) \equiv$
  $Negotiate(L);$
  $\boldsymbol{X} := \boldsymbol{0};$
  for $j := 1$ **to** $Len(L)$ **do** $\boldsymbol{X} := \boldsymbol{X} + Receive(L[j])$ **od**;
  for $j := 1$ **to** $Len(L)$ **do** $Send(\boldsymbol{X}, L[j])$ **od.**

Negotiate is run by the local sites and the helper site, and contains appropriate handshaking primitives to ensure that all sites participate and exit *Negotiate* only if an agreement has been reached. Each local site computes the sampled form of the estimate of $D[\,]$, and sends it to the helper. The helper receives the lists of sampled estimates, sums them by order in the lists to compile a list of global sample needed to compute the global estimate, and returns this list to all sites. Procedures *Send* and *Receive* implement appropriate blocking and handshaking to ensure the transmission takes place. Each local site uses the global sample in procedure *Reconstruct* to compute the values of the global density estimate and performs a DE-cluster algorithm to compute the corresponding local data clusters.

5.3 Complexity of the KDEC Scheme

In terms of the complexity in computation and communication one crucial point of the KDEC scheme is how many samples have to be computed and transferred among the sites. Since the largest frequency at which the absolute value of the Fourier transform of the kernel estimate is not negligible depends only on the window width h, and is not greater than a few $\frac{1}{h}$ units, at any site L_j, the number of samples to be computed at L_j is only a fraction of the number of data objects. The computational costs of the KDEC scheme in terms of used CPU cycles and I/O do not exceed the one in the centralized approach where clustering is performed on data collected in a single repository. The computational complexity is linear in the number of samples. Of course, the precise cost of computation of any KDEC-based DDC algorithm as an instance of the proposed scheme largely depends also on the used kernel function and local clustering algorithm. The DE-cluster algorithm we developed for the KDEC scheme in Section 4.2 is of complexity $O(Nq(N))$, where $q(N)$ is the cost of a nearest neighbour query (which in practical cases is close to $\log N$). Since the size of samples is usually much smaller than that of the corresponding data set, and the cost of communicating samples of local kernel density estimates among the local sites and the helper site is linear, the overall communication costs of our DDC approach in any case will be significantly lower than in a centralized approach.

5.4 Agent Technology for KDEC-Based DDC

We assume a DDM system consisting of a set of networked, homogeneous data sites. Each data site respects its local autonomy by individually granting read-only access to internal data mining agents.

One option of agent-based implementation of the KDEC based data clustering is to implement a set of appropriate stationary DM agents, each of which is associated to one of multiple, networked local data sites (Figure 1). According to the KDEC scheme, an agent takes the role of the helper and it engages the other site agents in a negotiation (Figure 2) to agree (a) what kernel function to use for computing the local density estimate samples, and (b) the use of the DE-cluster algorithm for local clustering based on the global estimate.

The diagram in Figure 2 is a possible implementation of interactions between agents according to the communicative acts of ACL FIPA. The interactions are based on the primitives of such a standard (e.g. *call-for-proposal, accept, inform...*) which give a first level semantics to message exchanged between agents.

Another option bases on the idea that under certain constraints a mobile helper agent can perform the KDEC based DDC computation in the overall DDM system (Figure 3). In this case, the helper agent moves through a sequence $\{L_n\}$ of data sites. At each site n the agent engages the agent site, which guarantees local data access services, in the initial negotiation as depicted in Figure 4; then the visiting agent—which carries in its data space, for every sampling point, the sum of all samples of every site L_m, $(m < n)$, at that point—(a) computes the sampled form of the density estimate of local data by means of the agent site and sums the local samples, and then (b) at the end of the trip returns the global estimate's sample to agent sites interested in the cluster analysis (see the *inform* message at the end of the interaction diagram in Figure 4).

Please note that the role of the central helper site in the KDEC scheme is taken by the mobile DM agent. This agent may be initiated either by some distinguished central site, or by any of the local sites which then is in charge of coordinating the actions of its mobile agent together with its stationery agent. In any case, an appropriate mobile agent infrastructure, a cooperation protocol

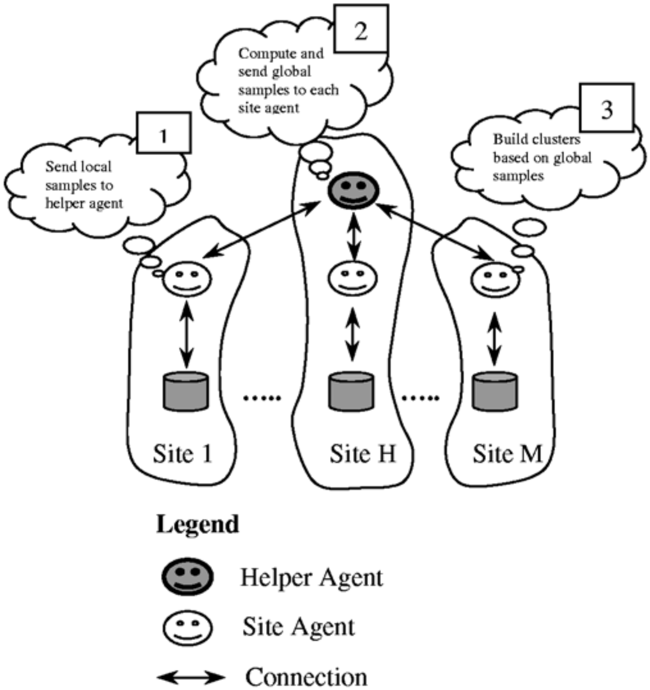

Fig. 1. Outline of agent-based implementation of the KDEC scheme for DDC

between the mobile agent and site agents, as well as a proper security architecture have to be implemented in such a system, mostly to avoid unauthorized data access and privacy violations.

As an application consider the task of market segmentation in a company structured into one headquarter and a large number of geographically distributed branches. The problem is how to effectively and efficiently partition the company's customers into classes depending on their properties in order to implement the same strategy for a given class across all branches. For this purpose, the company may launch a mobile agent which collaborates with each site agent for clustering all customers using the KDEC scheme with respect to one global set of cluster identifiers.

6 Related Work

Only a few approaches to solve the problem of distributed data clustering are available to date. In [22] a tree clustering approach is taken to build a global dendrogram from individual dendrograms that are computed at local data sites subject to a given set of requirements. In contrast to the KDEC based DDC approach the distributed data sets are assumed to be heterogeneous, therefore

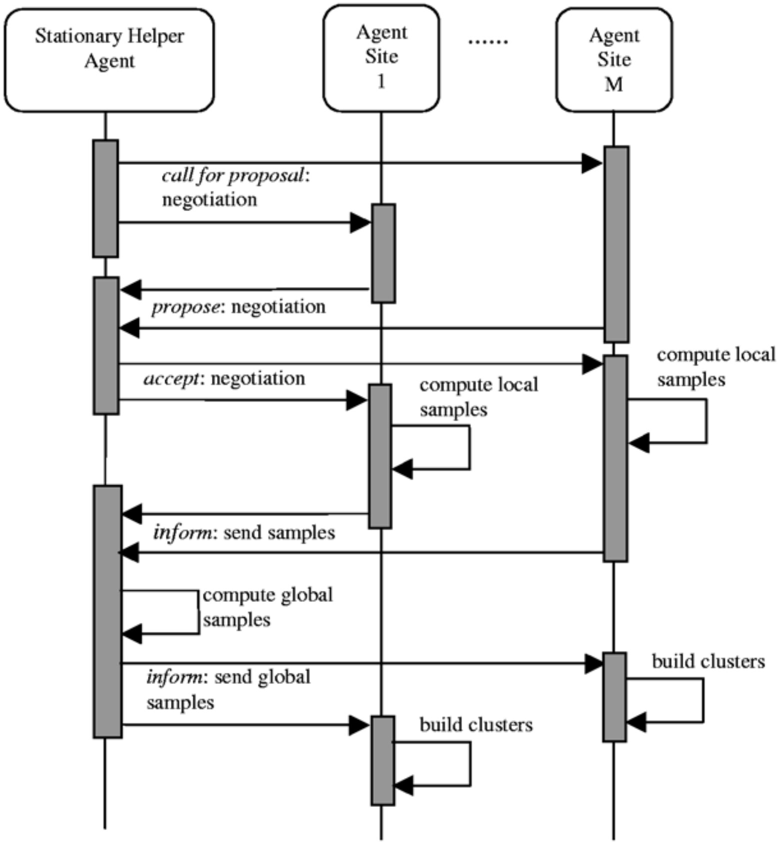

Fig. 2. Inter-agent interactions of the stationary agent-based implementation of the KDEC scheme for DDC

every site has access only to a subset of the features of an object. The proposed solution implements a distributed version of the single link clustering algorithm which generates clusters that are substantially different from the ones generated by density-based methods like the KDEC scheme. In particular, it suffers from the so-called chaining effect, by which any of two homogeneous and well separated groups of objects connected only by a dense sequence of objects are regarded as a single cluster. [23] proposes a technique for distributed principal component analysis, Collective PCA. It is shown that the technique satisfies efficiency and data security requirements and can be integrated with existing clustering methods in order to cluster distributed, high-dimensional heterogeneous data. Since the dimensionality of the data is reduced prior to clustering by applying PCA, the approach is orthogonal to ours.

Another related research direction concerns incremental clustering algorithms. The BIRCH [24] and related BUBBLE method [25], compute the most

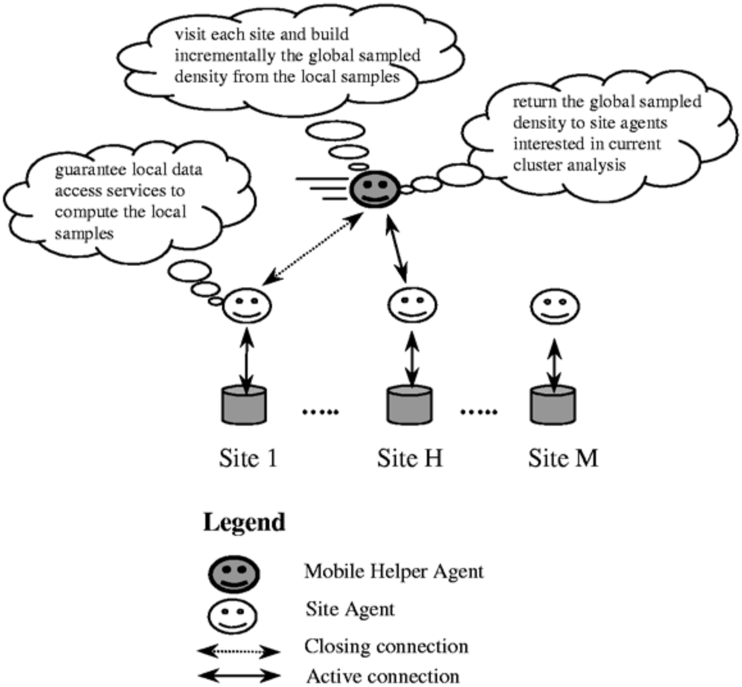

Fig. 3. Outline of the mobile agent-based implementation of the KDEC scheme for DDC

accurate clustering, given the amount of memory available, while minimizing the number of I/O operations. It uses a dynamic index structure of nodes that store synthetic, constant-time maintainable summaries of sets of data objects. The method is sufficiently scalable requiring $O(N \log N)$ time and linear I/O. However, since it uses the centroid to incrementally aggregate object, the method exhibits a strong bias towards globular clusters. IncrementalDBSCAN [26] is a dynamic clustering method supporting both insertions and deletions, which is shown to be equivalent to the well-known static DBSCAN algorithm. Since in turn DBSCAN can be shown to be equivalent to a method based on density estimation when the kernel function is the square pulse and the clusters are density-based, IncrementalDBSCAN is less general than methods based on kernel density estimates. Its time complexity is $O(N \log N)$.

It is worth noting that, in an agent-based framework, incremental clustering techniques are potentially useful to efficiently support the incremental updates a clustering model undergoes as an agent visits data sites. However, the latter two approaches require agents to access all data available thereby violating the data privacy requirement. Our KDEC scheme shows more flexibility in that it works for any implementation where agents may be granted read-only access by autonomous data sites. However, like all other existing DDM systems the KDEC

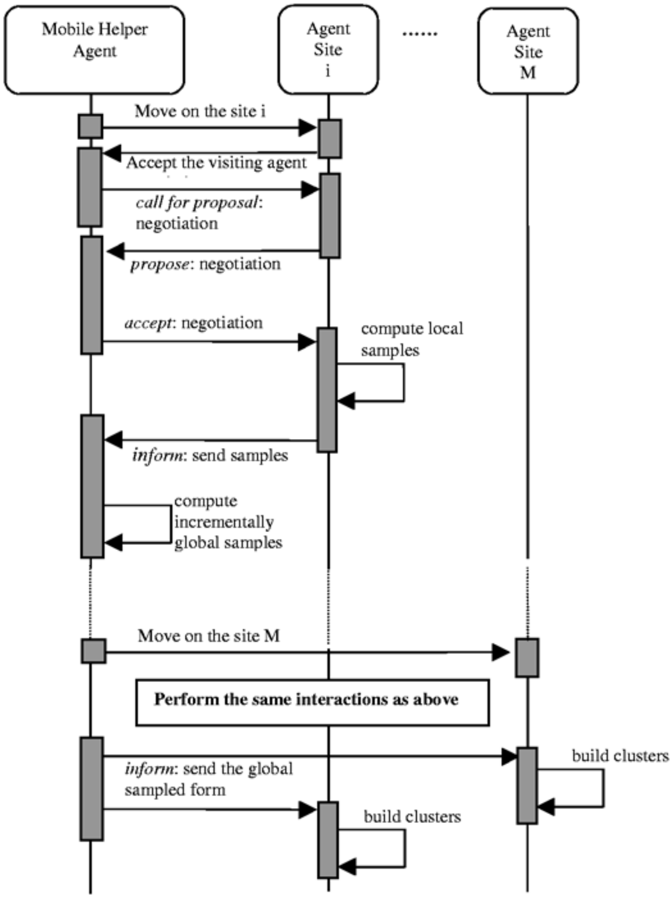

Fig. 4. Inter-agent interactions of the mobile agent-based implementation

scheme fails to solve the inference problem mentioned in Section 3. In fact, from a (sampled) density estimate one can easily generate data sets which look like the original one, though only the helper site agent can do this separately for the local data sets, and the actual data objects can never be identified.

7 Conclusion

Due to the explosion in the number of autonomous data sources there is a growing need for effective approaches to distributed agent-based knowledge discovery and data mining. In this paper, we have reviewed prominent approaches in the literature and presented a novel scheme for agent-based distributed data clustering. The approach exploits statistical density estimation and information theoretic sampling to minimize communications between sites. Moreover, the privacy of data is preserved to a large extent by never transmitting data values but kernel based density estimation samples outside the site of origin. The approach does not require CPU and I/O costs significantly higher than a similar centralized approach and its communication costs may be lower. Ongoing research focus in particular on implementations of a multiagent system for KDEC-based DDC in a peer-to-peer network, and investigation on methods to mitigate the risk of security and privacy violations in DDM environments.

Acknowledgments. This work was partially supported by CSITE-CNR, project D2I of the Ministry of Education, University and Scientific Research of Italy, and project AGRICOLA under grant 032002 of the Saarland Ministry of Economics, Germany. The authors are also thankful to Claudio Sartori for valuable discussions on the subject of the paper.

References

[1] Fayyad, U., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R.: Advances in Knowledge Discovery and Data Mining. AAAI Press/MIT Press (1996)

[2] Chen, M.S., Han, J., Yu, P.S.: Data mining: an overview from a database perspective. IEEE Trans. On Knowledge And Data Engineering **8** (1996) 866–883

[3] Kargupta, H., Park, B., Hershberger, D., Johnson, E.: 5, Collective Data Mining: A New Perspective Toward Distributed Data Mining. In: Advances in Distributed and Parallel Knowledge Discovery. AAAI/MIT Press (2000) 131–178

[4] Moro, G., Sartori, C.: Incremental maintenance of multi-source views. In: M. E. Orlowska, J. Roddick Edition, Proceedings of 12th Australasian Database Conference, ADC 2001, Brisbane, Queensland, Australia, IEEE Computer Society (2001) 13–20

[5] Dhar, V., Chou, D., Provost, F.J.: Discovering interesting patterns for investment decision making with glower - a genetic learner. Data Mining and Knowledge Discovery **4** (2000) 251–280

[6] Klusch, M.: Information agent technology for the internet: A survey. Data and Knowledge Engineering, Special Issue on Intelligent Information Integration. Elsevier Science **36** (2001) 337–372

[7] Wooldridge, M.: Intelligent agents: The key concepts. In Marík, V., Stepánková, O., Krautwurmova, H., Luck, M., eds.: Multi-Agent-Systems and Applications. Volume 2322 of LNCS., Springer-Verlag (2002) 3–43

[8] Kargupta, H., Hamzaoglu, I., Stafford, B.: Scalable, distributed data mining using an agent-based architecture. In D.Heckerman, H.Mannila, D.Pregibon, R.Uthurusamy, eds.: Proc. 3rd International Conference on Knowledge Discovery and Data Mining, Newport Beach, California, USA, AAAI Press (1997) 211–214

[9] Stolfo, S.J., Prodromidis, A.L., Tselepis, S., Lee, W., Fan, D.W., Chan, P.K.: JAM: Java agents for meta-learning over distributed databases. In David Heckerman, Heikki Mannila, D.P., ed.: Proc. Third International Conference on Knowledge Discovery and Data Mining (KDD-97), Newport Beach, California, USA, AAAI Press (1997) 74–81

[10] Bailey, S., Grossman, R., Sivakumar, H., Turinsky, A.: Papyrus: a system for data mining over local and wide area clusters and super-clusters. In: Proc. Conference on Supercomputing, ACM Press (1999) 63

[11] Papazoglou, M.P., Schlageter, G.: Cooperative Information Systems - Trends and Directions. Academic Press Ltd, London, UK (1998)

[12] Zhong, N., Matsui, Y., Okuno, T., Liu, C.: Framework of a multi-agent kdd system. In Yin, H., Allinson, N.M., Freeman, R., Keane, J.A., Hubbard, S.J., eds.: Proc. of Intelligent Data Engineering and Automated Learning - IDEAL 2002, Third International Conference, Manchester, UK. Volume 2412 of Lecture Notes in Computer Science., Springer-Verlag (2002) 337–346

[13] Sen, S., Biswas, A., Gosh, S.: Adaptive choice of information sources. In: Proc. 3rd International workshop on Cooperative Information Agents, Springer (1999)

[14] Theilmann, W., Rothermel, K.: Disseminating mobile agents for distributed information filtering. In: Proc. of 1st International Sympos. on Mobile Agents, IEEE Press (1999) 152–161

[15] Prasad, M., Lesser, V.: Learning situation-specific coordinating in cooperative multi-agent systems. Autonomous Agents and Multi-Agent Systems (1999)

[16] Grabmeier, J., Rudolph, A.: Techniques of cluster algorithms in data mining. Data Mining and Knowledge Discovery **6** (2002) 303–360

[17] Ankerst, M., Breunig, M.M., Kriegel, H.P., Sander, J.: OPTICS: Ordering points to identify the clustering structure. In: Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data, Philadephia, PA (1999) 49–60

[18] Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, OR (1996) 226–231

[19] Hinneburg, A., Keim, D.A.: An efficient approach to clustering in large multimedia databases with noise. In: Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD-98), New York City, New York, USA, AAAI Press (1998) 58–65

[20] Schikuta, E.: Grid-clustering: An efficient hierarchical clustering method for very large data sets. In: Proceedings of the 13th International Conference on Pattern Recognition, IEEE (1996) 101–105

[21] Silverman, B.W.: Density Estimation for Statistics and Data Analysis. Chapman and Hall, London (1986)

[22] Johnson, E., Kargupta, H.: Collective, hierarchical clustering from distributed heterogeneous data. In Zaki, M., Ho, C., eds.: Large-Scale Parallel KDD Systems. Lecture Notes in Computer Science. Springer-Verlag (1999) 221–244

[23] Kargupta, H., Huang, W., Krishnamoorthy, S., Johnson, E.: Distributed clustering using collective principal component analysis. Knowledge and Information Systems Journal **3** (2000) 422–448

[24] Zhang, T., Ramakrishnan, R., Livny, M.: BIRCH: An efficient data clustering method for very large darabases. In: Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data, Montreal, Canada (1996) 103–114

[25] Ganti, V., Ramakrishnan, R., Gehrke, J., Powell, A.L., French, J.C.: Clustering large datasets in arbitrary metric spaces. In: Proceedings of the 15th International Conference on Data Engineering (ICDE 1999), Sydney, Austrialia (1999) 502–511

[26] Ester, M., Kriegel, H.P., Sander, J., Wimmer, M., Xu, X.: Incremental clustering for mining in a data warehousing environment. In: Proceedings of the 24th International Conference on Very Large Data Bases (VLDB'98), New York City, NY (1998) 323–333

User Profiling with Privacy: A Framework for Adaptive Information Agents

Ian Dickinson[1], Dave Reynolds[1], Dave Banks[1], Steve Cayzer[1], and Poorvi Vora[2]

[1] Hewlett-Packard Laboratories
Filton Road, Stoke Gifford
Bristol BS34 8QZ, UK
{ian.dickinson, dave.reynolds, dbanks, steve.cayzer}@hp.com
[2] Hewlett-Packard Laboratories
1000 NE Circle Blvd Corvallis, OR 97330, USA
poorvi_vora@hp.com

Abstract. This paper presents a framework for personal agents that respect the privacy of the individual. We present some motivations and outline a framework for the use of personal agents and user profiling for information systems designed around web services. A key element of our approach in general is to consider the impact of user-profiling and autonomous agents on the user. One particular aspect, which we explore in this paper, is the need to respect user's privacy. One often-cited benefit of using personal agents is for personalising interaction. However, personalisation and privacy have contradictory goals in disclosing personal information. We explore some elements of our framework that allow the user to control the trade-offs around disclosure of personal information. We conclude with some motivating examples of the use of our framework in information-based tasks.

1 Introduction

A widely recognised problem for users of information systems today is the frustration caused by being unable to find and manage the right information. This is often referred to as *information overload*. The increasing range and sophistication of interesting sources of content on the World Wide Web provides more choices, and, in principle, a greater chance of ultimate success. Such plenitude can be elusive, however, when the costs to the user, in terms of time, effort and attention, are too great. One way that information overload can be addressed, is to use more information about the user's needs and objectives in the process of finding and using information resources – known generically as *personalisation* [1]. There are many ways to achieve personalisation [2], but here we focus on the use of information about the user to *adapt* the user's interaction to their specific needs. The study of *adaptive information agents* [3] addresses the design, use and evaluation of agent-based tools to assist a user to perform information-centric tasks effectively. Here the agent metaphor captures the intuitive concept of a knowledgeable third party, for example, a trained librarian. While the term *agent* is widely used to mean a number of different things [4], our present objective is to build tools to assist the end-user. Hence, we are not referring to mobile or infrastructure agents, but to what are often termed *personal agents* [5].

Applications built around the personal agent metaphor have been widely explored in the literature, and some illustrative examples are outlined below. A common feature of these applications is the use of knowledge about the user in order to provide

M. Klusch et al. (Eds.): Intelligent Information Agents, LNAI 2586, pp. 123-151, 2003.

an adaptive, or otherwise personalised, service. In this context, many authors raise the question of privacy for the end-user as a concern with such user models. A focus for our work is an attempt to build an open, extensible user-profiling platform, but which emphasises privacy and security of the user's personal data. The research we are reporting is still ongoing: this paper presents motivations, results to date, and an overall positioning of our research in the context of related research fields.

The paper is structured as follows: section 2 defines the general need for comprehensive representation of user preferences as a foundation for novel information management tools, including information agents. Section 2 also briefly reviews some prior work in user modelling, and describes extensions to prior approaches that motivate our research. In section 3, we outline the core components of our approach, which we call the *ePerson* approach[1]. We recognise that a fundamental requirement in any system that manages user information is privacy, and we regard our approach to privacy as one of the distinguishing features of our approach. Section 4 reviews the issues for privacy, and some of our research on solutions to these issues. Section 5 describes some application scenarios for the ePerson. We conclude with a review of similar work, and summary of future research issues.

2 Motivation: The Need for General User Models

The use of information about a user to modify that user's interaction with a computing system or information resource is well established. Typically, the stored information about the user is referred to as a *user profile* or *user model*. User-modelling is an established research field in artificial intelligence [6]. Many adaptive or personalised systems contain an embedded user model, and use this local model as the foundation for, e.g., personalised recommendations [7]. It has been observed that this approach requires the application developer to build and maintain their own user model, which has a number of drawbacks. Principle concerns about such an approach are:

- It requires the application developer to spend additional time designing and developing a user-modelling component. This adds to the effort required to build the application. In addition, the developer may not have the time, or expertise, to address all of the relevant concerns relating to the user model, including, for example, privacy protection.
- Information is typically not shared between user models. This has two impacts: first, the subset of user information in any single model gives a less complete, and arguably less helpful, picture of the user's needs and preferences. Secondly, there is a significant cost to the user in having to manage multiple instances of their personal preference data, especially as user model data is known to change frequently [8].

A *generic user model* attempts to alleviate these concerns by centralising user model construction and maintenance in one program (e.g. a user modelling server), which then makes the user data available to client applications. In a recent article,

[1] We note that the terminology "ePerson" was first coined by Roscheisen [85], though in the more specific context of rights contract negotiations.

Alfred Kobsa reviewed a range of generic user modelling tools, ranging from research prototypes to commercial products [9]. A companion paper by Fink and Kobsa compared the features of four classes of currently available commercial personalisation tools [2]. These general-purpose tools attempt to abstract away the problem of constructing and maintaining a user model, providing a *user modelling service* for other application components to utilise. Note that this is an abstract description, and is neutral about the architecture used to integrate a user model into the application[2]. A summary of Kobsa's list of abstract services includes:

- the representation of assumptions about one or more types of user characteristics in models of individual users, formation of those assumptions, and deriving new assumptions based on observations;
- the representation of relevant common characteristics of specific user subgroups of the application domain (often called *stereotypes*), classifying users according to these subgroups, and drawing inferences according to sub-group membership;
- the recording of user's behaviour, particularly their past interactions with the system;
- the generalisation of the interaction histories of many users into stereotypes;
- consistency maintenance in the user model (from Kobsa 1995, quoted in [9] p52).

There is every reason to suppose, going forward, that all of the above capabilities will continue to be useful or essential in the construction of personalised or adaptive user interfaces. However, as the world of networked information services grows in complexity, we can anticipate a number of additional requirements that will need to be addressed by a new generation of user profiling services, as follows:

- *small devices and intermittently connected networks* – adaptive interfaces will be needed on information appliances, characterised by reduced computing power and intermittent connection to the network (see also [10;11;12] and [9] p58,)
- *awareness of the user's current context* – including, but not limited to, the user's current interaction device (PDA, phone, kiosk, desktop computer, etc), the user's location and the availability of location-based services and the user's current role (see also [12;13;14;15])
- *open and extensible* – as more and more user information is available via web-based services, it is clear that the use model can never be specified as a closed system. Instead, the model should be able dynamically to incorporate information from disparate sources, and not duplicate information that is maintained elsewhere. Ideally, the user model should provide uniform, consistent and flexible access to all sources of user information, and support a multiplicity of means of acquiring information about the user;
- *privacy protection* – commensurate with the increasing range and scope of information about individuals stored on computers, concerns about protecting privacy have increased dramatically. We discuss the many dimensions of privacy in more detail in section 0.

---

[2] For example, the user model may be a separate server-based process, or may be a self-contained component that simply plugs into a stand-alone application. We will further discuss architectural issues in section 2.

In essence, it is these additional requirements that provide the motivation for our approach to the user profiling problem, as we discuss further below.

While there are many dimensions along which we can compare generic user modelling systems, two interesting clusters emerge from previous developments in academia and industry. On the one hand, academic approaches have typically favoured deep, sophisticated[3] models, but focussed less on practical issues, whereas commercial systems have generally employed shallower models, but emphasise scalability, security and performance. Kobsa identifies the key features of the academic approach as follows: generality (including domain independence), expressiveness, and strong inference capability. It is also common that academic generic user models employ the "mentalist" paradigm, recording the model's understanding of the user's beliefs, goals, knowledge, etc, often using a modal logic [16]. The alternative approach, typified by the more commercially oriented personalisation servers (and also by machine-learning approaches to user modelling [8]) does not attempt to define the user model in declarative, mentalist terms. For example, the GroupLens product from Net Perceptions [17] provides recommendations by associating patterns of user activity with outcomes, such as reading a news article or buying a book. GroupLens cannot produce a declarative explanation of the user's needs (e.g. an SQL programmer learning Java[4] might be predicted to have a need for information on JDBC), but, given enough data, can recognise a pattern (e.g. "people who bought books on SQL and on learning Java also bought books on JDBC). The reasoning in such systems is statistical, compared to the knowledge-based inferencing schemes in the mentalist academic user modelling systems. However, this has a strong benefit, in that such systems can be, and typically are, much more robust and predictable in their behaviour, and scalable to moderate size data sets. Consequently, such commercial systems are often better at addressing practical deployment issues, including scalability, privacy, and, to a lesser extent, integrating with existing or legacy sources of user data in the organisation (see review in [2]).

A user model may be extended by the addition of new user profile categories and individual data. This presents a difficulty in terms of the design goals we outlined above. For instance, we would like to empower users to express rich policies about when and how their private data may be used. To do this, the policy must be able to address the contents of the user model by function or category. For example, imagine the user wishes to express the following privacy policy:

```
Any service acting on behalf of my employer can access
my employment details, and preferences for job-related
information systems.
Any member of my photography circle can see details of
my published photographs on photo.com.
```

To carry out this policy, it must be possible, in the user model, to identify employment details and distinguish them from, say, contact details for friends. A common approach to this problem is exemplified by P3P [18]. P3P pre-defines a *standard data model*, containing *well-known names* for elements of the model. These

---

[3] Some would argue "complicated".

[4] Java is a registered trademark of Sun Microsystems.

well-known names are agreed in advance by users of the P3P standard to have a particular form and meaning. An example such name is `“user.home-info.online.email”`. This approach, however, does not support extensibility well. Even if a mechanism is provided for propagating new public names, their meaning will be hard to convey. What is needed is a means to be able to name or identify different parts of the user model, without identifying symbols necessarily being defined in a pre-existing standard. Beyond the difficulties of agreeing the meaning of shared symbols, we anticipate that significant research, and innovative design, will be necessary to make privacy policy specification amenable and attractive to the individual. The agent should give a significant amount of support to the average home or office user, ensuring that the goal of placing privacy under the user’s control is becomes real.

Another aspect of this problem concerns *open access* to the contents of the user model. It must be possible for services using the data from the user model to provide suitable adaptations, without necessarily knowing *a priori* what the elements of the model are. Even if tools like GroupMinds are able to recognise very robust patterns of behaviour, other services cannot integrate those patterns into their personalisation strategy unless there can be a meta-description of the affinities that have been detected.

Our goals for extending the user-profiling paradigm to include these features suggest a number of additional design goals that must be adopted. In summary, these are:

- Place the user in control of their own data;
- Make user profile data available more widely, including shareable between applications;
- Ensure that access to all profile information is governed by a privacy policy that the user controls;
- Do not assume a particular network topology; especially do not require all data to be centralised in one place;
- Allow the data model for the profile to be extended in a meaningful and open way.

3 An Outline Framework for Open Extensible User Profiles

Given the motivation, and in particular the design goals, outlined above, we are developing an open platform to provide a uniform means for applications to access and share user profile and user model information. This platform provides one basis for representing of the user in the shared information space. We use the term *ePerson* for this representation. Fig. 1, below, illustrates the key features of the platform. It provides a common, policy controlled, access mechanism for stored and indirectly obtained profile information that a variety of client agents can access.

For the user, such a common shared platform offers the advantages of convenience and control. Conceptually, the user profile is in one place and the user has one interaction point for checking and updating information, and adjusting the access control policies. In implementation, the profile might be distributed across several devices – for example, parts of it might be cached on the user’s PDA, phone or PC. However, it is important to maintain the illusion of a single control point both to simplify the user’s conceptual understanding of the profile (and how to control it), and to simplify the design of profile-using web services.

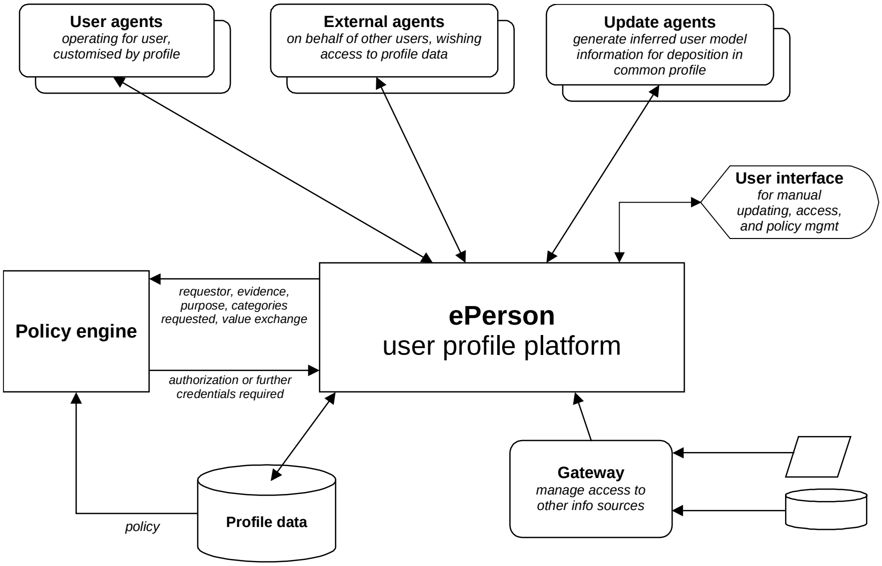

Fig. 1. Outline ePerson system framework

For applications, including information agents, wishing to access the user profile in order to carry out their tasks, the shared profile service provides uniform access to a richer set of user information than the application itself could have generated. It also relocates the problem of checking access requests against a user's privacy policy from the individual agents to a shared policy engine. By pooling this resource, the information agents can access better information more flexibly, while the user experiences greater control of the profile data.

Design Principles

This platform outlined above raises several design challenges, including:

- *Open platform*. The platform must support a dynamic, open-ended set of client agents and services.
- *Semantic extensibility*. The profile information itself must also be openly extensible. New agents may be able to generate and exploit new categories of profile information that the platform must accommodate. Even for a fixed class of information agents the precise structure and semantics of the profile information may evolve over time as the agents grow in sophistication.
- *User control*. The ePerson profile store is a store of information on behalf of one user, not a population profile. It is critical that the profile be trusted by the user. We propose to use policy-based access control, which is discussed further in section 4, below. It also suggests that solutions in which the information is directly under the user's control and pushed to the network edges are preferable which leads us to the following design principle:

- *Decentralization*. In addition to putting the data under clearer control of the user who owns it, decentralization of the platform is a core method for addressing issues of scalability and openness.
- *Partial network connections*. If the user data is held on the network edges, indeed sometimes in individual client devices, then the ability of external software agents to access that data will be compromised by any connectivity limitations. Conversely, client devices wishing to use the profile information to tailor their functions or interactions require access to parts of the profile even at times of intermittent connectivity. Thus, appropriate caching and replication strategies are required.
- *Legacy and external data support*. The ePerson profile is not a universal store. Rather, it attempts to provide an integrated view of data about the user in part by storing the data, and in part by referencing data stored elsewhere. The framework must support integrated access to data in such legacy or third-party services.

We now discuss these challenges, and the design principles they engender, in more detail.

Open Platform

One key motivation for the ePerson user profile is to provide an open-ended service that a variety of software agents and other services can use to better adapt to the user's preferences and needs. The consequence of supporting an such open platform is that it is not possible, or desirable, to fix a pre-specified set of services that will interact with the ePerson.

Our goal of building an open platform stems, in part, from the wider vision of the future of web services. Many commentators and researchers have suggested that IT applications of the future should be constructed from component, globally available, electronic services. HP was an early proponent of this vision of *e-Services* [19]. Other companies have articulated similar visions [20;21], leading to the recently industry moves towards an XML Protocol for web services [22] and associated directory and description solutions [23].

This move towards a common web services infrastructure is relevant to the development of a common ePerson user profile store in two ways. Firstly, general web services are as much in need of access to personal profile information to tailor their functionality and interface as are information agents. Secondly, in our framework, access to profile information is itself regarded as a web service, and specified in such a way that different implementations can interoperate using published web-services standards.

Semantic Extensibility

The role of the ePerson user profile platform is to allow an open-ended set of software agents and web services to access and exchange user profile information. This provides few constraints on the nature of the user profile itself. How can we provide useful structuring for such unconstrained data, without the problem reducing to a trivial lowest common denominator?[5] How can the ePerson define the structure and semantics for the profile information it is offering, in such a way that other agents can determine whether it meets their needs?

---

[5] It is not uncommon to see profile information only constrained as "keyword-value pairs".

We propose to satisfy these needs by building upon the *semantic web* technology platform. The semantic web is a vision for a future evolution of the World Wide Web to support the exchange of machine processable data [24]. In particular, it is envisioned as a key component of the information fabric through which personal agents exchange requirements and negotiate solutions [25]. The semantic web is based upon a tiered architecture, shown in Fig. 2:

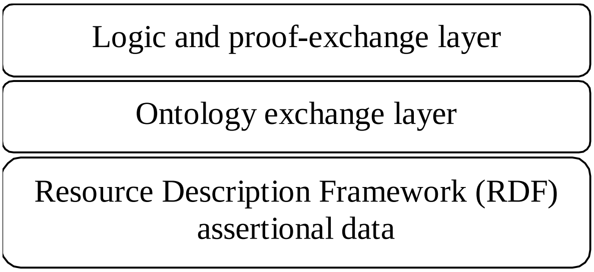

Fig. 2. Semantic web technology platform

The foundational layer, RDF, provides a common format for the exchange of data. It originated as metadata for describing web-addressable resources [26], but is now used more widely. RDF builds upon the XML standard for document formatting by providing a generic data model for graph-structured data, and a means of encoding that model using XML syntax. The data model is designed to support positive ground assertions of the form:

`<subject, predicate, object>.`

The `subject` and `predicate` values are described by RDF as resources, which are identified using uniform resource identifiers (URI's) [27]. The `object` value may also be a resource, or literal whose value is specified as a character string. A special case of resources is the *bNode*: a resource node that has no identifying URI. The bNode is equivalent to an existentially quantified variable. While this data model, frequently referred to in the RDF literature as *triples*, only directly encodes binary relations, it is straightforward to encode relations of any arity using the bNode construct.

RDF provides us with a common basis for the exchange of assertions that make up the user profile. The use of URI's as identifiers in the RDF data model allows it to support a weak form of openness: a guarantee that the set of terms in the profile can be extended without names clashing. For example, if one community of agents wishes to express information on, say, sports preferences, it could do so using resource URI's defined within a namespace under that community's control. A second group of agents wishing to record information on, say, news preferences, might then define their own resource URI's within their namespace. The distinctness of these namespaces then ensures that profile information referring to both news and sports will not suffer accidental duplication of identifiers.

However, ensuring distinct identifiers, while necessary, is not by itself sufficient. We also need to be able to discover and re-use extensions to the profile data model. Suppose the second group of agents, above, also wished to record specific sports-news preference information. It should be possible for them to discover that an appropriate namespace for sports information already exists, and build on that to create a rich profile that is compatible with other sports data. Even if the conceptual model of sports underlying the existing namespace is not powerful enough to support all of the needs of the second (news) agent group, it could still be possible to identify points of correspondence between the two conceptualisations of “sports”. This requires each agent group to provide an explicit, formal conceptual model for the vocabulary underlying the profile data they are using. This explicit conceptualisation is called an *ontology* [28].

The semantic web technology framework calls for a standard means of encoding and exchanging ontologies. Once this ontology support is in place, groups of agents and applications can use it to discover conceptual models they can reuse, to formalize (at least partial) mappings between such models, and to extend their domain vocabulary over time. At the time of writing[6], a W3C working group to define this standards layer is still being formed. However, a *de facto* standard already exists in the form of the DAML+OIL language [29]. DAML+OIL is the result of merging the OIL ontology standard, which resulted from the European OntoBroker project, with the early results of the DARPA Agent Markup Language (DAML) project.

DAML+OIL builds upon RDF, both in the sense that RDF ground facts can be used to provide instance data for the ontology, and in the sense that the ontology itself is encoded in RDF. This makes the ontology language openly extensible as well. DAML+OIL supports the declaration of schema information including class and property hierarchies, axiomatic relationships between class expressions, and restrictions on properties. DAML+OIL ontologies can refer to other ontologies permitting the modular construction of extended conceptualisations. The semantics of the language are precisely defined, using description logics as a formal foundation [29].

Our ePerson profile store supports storage of, and access to, profile information in RDF format. It associates that RDF encoding with a corresponding DAML+OIL ontology. An agent can query the store to determine if information is directly available in an ontology that it understands. In the case that a direct match is not available, we anticipate being able to use the emerging semantic web infrastructure to discover ontology mappings. Such mappings can provide at least partial transformations from the native ontology of the profile source to an ontology understood by the profile requestor.

We should add one qualifier here, which we see as an important direction for future research. The ontology approach to agent communication is well suited to reasoning with user information expressed in symbolic terms. For example, a taxonomy of sports, against which to express preference information, is easily expressed in this way. However, non-symbolic or sub-symbolic information will also be required. For example: a numerical score corresponding to the users interest in the given sports category, as extracted by a probabilistic learning algorithm. Existing ontology languages, including DAML+OIL, cannot express the semantics of such scoring schemes. Furthermore, in some cases, the categories themselves may not fit the symbolic, logic-based approach. For example, class-membership might be a numerically weighted membership property, rather than a crisp symbolic assertion. In the longer term, bridging symbolic declarative models with sub-symbolic and probabilistic models will be necessary to achieve a complete and useful user profile representation scheme.

---

[6] November, 2001

Decentralisation

Our framework places the ePerson profile store for a given user directly under the control of that user, and stored towards the network edge. This approach is agnostic as to whether the user runs a profile service directly on a personal machine (or distributed across a set of personal machines), whether they share use of a work-group or community-level server, or whether users subscribe to an independently managed web service offering profile management to many people. However, from the point of view of client agents, they cannot assume that all profiles are stored in the same place and need to treat each user's profile as if it were stored separately.

Network-edge services offer strong advantages. For the user, it gives them more flexibility, allowing them to take as much or as little control of their ePerson data as they wish. In particular, users with a strongly-felt need for maintaining personal privacy may prefer to host the profile service, whereas others would prefer to trust a third party to manage the service on their behalf. A decentralized storage system may also present a less attractive target for hacker attacks. For the system overall, decentralization increases scalability. There is no single point of storage, or access, that might become a bottleneck. It also helps to ensure openness: if anyone can create and run a conformant profile service, then the whole system is not dependent upon the actions of a single supplier (whether commercial or open source).

There are, however, drawbacks to a decentralised approach that must be overcome.

Firstly, in the absence of a single access point for a directory of profile services, there is the problem of service discovery. We can envisage many potential mechanisms for dissemination of profile server addresses. For example: they can be embedded in personal home pages, discovery by web robots, direct transmission via personal email, vCard-like encoding in electronic business cards [30], or embedded as extension attributes in X509v3 certificates. However, as with all such network technologies, we would expect an initial period of uncertainty before the successful formats and mechanisms emerge.

A second difficulty is that some uses of profile information currently expect to have access to centralized population statistics. For example, consider a collaborative filtering approach to recommending information sources [31]. Given a centralised population database from which recommendations are drawn, a recommendation agent could consult the ePerson profile, match that against the population data and make suggestions to the user. However, collecting data centrally is contrary to our goal of keeping users' profile data distinct and under privacy policy control. One of our long-term goals in this research is to develop decentralized solutions to services based on population data. For example, a recommendation agent might traverse a community of distributed profile stores, incrementally building up a statistical picture of the population without duplicating the raw data. Each interaction with the ePerson profile would negotiate access to the profile data under the terms of the user's privacy policy. Ultimately, the entire clustering and analysis process might be fully distributed using appropriate algorithms. See section 0 for more details of ongoing research into distributed social filtering algorithms.

Partial Network Connection

Increasingly, users interact with the information on the web (and their organisational intranets) through a variety of means. These include multiple personal computers (e.g.

one in the office and one at home), PDA's, mobile phones and public Internet cafés. Maintaining continuity and consistency across these different touch points is a frustrating experience for many users [32]. We can identify two core problems for the ePerson framework to address. The first is that as they user moves through their day, performing different tasks in a variety of contexts, they will use different client devices to access the information space on the Internet. We cannot rely, therefore, on storing user information in one client device. Further, we can assume that user profile information will be generated in multiple places, generating a problem of consistency. This scenario will be familiar to anyone who has attempted to maintain consistent sets of web bookmarks on multiple computers. The second problem is that a proportion of the client devices a user has access to will have intermittent connectivity to the network. This will vary by each user and their own situation, but, in general, we cannot assume that we can simply store all profile information in some Internet-based store, and expect the user always to be able to access it.

One approach to ensuring consistency of access to profile data is under investigation in HP Labs by Stephanie Riche et al [10]. In this project, consistency of user profile across multiple client devices is seen as analogous to the general problem of cache consistency. By investigating different usage patterns for profile data (such as whether it tends to group within the profile or be diffuse, or whether profile elements tend to migrate to the user's current interaction point), Riche et al have identified the characteristics of suitable cache consistency and coherence algorithms. Ongoing work is testing the performance of different algorithms under different operating conditions.

The ePerson platform will incorporate results for maintaining profile consistent across different access points as they become available. One other characteristic of the ePerson approach, shared with some other user agent designs, provides additional user benefit when network connections are intermittent. Given enough information, there are scenarios in which the user agent can respond to incoming requests on behalf of the user, when the user is unavailable or busy elsewhere. For example, a calendar-aware agent can answer queries about the user's ability to attend meetings on a give date, based on information from the user profile.

Legacy and External Data Support

The framework as discussed so far calls for all profile data to be accessed as positive ground assertions using the RDF data model with the underlying conceptual models being made explicit by encoding using DAML+OIL. We do not propose, however, that it is appropriate to store all user data within the ePerson profile, nor to store all data natively in RDF. A useful metaphor for the user profile is as a *meta-service*, part of the role of which is to provide an integrated view of user profile data, which may include pointers to data services residing elsewhere, or a transliteration of data stored in other formats.

Many existing data sources could be usefully regarded as part of the user profile, but do not natively use the RDF/DAML+OIL format. Our approach to this is to provide mediators [33], that map those legacy or external data sources to the semantic web format. In our current design, we see this primarily being a one-way access: client agents can request information from the legacy sources via the RDF mediator. However, in some cases, the legacy sources will continue to be updated. In these cases, the ePerson could provide the web service descriptors for the external data sources, that would allow the user profile to be updated directly by the client agent. For example, consider calendar or diary management. One approach would be for the ePerson profile service to provide an ontology for calendar entries, and provide mediators between RDF queries and any native calendar information stores. However, this might not always be practically feasible. Alternatively, the profile service might provide the address and description of the native information stores directly, using a standard web-services interface.

Summary of Design Principles

The ePerson profile store provides a common access protocol for user profile information in RDF format. It allows agents to query for profile information against an explicit ontology, and can potentially use ontology mappings to transform natively stored information into the requested ontology.

The store can be accessed using as a web service, using either remote procedure call (RPC) semantics (e.g. using SOAP [22]), or using stateless protocols, such as HTTP. The choice will be application and environment specific, and will be based on factors such as the expected latency and reliability of the network connection. In either case, each user may be using a different physical store and all of the access agents need to negotiate the terms of access and update.

Issues of how the stores are identified, how the access policies and expressed and enforced and how user authentication is handled without violation of privacy are key issues in the overall framework and are reviewed in the next section.

4 Privacy Protection and Controlled Disclosure

Introduction

Increasingly, individuals, corporations, governments and other commentators are raising concerns about the protection of personal information held online. Many researchers into personal agents and information agents cite user privacy as a concern, yet, to date, little research has been applied to ensuring privacy protection for these technologies. In society generally, and the Internet industry in particular, new technologies and legislative frameworks are being advanced that can enhance privacy (though sometimes the reverse is true). The legal and moral debate about the trade-off between strong privacy and the general well-being and security of society at large will continue. Our goal with the ePerson is more modest: the protection of personal information from unwarranted intrusion, and from the unwanted consequences of such intrusion – for example unsolicited advertising (“spam”).

Recent innovations around e-commerce in digital content (e.g. streaming video on-demand, electronic books, on-line music) compound the privacy problem, thanks to electronic tracking and user authentication for the purposes of copyright protection. These make the task of gathering of personally identifiable information easier, and hence increase the extent of potential privacy violations. Users must have trust that their personal information will be protected by the agent, in order to undertake to disclose it in the first place. For example, if agents are given detailed information to then autonomously carry out tasks on the user’s behalf, the user should be confident when, where and to whom such information will be disclosed – or, at least, that it will only be disclosed appropriately.

In this section, we outline some of the general issues for protecting the privacy of the user of adaptive agents, and discuss some of our solutions in the context of the ePerson.

Privacy and Minimal Information

For excellent material on privacy and its importance for the consumer, the corporation and the state, see the publications of Ann Cavoukian [34;35;36;37]. In particular, a simple and reasonably complete definition of privacy may be found in [38]:

> *"Personal control over the collection, use and disclosure of any recorded information about an identifiable individual"*

For our purposes, we consider privacy to consist of protection of (a) personal identifiers and (b) profiles (recorded information) associated with personal identifiers. The protection of personal identifiers presents a dilemma in the context of personalised services: how to get personalisation, while restricting the disclosure of who you are. Below, we describe various anonymity and other pseudo-anonymity techniques to address this problem. Protecting profile information amounts to controlled revelation of information, with conditions on future use of the disclosed data. In both cases, it is instructive to consider the *minimal information* that is necessary in any given transaction.

The notion of minimal information transfer has been an important principle for privacy in the non-digital world [39]. We consider that it will continue to be important in the digital age. The principle itself is simple to state: any information requested that is not *necessary* to complete the explicit purposes of an interaction with an agent or a service, needs to be called out as such by the requestor while making the request.

However, conforming to a minimal information principle is a difficult goal even in the cases when the requestor is well intentioned. For example, a transaction for the purchase and delivery of a physical good has very well defined minimal required information: (a) electronic payment information (b) delivery destination. On the other hand, minimal information for a gift recommendation is not well defined, and greater information transfer may or may not result in an improved recommendation. As the use of minimal information has been very successful as a good business practice in the non-digital world, it is worth trying to formalize the return to the user for revealing more information, in cases where minimal information cannot be easily determined.

Overall Privacy Approach

In our overall approach, the end user gains direct control over their private information through the application of the following principles:

- *Control of credentials and identifiers for user authentication*

  The degree of user anonymity is determined by the manner in which a user is deemed qualified to participate in an interaction. The qualification process itself reveals information about the individual, and determines the extent to which the information exchanged in a particular interaction can be linked to information produced by the same user in another interaction.

- *Control of which elements of the user profile are revealed and to whom*
  Following the principle of minimal information, the requestor should be able to identify which profile elements are necessary for a transaction, and which are optional to provide an enhanced service. For example if an interaction will result in physical delivery of items, the user's address would be part of the minimal information. Further, a user may be agreeable to the release of the minimal information, or may opt not to participate in the interaction, and ought to be able to make that decision. Lastly, he or she may be willing to release more than the minimal information, and that, too, ought to be possible.
- *Control over what the privacy policy for later use of this data is*
  Privacy is enhanced if the user has control over the conditions under which the credential and profile elements are revealed. For example, may the requestor sell the information to all types of buyers without restriction? To some types of buyers? May the requestor share the information with other business associates? Should the requestor destroy the data once it has been used for the current purpose? For what uses may the information ever be used? Should the information be used only when aggregated with similar information from a certain minimum number of other users? Note that, technically, we can only reconcile the *declared intent* of the requestor to the user's wishes. A separate physical auditing process [40]or technical auditing mechanism will be necessary to determine compliance with stated intent.
- *Control over the accuracy of data revealed*
  Conventional privacy techniques typically involve making a simple decision to either reveal a piece of profile data, or not, or reducing the precision of the data[7]. *Variable privacy* provides an alternative: the data is revealed, but it may or may not be accurate. By subjecting the data to deliberate corruption, according to a defined distribution, the data has reduced utility for making assertions about a specific user. However, it regains value when aggregated with similar data across a large population of users.

The next section describes the privacy-related elements of the ePerson framework, showing how we put these principles into practice.

EPerson Privacy Framework

In this section, we discuss more specifically about the ePerson framework for privacy: identity and authentication, policy engine, trusted aggregators and variable privacy.

Protecting the Identity of the User

In order to gain access to a particular user's profile, a third party (say a web site) needs to indicate to the profile store the *identity* of the user in which they are interested. Thus, the user needs to present some form of *identifier* to the web site, which can be passed on to the profile store and used to locate the user's profile.

In the simplest case, this identifier may be global in scope and persistent over a long period. An example of such an identifier is a username, or an email address. This scheme affords the user little in the way of privacy protection, since the same

[7] The statement "I earn between $20K and $40K per year" is less precise, and better protects privacy, than "I earn $25,600 per year".

identifier is given to each site, allowing long term tracking of the user, and correlation of the user's activities between different sites.

Privacy protection can be improved in several ways.

The user may be permitted to create and manage multiple separate identities, which may or may not reference the same user profile. They now have the choice as to which identity to present to which site. This could be used, for example, to separate work activities from non-work activities, by explicitly creating a different pseudonym for each. This behaviour is visible today in data available on the users of another.com [41], of which 80% are under 25s. This data shows, on average, that males have two active email addresses and females have four.

In general, there is no need for the identifier to be global in scope. The identifier also need not even be persistent over time. As long as the profile store is able to map it back to a specific user, then that is sufficient. However, unless the identifier is persistent over time at a given site, that site will not be able to recognise the user returning, and so personalization options will be more limited. Depending on the privacy preferences of the user, this behaviour may or may not be desirable. The following scenarios expand on these ideas, and afford the user better privacy protection than a persistent global identity.

1. At the start of a session, a user would authenticate them self with the profile store. This results in a random session identifier being allocated to the user, which only the profile store is able to map back to the real identity. Because the identifier changes each session, it cannot be used by the web site to track the user over long periods.
2. The user may have a single, global identity, but that is never used directly. Instead, a client side process generates a different identifier for each site visited. This would prevent users activities being linked between sites. This site-specific identifier would have to have the property that the profile store, and only the profile store, can map it back to the global identity. This is straightforward to achieve using public key cryptography.
3. The combination of (1) and (2).

In the above scenarios, the user authenticates with the profile store directly. However, identity services are starting to emerge which enable people to use a common identity (username and password) across multiple web sites. This avoids the need to register with each site visited, and in particular avoids the need to remember multiple usernames and passwords. These services are also referred to as single sign-on or unified sign-on services. Microsoft's Passport [42] service is the most prevalent example today. However, similar services include AOL's Magic Carpet, currently begin promoted at the screen name service, and in the near future a similar offering from the Liberty Alliance [43].

These third-party identity services also aspire to own the user's profile. However, we feel that a separation between the identity service (responsible for authentication) and the profile service (responsible for managing the user's profile) is logical, and beneficial to the privacy of the user. Thus, our intent is that the profile store not become an identity service in its own right, but instead be flexible enough to work with these third party identity services as they evolve, particularly ones that place more emphasis on privacy protection.

Policy Engine

The policy engine (see Fig. 1) implements fine-grain policy-driven access control over the user profile data. The user can establish a set of policies controlling who may access each part of the profile, and for what purpose. The profile store is organised hierarchically, and policies may be attached at different points in the hierarchy. In general, any given request will be subject to the set of policies encountered as a path is traced from the request point back to the root node. The policies at each level of hierarchy need to be aggregated, and any conflicts that arise must be resolved.

The policies will generally specify that some form of evidence needs to be presented by the requestor before access is granted. The evidence will constitute proof of who, or what, the requestor is. More specifically, the evidence may demonstrate the actual identity of the requestor, or it may demonstrate just that the requestor posses certain relevant attributes. The policy engine compares the evidence provided by the requestor against the policy covering the requested data. If the evidence satisfies all the policy pre-conditions, then access is granted. If insufficient evidence is provided, the request will be denied, and a reason stated. The requestor can then decide whether to supply the additional evidence, or simply abort the request.

Requests to access data from the profile store, and the policies to control that access, will be based on a policy language, some elements of which are outlined below.

- *Request path:* identifies the sub-set of information within the profile to which this policy applies by describing a sub-graph of the profile structure.
- *Request permissions:* specifies the access permissions to the profile data under this policy.
- *Usage policy:* this (machine-readable) policy dictates the terms-and-conditions covering how the requestor may use the profile data disclosed. It is returned to the requestor with the profile data.
- *Identity pre-conditions:* specifies the type of identity disclosure required. Restrictions may be placed on the technology used, and, once validated, the identity may be compared to a list of authorised parties or roles.
- *Attribute pre-conditions:* specifies the type of attribute disclosure required. Restrictions may be placed on the technology used, the specific attributes required and the range of acceptable values.
- *Post processing*. This section specifies what, if any, post processing should be performed on the profile data before returning it. Post-processing allows the profile data to be modified on the fly, for example to implement systematic lying needed to support variable privacy.
- *Audit and logging*. This section specifies what type of audit trail should be left when this policy is used. For example, failure of a requestor to meet all the pre-conditions may result in the user being notified by email.

This policy language shares some design goals with P3P [44]. However, there are some differences: P3P policies do not specify post-processing, auditing, or different ways in which to identify or authenticate requestors. P3P also does not have a means of identifying a hierarchical data structure and using it for specifying access/usage policy. P3P assumes that the data collection happens as a result of a user approaching a web site, while we need a policy language for the purposes of controlling access to data for a variety of purposes.

Attribute Based Evidence from the Requestor

When creating policies controlling access to the profile store, the presence of certain attributes may be more important than the actual identity of the requestor. For example, a user may want to limit access to part of his profile to web sites carrying either the TrustE privacy seal [40] or an acceptable P3P [18;44] encoded privacy policy. To gain access to the profile, the requestor must disclose these attributes, rather than just their identity.

One way of demonstrating attributes is to first demonstrate identity and then show a secure mapping from that identity to a set of attributes. This might be done through a secure directory service. Alternatively, additional X509 certificates could be issued by a certificate authority to the same effect, since X509 certificate extensions can be used to carry attributes. However, we are particularly interested in techniques that allow attributes to be demonstrated without first disclosing identity, since this improves the privacy protection of the requestor by allowing them to remain essentially anonymous. We propose using some form of anonymous or pseudo-anonymous attribute credential.

One scheme developed by members of our group involves user identity comprising a large pool of public/private key pairs, rather than just a single key pair [45]. Each attribute certificate is bound to a different public key drawn from the pool. If the pool is large enough, it then becomes difficult to link different attribute certificates back to the same user.

A more complex, but more powerful, scheme of attribute certificates has been developed by Stefan Brands [46]. This is a generalisation of Brands' research into anonymous digital cash. It is based on protocols that enable the blind issuing of attribute certificates and the selective disclosure of their attributes. Blind issuing prevents the certificate authority (CA) from being able to track the use of the credential, by ensuring nothing in the final credential can be linked back to the CA's view of the issuing protocol. Selective disclosure allows the owner to control which of attributes within the credential are disclosed. This is analogous to being able to erase fields in a paper credential before showing it. Further, it is possible to include additional attributes in the credentials, which act as a strong disincentive to sharing credentials.

Trusted Aggregators

The profile store allows trusted components to act as aggregators, or summarisers, of profile information. These components are authorised to read raw data from the profile, process it in some way, and write back the results to the profile. An example (described in more detail in section 0) is an aggregator that takes raw browser click stream data from the profile, and attempts to map this to a set of "current interests" for the user. The "current interests" part of the profile could be used, for example, by news web sites to provide the user with personalised news. It is unlikely that the user would have permitted these sites access to the raw click stream data.

Variable Privacy through Systematic Lying

For the exchange of usage information at a particular aggregation level in return for discounts and services, one needs a means of ensuring that information collected is indeed averaged. One way of doing this is to reveal incorrect information with a secured probability of truth for binary or multiple-valued information, and a secured variance for continuous-valued information [47]. Because the information is incorrect, it is not worth much by itself, and is valuable only in aggregation. This forces data collectors to aggregate the information. The accuracy of the calculated (i.e. aggregated) statistical information depends on the probability of individual answers being true when they are multiple-valued or binary-valued, or on the variance of the answers provided if they are continuous-valued. More uncertainty in the answers implies that more information has to be aggregated to obtain a particular level of accuracy.

The degree of accuracy affects the value of recommendations and ratings determined using this data. It also affects the value of this data to the data collector. Thus, it provides a manner of making explicit the degree of privacy violation as well as the returns (recommendations, other services) obtained through the privacy violation. It may be used as a substitute in cases where minimal information is not determinable.

5 Application Examples

Here we consider some of the applications that are enabled by the ePerson framework. Some of these we have working prototypes of, while others are in development or at the conceptual design stage.

History Store

The first application example we will describe is called the *personal history store*, and is in use as an experimental prototype in our Bristol research lab. The personal history store collects data from the web pages visited by the user through one or more web browser sessions. It takes its name from the history capability provided by most major browsers, but extends the capability of the standard history listing by collecting additional metadata, including full-text keyword indexes on the pages visited. This enables the user to pose requests such as “I recall that sometime last week, I visited a web site about Java implementations of collaborative filtering algorithms and I want to go back there”. It thus provides a powerful search capability, analogous to search engines such as Google [48], but personalised to the user’s own interaction history. The personal history store comprises a number of functional elements: data collection, history search, and data aggregator, which we outline further below.

Data collection Data collection in our current prototype occurs through inserting a profile-capturing service between the user and the wider web. This is implemented as a web proxy, that is pointed to by the user’s web browser proxy settings. The history store then keeps a record of each web page visited. Many users are quite sensitive about their web browsing patterns. Thus, even though the history store web-proxy is a shared resource, the standard behaviour is to partition the store by user. In principle, the partitions could be encrypted with a user-controlled key, but we are not doing this at present. For each page visited, the title, URL, date, and raw page content are stored. In addition, the page content is converted to plain text and passed to a keyword-indexing package[8].

---

[8] We used Lucene [86], now part of the Apache Jakarta project [87]. Lucene is a high-performance, full-featured text-indexing engine written entirely in Java.

Due to the volume of data maintained, we chose to implement the history store as a profiling service, outside the actual profile itself. Data is available from this service in RDF, but RDF is not used as the storage mechanism.

History search service A web interface provides a search capability, similar to standard web search engines. The search algebra supports access to the meta-data captured by the data collector, Boolean expressions, phrase queries (including wildcard terms), and temporal constraints. Using this interface, users can easily return to pages they have browsed in the past. As the history store keeps cached copies of pages; the original content is still available if the original site is down, or has changed. These cached copies can be aged as necessary to reduce storage overheads.

Trusted aggregator Within the user profile framework we have created a trusted aggregator whose purpose is to mine the history store and attempt to determine a set of current-interests for each user, based on their web-browsing history. The classifier tries to match the page URL against a local copy of the DMOZ [49] database. This database contains 2.7 million URLs organised into over 370,000 topic categories. If a match of the complete URL fails, then the path part of the URL is gradually truncated. If there is still no match, we apply some simple re-writing rules to the host part of the URL. This approach has proven to have a high probability of generating a match. We maintain counts of how many URLs map to each DMOZ topic category. After all of the user's URLs are classified, the trusted aggregator writes any non-empty topic categories and weights to the current-interests section of the ePerson profile. These aggregated categories may then be governed by different privacy rules than the raw data itself. Since its functionality is well defined, the aggregator is trusted by the user, and hence granted privileged access to the make changes to the profile.

In summary, this example illustrates the use of a trusted aggregator to distil highly sensitive private click-stream data into a set of current interests that may be more openly shared.

Web Assistants

The remaining application examples in this section describe illustrative scenarios using the ePerson platform. The intent is to ground the abstract platform descriptions, above, in terms of use cases and scenarios that illustrate how users could benefit from the use of ePerson-based information agents.

One category of information agents that has been widely investigated is the web-browsing assistant. Some examples (among many) include Letizia [50], WebWatcher [51], Personal WebWatcher [52] and WebMate [53]. Generically, the web-browsing assistant can play two roles in relation to the user: a *mediator* for the user's web-browsing experience, or a *benign observer*. The difference being, in essence, whether the user's interaction requests go direct to the agent, which then interacts with resources on the web to satisfy the request, or whether the agent "looks over the user's shoulder" [54] and makes suggestions in a parallel user dialogue. Partly this choice comes down to available technology. Usability studies on personal web-browsing agents have faced difficult choices in the past between using custom, instrumented software, delivering more accurate data, and standard browsers that the users are comfortable with and contain their personal bookmarks. Both the mediator and assistant design concepts have serious design challenges to address, and would make use of the profile store in different ways. For brevity, we consider only the assistant further in this paper.

Assume that the user has browsed to a particular web page. There are various ways in which the web assistant can support the user's information gathering needs. First, we assume a minimum of context is available for the current task from the user profile. The tasks that the agent could perform include:

- Looking ahead to the links on the page, and identifying those that match well against the user's profile of interests, or which score highly in social-filtering measures based on the user's interests. If the agent is able to collaborate with a network-edge proxy service, through which the user's browser retrieves web pages, this identification could be done in-line by highlighting words, inserting additional mark-up or generating links in a browser side-bar. Alternatively, the agent can pick out the hyperlinks in a separate window, perhaps displaying them in rank order [50].
- Identifying sub-sections of the page that match well with the user's interest profile, again using page modification as a means of feedback [55] or generating a section index in a parallel window.
- Auto-generating a site map for the current web site, using profile information to highlight regions of the site that might be of high interest.
- Use feedback from the user, either explicitly or through inferences made on observations such as time spent on a page, to refine the user profile.

With additional information about the user's context, more assistance can be given. For example, the user may choose to indicate to the agent a current goal or role. There are clearly usability issues here: users typically will not want to interrupt the flow of their interaction to engage in extensive consultations with an agent. Our design goal must be to capture as much information as possible for as little cost to the user as possible, and attempt iterative refinement to improve the resolution on the user's need. A good starting point is that users do attempt to express an approximation of their goal to a search engine during an information discovery or retrieval task. Observing or mediating this dialogue would give a good approximation to an initial goal for the context. Given this context, the agent can do a better job at the above tasks, and can provide additional refinements –such as thesaurus or ontology search on terms from the context [56].

An advantage to the user in having their information stored in a network-edge service, such as the ePerson, is that it allows them to move between different web interaction points and maintain a degree of continuity. Many people experience the frustration of attempting to keep sets of bookmarks consistent between a PC in their home and one in their place of work. As the information in the user profiles grows in size and value to the user, maintaining this consistency will only grow in importance to the user, along with the need to keep that information secure and private.

As the information in the user profile accumulates, additional services become possible. Consider a computer science researcher, attempting to keep abreast of developments in his or her field of research. Various tools exist for discovering, say, relevant new articles and papers (e.g. the monitoring option at ResearchIndex [57]). However, such tools never have complete coverage, and still require the researcher to evaluate papers that are discovered to see if they are interesting and truly relevant, and to devote effort to keeping the matching filters up to date. If a profile of the researcher's professional interests was maintained in their ePerson profile store, then, assuming this was permissible under the privacy policy, this data could be used to discover a community of researchers with similar interests. References to papers discovered by one member of the community could be propagated about, sharing the load of the discovery among many peers. Furthermore, the use of *social filtering*[9] could reduce the burden on the user to read and evaluate every resource discovered, by using the ratings assigned by members of the community.

Arguably, it would also be useful if the policy language used to describe the user's preferences regarding the sharing of personal data incorporated constructs that would allow trust networks to be built up. In standard social filtering algorithms [7], every rating has equal weight. In reality, we typically find ourselves placing more trust in particular individuals – so called *opinion leaders* – than we do in others. Social filtering algorithms aim to build robust recommendations from the statistical properties of large populations, but are brittle when the amount of data is limited. A trust network that overlaid the (dynamically discovered) community-of-peers could partly address this limitation, by allowing users to gain robust recommendations from fewer data points.

Distributed Collaborative Filtering

A class of powerful end-user applications, such as collaborative news recommendation, are enabled by the application of social filtering techniques in conjunction with the privacy-enhanced profile store. Social, or collaborative, filtering (henceforth CF) is a term for a broad range of algorithms that use similarity measures between individuals in a population to generate recommendations. The information we collect and store in the user profile provides a rich source of data for such similarity measures. However, as we mentioned in section 0, standard CF algorithms require modification to work with decentralised profile data and privacy protection rules. In this section, we summarise the distributed collaborative filtering problem, and discuss our work on artificial immune system algorithms.

Most CF algorithms work with statistical data. With suitable similarity measures, CF algorithms allow us to directly address the problem introduced earlier, regarding the need to include both mentalist and sub-symbolic models. The simplest approach to social filtering is the *k*-Nearest-Neighbour algorithm, which uses a given similarity method to define a neighbourhood of at most *k* highly similar users. The votes from these users, suitably weighted, are used to make predictions and recommendations. Many improvements on this method have been investigated in the literature (see, among many examples [58] and [31]). Even assuming such improvements however, these methods assume the availability of a centralised collection of profiles to iterate over. In our decentralised framework, we do not make any such assumption.

Viewed abstractly, the task of collaborative filtering can be reformulated as the problem of finding a suitable *neighbourhood* of similar individuals. The metaphor can be further generalised into the task of forming a *community*. The challenge is to provide mechanisms that will allow the build up of an *appropriate* community, the definition of which may be task-dependent. However, we assume no central authority or oracle: it is up to the members of the community to discover themselves.

We have built an experimental recommender prototype to begin to address these issues. More details can be found in [59]. The recommender prototype is based on a novel approach for discovering affinities in a data set called the *artificial immune system* (AIS) [60]. We apply AIS to the problem of community discovery. In brief, we

---

[9] Sometimes called *collaborative* or *clique-based* filtering.

encode a user profile as an antigen, and potential neighbours as antibodies that bind to this antigen. Neighbourhood formation is dynamic and depends on both antigen-antibody and antibody-antibody interactions. The basic elements are as described below.

Data is collected from individual user profiles to create antigens and antibodies. There is a variety of sources for this data, from publicly accessible preference data [61] to our own experimental bookmark managers or the personal history store (above).

Typically, a neighbourhood will gradually build up, with most antibodies leaving almost immediately and only a few persisting. At this early stage, antibody-antigen interactions are dominant. As the neighbourhood grows, antibody-antibody interactions, based on concentration, will have an increasingly noticeable effect.

To achieve distributed community formation, the antigen can be passed around a population. Each ePerson adds an antibody to the evolving AIS, until a sufficiently large neighbourhood has formed. This idea is similar in spirit to the construction algorithm proposed by Delgado [62], and to the idea of a distributed query, as used in Gnutella [63]. However rather than broadcasting the antigen, each individual propagates the antigen via a local narrowcast, based on some rational criteria.

Thus, we can see that a virtual community can be cached, learnt or recalculated as required. In principle, there is no reason why a single ePerson should not belong to multiple communities at the same time. For example, the use of different metrics (film preferences, mailing lists, homepage analysis) may produce different communities. Adamic and Adar [64] show this effect for a university social web. In addition, we have shown that the parameters used to select a community, e.g. the importance of diversity, will provide different community characteristics [59].

6 Related Work

As discussed above, user modeling, personalisation, and adaptation are well-researched topics with a significant body of prior literature. Rather than attempt any complete review of relevant prior work, we here summarise some of the key projects that share at least some goals with our own. This review is illustrative, rather than necessarily complete or representative.

Stuart Soltysiak and Barry Crabtree report the use of user profiling as a key component of a range of personal agents developed at BT Labs [65;66]. Independent agents including Grapevine (interest-based matchmaking in an enterprise), Pandora (proactive topic suggestions) and Radar (information presentation) all share a common user profile. The profile itself is represented as a TFIDF vector [67], which uses the frequency of occurrence of terms as a measure of the user's interest. Terms that occur more frequently than they do in the domain corpus are picked out as representing the user's interests. The TFIDF representation acts as a common representation formalism between different agents, and is quite flexible in that many different sources of information (favourite web pages, saved emails, documents, etc) can be mapped to a single form. It is limited in expressive power however, and cannot, for example, easily hold demographic data or other symbolic information. Soltysiak and Crabtree do acknowledge the importance of maintaining privacy of the user's profile, though the published papers do not detail their approach to this topic. The success the BT group has had in sharing information among different user agents using only the TFIDF representation is a testimony to the utility of sub-symbolic representations, and it is clear that any general profiling scheme will have to encompass both styles of representation.

The use of ontologies in adaptive web navigation has been investigated by Jason Chaffee and Susan Gauch [56;68]. In this work, ontology is taken to refer to the hierarchy of categories used by Lycos to organise their web directory, rather than the direct application of an ontology language such as DAML+OIL. The ontology also includes a selection of characterising web pages for each category, so that it is possible to compare other organisation schemes (such as the user's bookmark hierarchy) to the reference hierarchy. The mapping derived from this comparison can be used to assist translations between the user's "personal ontology" and the system ontology. While this approach is interesting, and the use of characteristic documents to aide the translation process is certainly intriguing, we would hope that using an ontology language with a more formal basis will provide a sounder and richer basis for performing such translations, as well as providing the basis for an extensible representation scheme.

Self-contained user -modelling components or services have been investigated by several authors, though few have enjoyed significant use outside the laboratory where they were developed. Examples include BGP-MS [16], Doppelgänger [69], Tagus [70] and UM [71]. All of these systems pre-date recent developments in open representation systems such as RDF and DAML+OIL, and so do not emphasise extensibility as a design goal.

Some basic user-profile ontologies have been published for DAML+OIL, though at the time of writing these are relatively immature. Such ontologies do not compare to the user-modelling tools outlined above, since they define only the terms used and not the reasoning processes and other supporting services that comprise a complete profiling system. Published DAML+OIL ontologies are available from [72]. An alternative approach was taken by the Customer Profile Exchange (CPExchange) consortium [73]. CPExchange is a custom XML format for representing a range of customer data typically held in enterprise databases and customer relationship management (CRM) tools, with the goal of being able to exchange data between such stores. Despite the use of XML, the CPExchange format suffers from inflexibility and limited expressive power, and has not so far gained widespread use. In addition, it is not freely available but must be licensed from the CPExchange consortium, and is therefore unsuitable for our purposes. It would, however, be technically straightforward to map assertions from our DAML-based ontology to the CPExchange format and vice-versa.

A more complete and grounded use of ontologies to assist user interaction is given by Weal *et al* [74]. Here, the emphasis is on the use of ontological descriptions to generate links in a hypermedia space dynamically. Although the system is based on an agent framework, this is not apparent in the user's interface. Instead, agents are used as infrastructure to provide the dynamic behaviour of the system. The ontological information is held in steady state while the links provided to the user are dynamically generated. In our approach, we gather new information related to ontological categories, and new categories, as the user interacts with the system and performs information gathering and management tasks.

Privacy protection in the ePerson framework may be expressed in terms of policy expression, protocols and policy compliance – similar to the use of these terms in [75;76].

The policies that govern the use of collected data and associated credentials need to be expressed in machine-readable form. It is clear that the language used for the privacy policies needs to be open and extensible. At the moment, there are two candidate languages: W3C's Platform for Privacy Preferences (P3P) [18] and Zero Knowledge System's Privacy Rights Management Language (PRML) [77]. P3P is an XML-based specification and is tailored specifically for the collection of regular consumer information by websites. So, for example, it does not include detailed handling of usage profiles. W3C has an associated specification for a language, APPEL [78], meant for specifying client-side privacy preferences incorporated into the browser. When a consumer visits a website, the browser indicates whether the site has a P3P policy, and whether the terms of the policy match the preferences set by the consumer. Zero Knowledge Systems has not made public details on PRML, so not much is known about it. PRML is expected to address the expression of policies associated with data collection laws in various countries. If this is all it addresses, it too will not be sufficient for our needs. If, however, it is extensible, it might be possible to extend it for our purposes.

Changing the accuracy of collected data for the purpose of maintaining privacy has been described in the statistical database [79;80;81;82] and the data mining [83] literature. In these fields, it has largely been used with continuous-valued data, and the experiments reported were performed on accurately collected data which is perturbed after collection. The work on continuous-valued data demonstrates that considerable statistical information can be gleaned with reasonable accuracy from the perturbed individual pieces of information. In our system, we plan to use the technique for both continuous and discrete-valued data, and propose that the data be perturbed by the user at the time of collection, with a user-specified degree of perturbation. The results from the data mining and statistical database literature are consistent with our expectations that our system will enable the user-controlled release of single units of information with varying degrees of privacy, such that the information may be used in different ways, for different purposes, through dynamic aggregation.

In some respects, Microsoft's ".NET My Services" [84] component of the .NET architecture shares similar goals to the ePerson platform. .Net My Services relies on a single, unique user identifier authenticated by Microsoft Passport, a variant of the Kerberos protocol. Users authenticate once to Passport, and are then able to authenticate themselves to Passport enabled web sites. .NET My Services has an extensible data model for profile information, based on XML. However, while it is clear that a given application or service can store information in the .NET My Services profile, it is hard to see how this data can be shared between applications without some additional support for ontology description. The Liberty Alliance [43] has a goal of providing an identity service similar to that provided by Passport, but is positioned to embrace open standards and the federation of independent authentication services. However, at the time of writing the detailed work programme for the Liberty Alliance project is not available.

7 Conclusions

In this paper, we have outlined an approach to providing general, open information agent services to the user based on a collection of key enabling technologies. Many of these approaches have been investigated in isolation; we propose that together they constitute a novel basis for the development of robust, usable agents. In part, this robustness arises from the use of emerging standards and technologies for next-generation Internet applications: RDF, DAML+OIL and extensible, interoperable web services. In part, it arises from driving the basic requirements for an agent framework from user priorities – hence our emphasis on privacy and user control.

The ePerson platform brings these technologies together as a coherent platform. It is also, however, a key abstraction in the design of larger scale information handling and processing applications. As an element of the design of such applications, the ePerson is the locus of user identity, profile information and preferences. It also provides a platform for autonomous information-based processes, such as web-browsing assistants, that ultimately will assist our users to become more effective in their information handling tasks. Ongoing research work in our laboratory is focussed on generating practical end-user applications based on the ePerson framework.

Acknowledgements. The authors would like to thank colleagues at HP Laboratories, discussions with whom have helped shape the ePerson agenda. Particularly we would like to thank Joe Pato, Gavin Brebner, Stephanie Riche, Martin Griss and Bernard Burg. Thanks also to Martin Sadler and Andy Seaborne for helpful comments on earlier drafts of the paper.

References

1. Pretschner, A and Gauch, S. *Personalization on the Web*. (ITTC-FY2000-TR-13591-01) Information and Telecommunication Technology Center, University of Kansas. 1999. Available from: `http://homer.ittc.ukans.edu/website/publications/papers/pers-tr.pdf`
2. Fink J. & Kobsa A. "A Review and Analysis of Commercial User Modeling Servers for Personalization on the World Wide Web". *User Modeling and User Adapted Interaction*. Vol. 11. Kluwer, 2001. pp. 209–249.
3. Crabtree I. B., Soltysiak S. J. & Thint M. P. "Adaptive Personal Agents". *Personal Technologies*. Vol. 2:3. 1998. pp. 141–151.
Available from: `http://www.bt.com/bttj/vol16no3/13.htm`
4. Franklin, S. & Graesser, A. "Is It an Agent or Just a Program? A Taxonomy for Autonomous Agents". In: Muller, J. P., Wooldridge, M., & Jennings, N. (eds) *Intelligent Agents III Proceedings of Agent Theories, Architectures and Languages (ATAL 96).* Springer, 1996. pp. 21–36.
5. Dickinson, Ian. *The interface as agent: a comparative review of human-agent interaction*. HP Labs Technical Report. In preparation.
6. User Modelling Inc. *User Modelling Inc Home Page*. 2001.
Web site: `http://www.um.org`
7. Pazzani M. "A Framework for Collaborative, Content-Based and Demographic Filtering". *Artificial Intelligence Review*. Vol. 13:5 - 6. 1999. pp. 393–408.
8. Webb G., Pazzani M. & Billsus D. "Machine Learning for User Modeling". *User Modeling and User Adapted Interaction*. Vol. 11. Kluwer, 2001. pp. 19–29.
9. Kobsa A. "Generic User Modeling Systems". *User Modeling and User Adapted Interaction*. Vol. 11. 2001. pp. 49–63.
10. Riché, Stephanie, Brebner, Gavin, and Gittler, Mickey. *Client-side profile storage: a means to put the user in control*. (in preparation) HP Laboratories. 2001.

11. Kindberg, Tim and Barton, John. *A web-based nomadic computing experience*. (HPL-2000-110) HP Laboratories. 2000. Available from: http://www.hpl.hp.com/techreports/2000/HPL-2000-110.html
12. Chen, Harry and Tolia, Sovrin. *Steps towards creating a context-aware software agent system*. (HPL-2001-231) Hewlett-Packard Laboratories. 2001. Available from: http://www.hpl.hp.com/techreports/2001/HPL-2001-231.html
13. Butler, Mark. DELI: *a delivery context library for CC/PP and UAProf* . (HPL-2001-260) HP Laboratories. 2001. Available from: http://www.hpl.hp.com/techreports/2001/HPL-2001-260.html
14. Butler, Mark. *Current technologies for device independence*. (HPL-2001-93) HP Laboratories. 2001. Available from: http://www.hpl.hp.com/techreports/2001/HPL-2001-83.html
15. Barton, John and Kindberg, Tim. *The Cooltown user experience*. (HPL-2001-22) HP Laboratories. 2001. Available from: http://www.hpl.hp.com/techreports/2001/HPL-2001-22.html
16. Pohl W. "Logic-Based Representation and Reasoning for User Modelling Shell Systems". *User Modeling and User Adapted Interaction*. Vol. 9:3. Kluwer, 1999. pp. 217–282.
17. *Net Perceptions*. 2001. Web site: http://www.netperceptions.com
18. World Wide Web Consortium (W3C). *Platform for Privacy Preferences Project (P3P)*. 2001. Web site: http://www.w3.org/P3P/
19. *HP Web Services*. 2001. Web site: http://www.e-speak.hp.com
20. *IBM Web Services*. 2001. Web site: http://www.ibm.com/software/solutions/webservices/
21. *Microsoft Hailstorm*. 2001. Web site: http://www.microsoft.com/presspass/features/2001/mar01/03-19hailstorm.asp
22. World Wide Web Consortium. *XML Protocol*. 2001. Web site: http://www.w3.org/2000/xp
23. *UDDI – Universal Description Discover and Integration*. 2000. Web site: http://www.uddi.org
24. W3C. *W3C Semantic Web Activity*. 2001. Web site: http://www.w3.org/2001/sw/
25. Berners-Lee, Tim, Hendler, James, and Lassila, Ora "The Semantic Web". *Scientific American*. 2001.
26. World Wide Web Consortium (W3C). *The Resource Description Framework (RDF)*. 1999. Web site: http://www.w3.org/TR/REC-rdf-syntax/
27. Berners-Lee, Tim, Fielding, and Masinter, L. *Uniform Resource Identifiers (URI): Generic Syntax*. (RFC2396) Internet Draft Standard. 1998.
28. Chandrasekaran B., Josephson J. R. & Benjamins V. R. "What Are Ontologies, and Why Do We Need Them?". *IEEE Intelligent Systems*. Vol. 14:1. 1999. pp. 20–26.
29. van Harmelen, Frank, Patel-Schneider, Peter, and Horrocks, Ian. *A Model-Theoretic Semantics for DAML+OIL* (March 2001) Revision 4.1. 2001. Available from: http://www.daml.org/2001/03/model-theoretic-semantics.html
30. Internet Mail Consortium. *vCard White Paper*. 1997. Web site: http://www.imc.org/pdi/vcardwhite.html
31. Good, N., Schafer, J. B., Konstan, J., Borchers, A., Sarwar, B., Herlocker, J., & Riedl, J. "Combining Collaborative Filtering With Personal Agents for Better Recommendations". In: *Proc. 16th National Conference on AI (AAAI-99)*. AAAI Press, 1999. pp. 439–446.
32. Brown, Barry and Sellen, Abigail. *Exploring users' experiences of the web*. (HPL-2001-262) HP Labs Technical Report. 2001. Available from: http://www.hpl.hp.com/techreports/2001/HPL-2001-262.html
33. Wiederhold G. "Mediators in the Architecture of Future Information Systems". *Computer*. Vol. 25:3. 1992. pp. 38–49.

34. Cavoukian A, Tapscott D. *Who knows: safeguarding your privacy in a networked world* McGraw-Hill, 1996.
35. Cavoukian, Ann. *Privacy: The Key to Electronic Commerce*. The Information and Privacy Commissioner/Ontario . Available from: `http://www.ipc.on.ca/english/pubpres/papers/e-comm.htm`
36. Cavoukian, Ann. *Data Mining: Staking a Claim on Your Privacy*. Information and Privacy Commisioner/Ontario. 1998. Available from: `http://www.ipc.on.ca/english/pubpres/papers/datamine.htm`
37. *Privacy as a Fundamental Human Right vs. an Economic Right: an Attempt at Conciliation* . Information and Privacy Commissioner/Ontario. 1999. Available from: `http://www.ipc.on.ca/english/pubpres/papers/pr-right.htm`
38. Cavoukian, Ann and Gurski, Mike "Managing Privacy: a Challenge in Designing Today's Systems". *Municipal Interface*. 2000. Available from: `http://www.ipc.on.ca/english/pubpres/ext-pub/misa-00.pdf`
39. Wright, Tom. *Privacy Protection Makes Good Business Sense*. 1994. Available from: http://www.ipc.on.ca/english/pubpres/papers/busi-e.htm
40. TRUSTe. *Building a framework for global trust*. 2001. Web site: `http://www.truste.org`
41. Bowbrick, S. "Multiple Identities in the Online World". In: *Proc. First Digital Identity Workshop.* hyperion.co.uk, 2000. Available from: `http://www.consult.hyperion.co.uk/PDFlibrary/y2000/bowbrick.pdf`
42. *Microsoft .NET Passport*. 2001 . Web site: `http://www.passport.com`
43. *The Liberty Alliance Project*. 2001. Web site: `http://www.projectliberty.org`
44. World Wide Web Consortium (W3C). *The Platform for Privacy Preferences Project (P3P) 1.0 Specificatio*n. 2001. Web site: `http://www.w3.org/TR/p3p`
45. Banks, David and Reynolds, David. *Method and System for Controlling the On-Line Supply of Digital Products or the Access to On-Line Services*. Patent application (UK 0103984.1). Filed 17-Feb.-2001.
46. Brands S. *Rethinking Public Key Infrastructures and Digital Certificates; Building in Privacy* ISBN 0-262-02491-8. MIT Press, 2000.
47. Vora, Poorvi , Vazirani, Umesh, and Knapp, Verna. *Probabilistic Privacy Protection*. Patent applied for. (USA 10004309-1). Filed Dec.-2000.
48. Google.com. *Google search engine*. 2001. Web site: `http://www.google.com`
49. ODP - *The Open Directory Project*. Web site: `http://www.dmoz.org`
50. Lieberman, H. "Letizia: an Agent That Assists Web Browsing". In: *Proc. Internation Joint Conference on Artificial Intelligence (IJCAI '95)*. Morgan Kaufmann, 1995. pp. 924–929.
51. Joachims, T., Freitag, D., & Mitchell, T. "WebWatcher: a Tour Guide for the World Wide Web". In: *Proc. 15th International Joint Conference on AI (IJCAI 97*). Morgan-Kaufmann, 1997. pp. 770–775.
52. Mladenic D. "Text-Learning and Related Intelligent Agents: a Survey". *IEEE Intelligent Systems*. Vol. 14:4. IEEE, 1999. pp. 44–54.
53. Chen, L. & Sycara, K. "WebMate - a Personal Agent for Browsing and Searching". In: *Proc. 2nd International Conf on Autonomous Agents*. ACM, 1998. pp. 132–139.
54. Maes P. "Agents That Reduce Work and Information Overload". *Communications of the ACM*. Vol. 37:7. ACM, 1994. pp. 31–40.
55. Anderson, C., Domingos, P., & Weld, D. "Web Site Personalizers for Mobile Devices". In: Anand, S. S. & Mobasher, B. (Co-chairs) Proc. *IJCAI 2001 Workshop on Intelligent Techniques for Web Personalization*. IJCAI Inc, 2001. pp. 47–52.
56. Pretschner, A. & Gauch, S. "Ontology Based Personalized Search". In*: Proceedings 11th International Conference on Tools with Artificial Intelligence. TAI 99*. IEEE Computer Society. pp. 391-398.

57. NEC Research Institute. *ResearchIndex - the NECI Scientific Literature Digital Library.* 2001. Web site: `http://www.researchindex.com`
58. Aggarwal, C. C. & Yu, P. S. "On Text Mining Techniques for Personalization ". In*: Proc. 7th Internaltional Workshop on New Directions in Rough Sets, Data Mining and Granular Soft Computing RSDMGrC'99*. Springer-Verlag, 1999. pp. 12–18.
59. Cayzer, Steve and Aickelin, Uwe. *A Recommender System based on the Immune Network.* (in preparation) 2001.
60. Dasgupta D. *Artificial Immune Systems and Their Applications* Springer-Verlag, 1998.
61. Compaq Systems Research Centre. *EachMovie Collaborative Filtering Data Set*. Web site: `http://www.research.compaq.com/SRC/eachmovie/`
62. Delgado J. & Ishii N. "Multi-Agent Learning in Recommender Systems For Information Filtering on the Internet ". *International Journal of Cooperative Information Systems*. Vol. 10:1-2. 2001. pp. 81–100.
63. Gnutella. Web site: `http://gnutella.wego.com`
64. Adamic, Lada and Adar, Eytan. "Friends and Neighbors on the Web " *unpublished manuscript.* 2000 Available from: `http://www.hpl.hp.com/shl/people/eytan/fandn.html`
65. Soltysiak S. J. & Crabtree I. B. "Automatic Learning of User Profiles - Towards the Personalisation of Agent Services". *BT Technology Journal*. Vol. 16:3. 1998. pp. 110–117. Available from: `http://www.bt.com/bttj/vol16no3/13.htm`
66. Soltysiak, S. J. & Crabtree, B. "Knowing Me, Knowing You: Practical Issues in the Personalisation of Agent Technology". In: *Proc. Third International Conf. on the Practical Applications of Intelligent Agents and Multi-Agent Technology (PAAM98).* 1998. pp. 467–484.
67. Salton T. & Buckley C. "Term-Weighting Approaches in Automated Text Retrieval". *Information Processing & Management*. Vol. 24:5. 1988. pp. 513–23.
68. Chaffee, J. & Gauch, S. "Personal Ontologies for Web Navigation". In*: Proc. 9th International Conf. on Information and Knowledge Management (CIKM'00).* 2000. pp. 227–234. Available from: `http://homer.ittc.ukans.edu/website/publications/papers/chaffeeCIKM2000.pdf`
69. Orwant J. "Heterogenous Learning in the Dopplegänger User Modeling System". *User modeling and user adapted interaction*. Vol. 4:2. 1994-1995. pp. 107–130.
70. Paiva A. & Self J. "TAGUS - a User and Learner Modeling Workbench". *User Modeling and User Adapted Interaction*. Vol. 4:3. 1995. pp. 197–226.
71. Kay, J. "Vive La Difference! Individualised Interaction With Users". In*: Proc. International Joint Conference on AI (IJCAI)* 1995. Morgan-Kaufmann, 1995. pp. 978-84.
72. *The DARPA Agent Markup Language Homepage*. 2001. Web site: `http://www.daml.org`
73. *CPExchange home page.* 2000. Web site: `http://www.cpexchange.org`
74. Weal, Mark, Hughes, Gareth, Millard, David, and Moreau, Luc. *Open Hypermedia as a Navigational Interface to Ontological Information Spaces*. (ECSTR-IAM01-004) University of Southampton. 2001. Available from: http://www.bib.ecs.soton.ac.uk/records/5161
75. Erickson, J. , Williamson, M., Reynolds, D., Vora, P., & Rodgers, P. "Principles for Standardization and Interoperability in Web-Based Digital Rights Management*". In: Proc. W3C Workshop on Digital Rights Management (DRM).* Jan 2001: Available from: `http://www.w3.org/2000/12/drm-ws/pp/hp-erickson.html`
76. Vora, P., Reynolds, D., Dickinson, I., Erickson, J., & Banks, D. "Privacy and Digital Rights Management". In: *Proc. W3C Workshop on Digital Rights Management (DRM).* 2001. Available from: `http://www.w3.org/2000/12/drm-ws/pp/hp-poorvi2.html`

77. Zero Knowledge Systems. *Enterprise Privacy Manager*. 2001. Web site: http://www.zeroknowledge.com/business/privacyrights.asp
78. World Wide Web Consortium (W3C). *A P3P Prefenence Exchange Language (APPEL) 1.0 Working Draft*. 2001. Web site: http://www.w3.org/TR/P3P-preferences.html
79. Duncan G. T. & Mukherjee S. "Optimal Disclosure Limitation Strategy in Statistical Databases: Deterring Tracker Attacks Through Additive Noise". *Journal of the American Statistical Association*. Vol. 95:451. 1998. pp. 720–729. Available from: http://duncan.heinz.cmu.edu/GeorgeWeb/OptimalDisclosure Limitation 98 June 16.htm
80. Beck L. L. "A Security Mechanism for Statistical Databases". *ACM Transactions On Database Systems*. Vol. 5:3. 1980. pp. 316–318.
81. Schlorer J. "Security of Statistical Databases: Multidimensional Transformation". *ACM Transactions On Database Systems*. Vol. 6:1. 1981. pp. 95–112. Available from: http://www.acm.org/pubs/articles/journals/tods/1981-6-1/p95-schlorer/p95-schlorer.pdf
82. Adam N. R. & Wortmann J. C. "Security-Control Methods for Statistical Databases: a Comparative Study". *ACM Computing Surveys*. Vol. 21:4. 1989. pp. 515–556.
83. Agrawal, R. & Srikant, R. "Privacy-Preserving Data Mining". In: *Proc. of the 2000 ACM SIGMOD Conference on Management of Data*. ACM, 2000. pp. 439–450.
84. Microsoft. *Introducing .NET My Services*. 2001. Web site: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/Dndotnet/html/ Myservintro.asp?frame=true
85. Rosheisen, R. M. *A Network-Centric Design for Relationship-Based Rights Management (PhD Thesis)*. Stanford University. 1997.
86. *Jakarta Lucene*. 2001. Web site: http://jakarta.apache.org/lucene
87. *The Apache Jakarta Project*. 2001. Web site: http://jakarta.apache.org

Implicit Culture for Information Agents

Enrico Blanzieri[1,2] and Paolo Giorgini[3]

[1] ITC-irst Trento - Italy
[2] Department of Psychology - University of Turin - Italy
[3] Department of Information and Communication Technology
University of Trento - Italy - paolo.giorgini@dit.unitn.it

Abstract. Earlier work introduced the concept of Implicit Culture and its use in multi-agent systems. Implicit culture support can be seen as a generalization of Collaborative Filtering and it can improve agents' performances. In this paper, we present an implementation of a System for Implicit Culture Support, results obtained in a recommendation-problem domain, and an application to the eCulture Brokering System, a multi-agent system aimed to mediate the access to cultural information.

1 Introduction

Given the problem of information overload, the development of information agents is a mayor issue [11]. On the other hand, Collaborative Filtering demonstrated to be an effective solution to the problem of information overload, specifically for products recommendation. In [1], we have introduced the notion of Implicit Culture, in which the application of the collaborative filtering approach to agent-based systems extends the performances of information agents.

Systems for Implicit Culture Support (SICS in the following) have the goal of estabilishing an Implicit Culture phenomenon. Informally, Implicit Culture is the relation existing between a set and a group of agents such that the elements of the set behave according to the culture of the group. Supporting Implicit Culture is effective in solving the problem of improving the performances of agents acting in an environment where more-skilled agents are active. This concept can help the analysis of existing systems and suggest ideas for the synthesis of new ones. In fact, support of Implicit Culture can be useful for various applications: Computer Supported Collaborative Work, Group profiling, Cognitive modelling of social phenomena, e-books, Computer Mediated Communication Systems and, as we briefly present in the following, generalization of collaborative filtering, Knowledge Managment and requirements and interaction control of agent-based systems.

Collaborative filtering [8,9,12,17] is a popular technique exploited in recommendation systems. The goal is information filtering, namely to extract from a usually long list of items, e.g. links or products, a little set that the user could prefer. Collaborative filtering exploits correlations in the pattern of preferences expressed actively o passively by other users in terms of ratings. Differently from the content-based filtering, collaborative filtering does not rely on the content or

M. Klusch et al. (Eds.): Intelligent Information Agents, LNAI 2586, pp. 152–164, 2003.

shape of objects. The central idea is to automate the process of recommending items to a user on the base of the opinions of people with similar preferences. As suggested by Blanzieri and Giorgini [1] collaborative filtering can be seen as a System for Implicit Culture Support (SICS).

In Knowledge Management, generally knowledge is categorized as being either codified (explicit) or tacit (implicit). Knowledge is said being explicit when it is possible to describe and share it among people through documents and/or information bases. Knowledge is said being implicit when it is embodied in the capabilities and abilities of the members of a group of people. In [14], knowledge creation processes have been characterized in terms of tacit and explicit knowledge transformation processes, in which, instead of considering new knowledge as something that is added to the previous, they conceive it as something that transforms it. Implicit Culture can be applied successfully in this context. In particular, the idea is to build systems able to capture implicit knowledge, but instead of sharing it among people, change the environment in order to make new people behave in accordance with this knowledge.

Advantages of Implicit Culture support has been proposed also for artificial agents [6,2], in particular for controlling the requirements of agent-based systems and supporting their interaction. Autonomy of agents, unknown properties of the environment and insertion of new agents do not allow to foresee completely the multi-agent system's behavior in the modeling phase. As a consequence, the overall system can fail to fulfill the desired requirements. In particular, requirements should persist after a composition changing of the group of agents, that is the new agents should act consistently with the culture of the group. Using SICSs, it is possible to modify the view that the agents have of the environment and, consequently, change the set of possible actions that the agents can perform in the environment. Working on the possible actions, a SICS is able to lead the new agents to act consistently with the behavior of the group.

The architecture of SICS proposed in [1,6] relies on the exploitation of learning techniques. In particular it is possible to identify two learning problems: (i) induction of a cultural theory on the behavior patterns of the group and (ii) prediction of a scene such that the elements of the set will behave consistently with the cultural theory. The first problem can be solved by standard data mining techniques. In this paper we present an implemented SICS that solves the second problem exploiting an original generalization of a memory-based Collaborative Filtering algorithm. The SICS is applied to two different information access problems.

The paper is organized as follows. Section 2 presents the SICS architecture and the learning problems. In Section 3 we show a solution to the problem of prediction of the scene and in Sections 4 and 5 we present some experimental results and a real-world application, respectively. The final section hosts conclusions and future work.

2 Systems Implicit Culture Support

The goal of a SICS is to establish an implicit culture phenomenon. In the following, we informally introduce the notions of implicit culture and implicit culture phenomenon (Appendix A reports the formal definitions given in [2]). The second part of the section presents the general architecture of a SICS and shows how it relies on learning techniques.

An Implicit Culture phenomenon is a pair composed by a set and a group of agents such that the elements of the set behave according to the culture of the group and Implicit Culture is the relation between the elements of the pair. The definitions are expressed in terms of expected situated actions and cultural constraint theories. Some assumptions underlie these concepts.

We assume that the agents perform situated actions. Agents perceive and act in an environment composed of objects and other agents. In this perspective, agents are objects that are able to perceive, act and, as a consequence of perception, know. Before executing an action, an agent faces a scene formed by a part of an environment composed of objects and agents. Hence, an agent executes an action in a given situation, namely the agent and the scene at a given time. After a situated action has been executed, the agent faces a new scene. At a given time the new scene depends on the environment and on the situated executed actions.

Another assumption states that the expected situated actions of the agents can be described by a cultural constraint theory. The action that an agent executes depends on its private states and, in general, it is not deteministically predictable with the information available externally. Rather, we assume that it can be characterized in terms of probability and expectations. Given a group of agents we suppose that there exists a theory about their expected situated actions. Such a theory can capture knowledge and skills of the agents about the environment and so it can be considered a cultural constraint of the group.

We call Implicit Culture a relation between a set of agents G' and a group G such that the agents of G' perform actions that satisfy a cultural constraint for G. When a set and a group of agents are in Implicit Culture relation, we have an Implicit Culture phenomenon. The definitions do not require the empirical validation of the cultural constraint theory against the executed actions of G.

The definition of Implicit Culture (Appendix A briefly discusses the use of "implicit") does not give sufficient conditions for its realization, posing the problem of its support in practise. The general architecture of a SICS proposed in [1] allows to achieve the goal of estabilishing an implicit culture phenomenon following two steps. First, the elaboration of a cultural constraint theory $\mathbf{\Sigma}$ from a given domain and a set of situated executed actions of a group G. Second, the proposal to a group G' of a set of scenes such that the expected situated actions of the set of agents G' satisfies $\mathbf{\Sigma}$. Both the steps present learning problems.

A general SICS (see Figure 1-a) consists of three components: observer, inductive module and composer. The observer stores the situated executed actions of a group of agents G in oder to make them available for the other components. The inductive module uses these actions to produce a cultural constraint theory

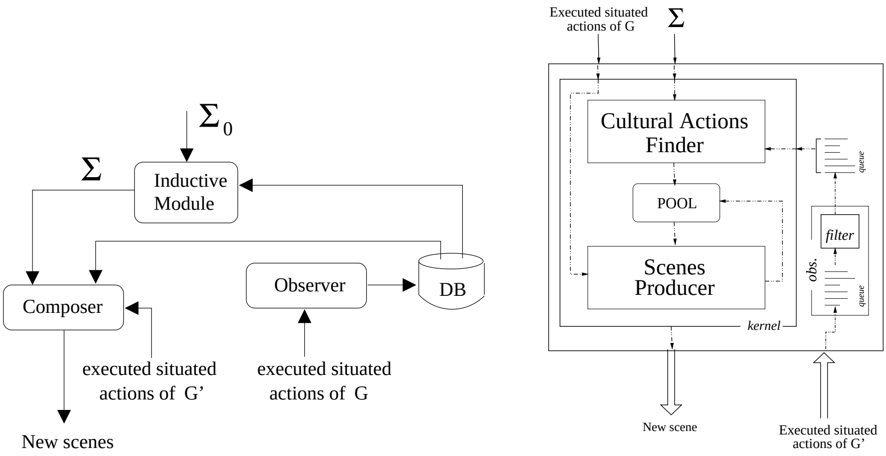

Fig. 1. Architecture (a) The composer in detail(b)

$\mathbf{\Sigma}$ for G. Finally, the composer, using the theory $\mathbf{\Sigma}$ and the actions, manipulates the scenes faced by a set of agents G' in such a way that their expected situated actions are cultural action w.r.t G. As a result, the agents of G' executes (on average) cultural actions w.r.t G (implicit culture phenomenon).

In Figure 1-a the composer proposes to the agents a, b, and c the scenes σ_{t+1}, σ'_{t+1}, and σ''_{t+1}, respectively. Notice that in this case the agents b and c belong to both G and G'. This means that also their situated actions are stored in DB and thus they are used to elaborate the theory $\mathbf{\Sigma}$ and the new scenes.

In general, our implemented architecture accepts cultural theories expressed by a set of rules of the form:

$$A_1 \wedge \cdots \wedge A_n \rightarrow C_1 \wedge \cdots \wedge C_m$$

in which $A_1 \wedge \cdots \wedge A_n$ is said antecedent and $C_1 \wedge \cdots \wedge C_m$ consequent. The idea is to express that "if in the past the antecedent has happened, then there exists in the future some scenes in which the consequent will happen". Antecedent and consequent are conjunctions of atoms, namely two types of predicates: observations on an agent and conditions on times. For instance, $\texttt{request(x,y,s,t}_1\texttt{)}$ is a predicate of the first type, that says that the agent $\texttt{x}$ requests the agent $\texttt{y}$ for the service $\texttt{s}$ at time $\texttt{t}_1$; whereas $\texttt{less(t}_1\texttt{,t}_2\texttt{)}$ is an example of the second type and it simply says that $\texttt{t}_1 < \texttt{t}_2$.

The composer proposes to a set of agents G' a set of scenes such that their expected situated actions satisfy a cultural constraint theory $\mathbf{\Sigma}$ for a group G. The main idea is splitting the problem in two sub-problems: (i) find the cultural actions and (ii) find the scenes where such actions are the expected situated actions. Figure 1-b shows the composer in detail. Basically, the composer consists of two main submodules and an additional component:

– the *Cultural Actions Finder* (CAF), that takes as inputs the theory $\mathbf{\Sigma}$ and the executed situated actions of G', and produces as output the cultural

actions w.r.t. G (namely, the actions that satisfy $\mathbf{\Sigma}$). The CAF matches the executed situated actions of G' with the antecedents of the rules of $\mathbf{\Sigma}$. If it finds an action that satisfies the antecedent of a rule, then it takes the consequent of the rule as a cultural action.
- the *Scenes Producer* (SP), that takes one of the cultural action produced by the CAF and, using the executed situated actions of G, produces scenes such the expected situated action is the cultural action.
- the *Pool*, an additional component, which manages the cultural actions given as input from the satisfaction submodule. It stores, updates, and retrieves the cultural actions, and solves possible conflicts among them.

The SICS architecture requires the solution of two learning problems. A problem of induction of the cultural constraint theory (Inductive Module) and a problem of prediction of scenes (Scenes Producer).

Inductive Module Problem. Given a set of situated executed actions performed by the agents of G, find a cultural constraint theory.

Scene Producer Problem. Given a set of situated executed actions of the agents of G and G', and given a cultural action α for the agent x, find a scene s such that the expected situated action of x in the scene s is α.

The Inductive Module Problem is a rather standard learning problem: inducing the behavior patterns of a group and it is possible to solve using standard data mining techniques. As we previously noted, we do not require any specific validation of the theory. Obviously, very different effects will be reached depending on the fact that the theory is validated or not. In the Scene Producer Problem the request is on the effectiveness of the scene w.r.t the goal of producing the execution of a given action, namely its *persuasiveness.*

3 A Solution to the Scene Producer Problem

The solution exploits the principles of instance-based learning (namely, memory-based or lazy). Given a cultural action α for the agent x that performed actions on the set of scenes $S(x)$, the algorithm used in the scenes producer consists of three steps:

1. find a set of agents Q that performed actions similar to α;
2. select a set of agents $Q' \subseteq Q$ similar to x and the set of scenes S in which they performed actions;
3. select and propose to x a scene of S.

Figure 2 shows the algorithm used in step 1. An agent y is added to the set Q if the similarity $sim(\beta_y, \alpha)$ between at least one of its situated executed actions β_y and α is greater than the minimum similarity threshold T_{min}. The scenes s in which the β_y actions have been executed are added to $S(y)$, that is the set of scenes in which y has performed actions similar to α. Sections 4 and 5 contain two examples of similarity function (eq. 5 and 7).

```
for all y ∈ G'
    for all situated executed actions β_y of y
        if sim(β_y, α) > T_min then {
            if y ∉ Q then  y → Q
            s → S(y)
        }
```

Fig. 2. The algorithm for step 1

Step 2 selects in Q the k nearest neighbors to x with respect to the agent similarity defined as follows:

$$w_{x,y} = \frac{1}{|S_{xy}|} \sum_{s \in S_{xy}} \frac{1}{N_x(s) N_y(s)} \sum_{\beta_x \in N_x(s)} \sum_{\beta_y \in N_y(s)} sim(\beta_x, \beta_y) \quad (1)$$

where $S_{xy} = S(x) \cap S(y)$ is the set of scenes in which both x and y have executed at least an action. $N_x(s)$ and $N_y(s)$ are the set of actions that x and y have respectively performed in the scene s. Eq. 1 can be replaced by a domain-dependent agent similarity function if needed (e.g., Eq. 4 in Section 4).

Step 3 selects the scenes in which the cultural action is the expected situated action. To do this, firstly we estimate for any scene $s \in S = \bigcup_{y \in Q} S(y)$ the similarity value between expected action and cultural action, and then we select the scene with the maximum value. The function to be maximized is the expected value $E(sim(\beta_x, \alpha)|s)$, where β_x is the action performed by the agent x, α is the cultural action, and $s \in S$ is the scene in which β_x is situated. The following estimate is used:

$$\hat{E}\left(sim(\beta_x, \alpha)|s\right) = \frac{\sum_{u \in Q'} \hat{E}_1 \left(sim(\beta_u, \alpha)|s\right) * w_{x,u}}{\sum_{u \in Q'} w_{x,u}} \quad (2)$$

that is we calculate the weighted average of the similarity of the expected actions for the neighbor of the scene, $w_{x,u}$ is the similarity between the agent x and the agent u, whereas E_1 is estimate as follows:

$$\hat{E}_1 \left(sim(\beta_u, \alpha)|s\right) = \frac{1}{|N_u(s)|} \sum_{\beta_u \in N_u(s)} sim(\beta_u, \alpha) \quad (3)$$

that is the average of $sim(\beta_u, \alpha)$ over the set of actions $N_u(s)$ performed by u in s.

The algorithms described above and the general architecture described in Section 2 are fully implemented in Java using XML for expressing the cultural constraint theory.

4 Experiments in a Recommendation-Problem Domain

Collaborative filtering can be seen as a particular SICS. In this section we present the results of some experiments aimed to compare collaborative filtering and the

SICS parametrized for a recommendation problem. The goal of the experiments is validating the system against a well-estabilished method on a particular domain.

For collaborative filtering we use a memory and neighborhood based algorithm presented by Herlocker et al. in [9]. The algorithm consists of three basic steps. In the first step all the users are weighted with respect to their similarity with the active user by *Pearson correlation coefficient*:

$$w_{a,u} = \frac{\sum_{i=1}^{m}(r_{a,i} - \overline{r}_a) * (r_{u,i} - \overline{r}_u)}{\sigma_a * \sigma_u} \tag{4}$$

where m is the number of objects co-valutated, $r_{a,i}$ is the ranking given by the user a to the object i, $\overline{r}_a$ and σ_a are respectively the average and variance of the rankings of a. In the second step the best k_{CF} correlates are picked in order to compute a prediction (third step) using the *deviation-from-mean* approach introduced in [17].

The SICS used for the experiments is characterized by: a set of agents (users) $\mathcal{P} = \{u_1, \cdots, u_n\}$; a set of objects $\mathcal{O} = M \cup V$ (M is the set of items and $V = \{0, 0.2, \cdots, 0.8, 1\}$ the set of the possible votes) and a set of possible actions $\mathcal{A} = \{vote, request\}$. For a recommender system the cultural theory is specified in advance so no inductive module is needed:

$$\forall x \in P, m \in M: \quad request(x) \rightarrow vote(x, m, v_{max})$$

is expressed by the following rule:

$$request(x, t_1) \rightarrow vote(x, m, v_{max}, t_2) \wedge less(t_1, t_2)$$

that states that if x requests a suggestion, then x will assign the maximum vote to the proposed item. For a recommender system we require the satisfation of the user with the recommended items. For similarity between actions we use the following domain-dependent function:

$$sim(A_1, A_2) = \begin{cases} 0 & if(A_1 = vote \wedge A_2 = request) \vee \\ & (A_1 = request \wedge A_2 = vote) \vee \\ & (A_1 = vote(x, o, v_1) \wedge \\ & A_2 = vote(y, p, v_2) \wedge o \neq p) \\ 1 & if(A_1 = A_2 = request) \\ 1 - |v_1 - v_2| & if(A_1 = vote(x, o, v_1) \wedge \\ & A_2 = vote(y, p, v_2) \wedge o = p) \end{cases} \tag{5}$$

that means that the similarity is zero when the actions are different or when they are both *vote* but about distinct objects; it is maximum (namely 1) when the actions are both *request*; and finally, it is $1 - |v_1 - v_2|$ when the actions are both *vote* about the same object, but with different numerical votes.

For the experimentation we used the database EachMovie [13] that collects data of 72961 users who voted 1623 movies. We built a dataset considering the first 50 movies and the 119 users with at least one vote among the first 300 and we run a leave-on-out w.r.t the users. The CF algorithms returns a list of estimated ranking while in this case the SICS returns only a scene composed

Table 1. Experimental results. Distribution in percentage of the items proposed by SICS in the rankings position proposed by the CF algorithm with $K_{CF} = 30$.

action sim	eq. 5 (domain-dependent similarity)								eq 7
agent sim	eq. 4 (Pearson Correlation coefficient)							eq. 1	eq. 4
sim min	T_{min}=0						T_{min}=0.8	T_{min}=0	T_{min}=0
neighbors	k=10	k=30	k=50	k=80	k=150	k=300	k=80	k=80	k=80
ranks									
1	25.4	57.6	65.3	68.6	68.6	69.5	66.1	58.0	6.8
2	14.4	11.0	11.0	10.2	10.2	11.0	12.7	14.3	4.2
3	8.5	7.6	6.8	7.6	8.5	6.8	6.8	8.4	4.2
4	10.2	5.1	5.1	4.2	4.2	3.4	5.9	5.9	4.2
5	6.0	2.5	1.7	0.0	0.0	1.7	0.0	1.7	5.1
over 5	35.5	16.2	10.1	9.4	8.5	7.6	8.5	11.7	75.5

by a movie. We compared the movie proposed by the SICS and the best 10 movies ranked by the CF algorithm computing the distribution in percentage. We run our implementation of the CF algorithm obtaining a Mean Absolute Error comparable to the ones presented in the literature [3,7]. We run the experiments on the SICS changing the number of neighbours, the threshold and the functions used to compute the similarity among agents and actions. Table 1 presents the results of the experiments.

A low k implies a low number of scenes (items) valutated and consequently low performance (compare columns 1-6) because it is likely to miss some valuable items. With $T_{min} = 0$ the number of valutated scenes is on average 38.4 against 25.2 with $T_{min} = 0.8$, hence the speed of presentation is higher with a slight decrease in the performance (compare columns 4 and 7). The domain-dependent agent similarity function eq 4 is slightly better that the general one 1 (compare columns 4 and 8). Finally, a very simple action similarity (eq. 7) performs poorly (compare columns 4 and 9). We can conclude that with domain-dependent similarities our SICS is comparable with CF. More interestingly, comparing the last cells of column 4 and 8 it is possible to conclude that also the version without the domain-dependent agent similarity but with a general one performs satisfactorily.

5 An Application: The eCulture Brokering System

In this section, we present the eCulture Brokering System [2], a multi-agent system for cultural information brokering where we have applied the SICS. The multi-agent system has been developed using JACK Intelligent Agents [4].

In a multi-agent system, a SICS can be either a general capability of the overall system or a specific capability of a single agent. In the former case, the SICS observes all the agents acting in the system and manipulates the environment. In the latter, the SICS is applied to what the agent is able to observe and change, namely the part of environment and the agents it interacts with. The SICS capability, both general and specific, affects the whole system. In the system we present here, we choose to adopt the second option where a SICS is a capability of a single agent. In order to gain effectiveness we embedded a SICS in a Directory Facilitator (DF), namely an agent that plays a central role in the interactions. In particular, we adopt the idea of DF from FIPA specifications [5] and we extends its capabilities with the SICS.

A DF is a mandatory agent of an agent platform that provides a yellow pages directory service to agents. Every agent that wishes to publicize its service to other agents, requests the registration to the DF providing a description of its services. An agent can ask the DF in order to request information about the services available and the agents registered for such services. By means of a SICS, the DF can produce the Implicit Culture relation between an agent and the agents that have previously requested information, and provide information that encounters the preference of the agent.

In particular, it focus on the agents interaction for which we use the SICS. The platform contains a set of personal agents `Pa_1`,...,`Pa_h`, a `DF` that provides information about a set of brokers `B_1`,...,`B_n` and a set of wrappers `W_1`,...,`W_m`. A personal agent is created and assigned to each user who accesses the system by means of a web browser. The brokers are specialized in providing information about a specific cultural area (for instance, history, archeology, art, etc...), and they can collect information from different wrappers. Each wrapper is built for a specific museum's database. Basically, the databases are of two types: Microsoft Access and Oracle. The complete architecture includes other agents, like for instance the agent resource broker, which provides information about the resources available outside the multi-agent system. The rule used to express the cultural theory is the following:

$$\begin{array}{l}\mathtt{request}(\mathtt{x},\mathtt{DF},\mathtt{s},\mathtt{t}_1)\wedge\mathtt{inform}(\mathtt{DF},\mathtt{x},\mathtt{y},\mathtt{t}_2)\wedge\mathtt{less}(\mathtt{t}_1,\mathtt{t}_2)\rightarrow\\ \quad\mathtt{request}(\mathtt{x},\mathtt{y},\mathtt{s},\mathtt{t}_3)\wedge\mathtt{less}(\mathtt{t}_2,\mathtt{t}_3)\end{array} \tag{6}$$

that states that if `x` asks DF for the service `s`, and DF replays informing `x` that `y` can provide such a service, then `x` will request to `y` the service `s`.

The similarity function between two actions is the following:

$$sim(\beta,\alpha)=\begin{cases}1 & if\ \beta=\alpha\\ 0 & otherwise\end{cases} \tag{7}$$

where $\beta=\alpha$ if the actions are of the same type (namely, `request` or `inform`) and have the same arguments.

The `DF` uses the SICS to suggest the broker to the personal agent. In particular, for each personal agent that sends a request, the `DF` finds all the agents that have previously performed similar actions (requests and consequent response messages), and then suggests the personal agent with the broker for which such agents would be satisfied. The experiments we have made have shown that the SICS can effectively improve the interaction among agents. In particular, it can help new agents (users), that do not know the domain, to interact with the multi-agent system.

6 Conclusions and Future Work

We have presented an implementation of a System for Implicit Culture Support that exploit an application of instance-based learning techniques. We have showed that, in a particular domain and with a simple a priori theory, the system is functionally equivalent to Collaborative Filtering. Moreover we have presented a real-world application. Our three-steps algorithm for proposing the scene can be considered a generalization of a Collaborative Filtering algorithm, where the similarity is performed on executed actions and not on ratings. This generalization is non–trivial for it puts the SICS in a wider framework than simple CF. In fact it is possible to vary the domain, the cultural contraint theory, and also to deal with artificial agents as we have shown with the eCulture system. A relevant portion of research on multi-agents learning [18] deals with reinforcement learning (e.g., [10]). From the point of view of reinforcement learning the works of Price and Boutillier [16] on *Implicit Imitation* are relevant to our work. The critical difference is that a SICS does not imitate but *induce* an agent to imitate or more generally to act consistently with another group of agents. In the broad area of web personalization a relevant work is the one by Paliouras et. al. [15] who clusters communities of on-line users. In our perspective the groups are given, consequently an integration of the methods would be interesting. Finally, our approach can be seen as a temptative of supporting organizational learning in the direction proposed in [19]. Future work will be devoted to experimentation on domains with a wider range of actions, more complex scenes and more complex, possibly induced, cultural constraint theories.

Acknowledgments. The authors wish to thank Fausto Giunchiglia and Paolo Traverso for their precious support and Paolo Massa and Sabrina Recla for their work on the earlier phases of the project.

APPENDIX A: Formal Definition of Implicit Culture

We consider *agents* and *objects* as primitive concepts to which we refer with strings of type *agent_name* and *object_name*, respectively. We define the *set of agents* $\mathcal{P}$ as a set of *agent_name* strings, the *set of objects* $\mathcal{O}$ as a set of *object_name* strings and the *environment* $\mathcal{E}$ as a subset of the union of the set of agents and the set of objects, i.e., $\mathcal{E} \subseteq \mathcal{P} \cup \mathcal{O}$.

Let *action_name* be a type of strings, E be a subset of the environment ($E \subseteq \mathcal{E}$) and s an *action_name*.

Definition 1 (action). *An action α is the pair $\langle s, E \rangle$, where E is the argument of α ($E = arg(\alpha)$).*

Let $\mathcal{A}$ be a set of actions, $A \subseteq \mathcal{A}$ and $B \subseteq \mathcal{E}$.

Definition 2 (scene). *A scene σ is the pair $\langle B, A \rangle$ where, for any $\alpha \in A$, $arg(\alpha) \subseteq B$; α is said to be* possible in σ. *The* scene space $\mathcal{S}_{\mathcal{E},\mathcal{A}}$ *is the set of all scenes.*

Let T be a numerable and totally ordered set with the minimum t_0; $t \in T$ is said to be a *discrete time*. Let $a \in \mathcal{P}$, α an action and σ a scene.

Definition 3 (situation). *A situation at the discrete time t is the triple $\langle a, \sigma, t \rangle$. We say that a* faces *the scene σ at time t.*

Definition 4 (execution). *An* execution *at time t is a triple $\langle a, \alpha, t \rangle$. We say that a* performs *α at time t.*

Definition 5 (situated executed action). *An action α is a* situated executed action *if there exists a situation $\langle a, \sigma, t \rangle$, where a performs α at the time t and α is possible in σ. We say that a performs α in the scene σ at the time t.*

When an agent performs an action in a scene, the environment reacts proposing a new scene to the agent. The relationship between the situated executed action and new scene depends on the charateristics of the environment, and in particular on the laws that describe its dynamics. We suppose that it is possible to describe such relationship by an environment-dependent function defined as follows:

$$F_{\mathcal{E}} : A \times \mathcal{S}_{\mathcal{E},\mathcal{A}} \times T \rightarrow \mathcal{S}_{\mathcal{E},\mathcal{A}} \tag{8}$$

Given a situated executed action α_t performed by an agent a in the scene σ_t at the time t, $F_{\mathcal{E}}$ determines the new scene σ_{t+1} $(= F_{\mathcal{E}}(\alpha_t, \sigma_t, t))$ that will be faced at the time $t + 1$ by the agent a.

While $F_{\mathcal{E}}$ is supposed to be a deterministic function, the action that an agent a performs at time t is a random variable $h_{a,t}$ that assumes values in $\mathcal{A}$.

Let $a \in \mathcal{P}$ and $\langle a, \sigma, t \rangle$ be a situation.

Definition 6 (expected action). *The* expected action *of the agent a is the expected value of the variable $h_{a,t}$, that is $E(h_{a,t})$.*

Definition 7 (expected situated action). *The* expected situated action *of the agent a is the expected value of the variable $h_{a,t}$ conditioned by the situation $\langle a, \sigma, t \rangle$, that is $E(h_{a,t} | \langle a, \sigma, t \rangle)$.*

Definition 8 (party). *A set of agents $G \subseteq \mathcal{P}$ is said to be a* party*.*

Let $\mathcal{L}$ be a language used to describe the environment (agents and objects), actions, scenes, situations, situated executed actions and expected situated actions, and G be a party.

Definition 9 (cultural constraint theory). *The* Cultural Constraint Theory for G *is a theory expressed in the language $\mathcal{L}$ that predicates on the expected situated actions of the members of G.*

Definition 10 (group). *A party G is a* group *if exists a cultural constraint theory $\mathbf{\Sigma}$ for G.*

Definition 11 (cultural action). *Given a group G, an action α is a* Cultural Action w.r.t. G *if there exists an agent $b \in G$ and a situation $\langle b, \sigma, t\rangle$ such that*

$$\{E(h_{b,t}|\langle b, \sigma, t\rangle) = \alpha\}, \mathbf{\Sigma} \not\vdash \perp$$

where $\mathbf{\Sigma}$ is a cultural constraint theory for G.

Definition 12 (implicit culture). Implicit Culture *is a relation $\rtimes\!\!\!\ltimes$ between two parties G and G' such that G and G' are in relation ($G \rtimes\!\!\!\ltimes G'$) iff G is a group and the expected situated actions of G' are cultural actions w.r.t G.*

Definition 13 (implicit culture phenomenon). Implicit Culture Phenomenon *is a pair of parties G' and G related by the Implicit Culture.*

We justify the "implicit" term of implicit culture by the fact that its definition makes no reference to the internal states of the agents. In particular, there is no reference to beliefs, desires or intentions and in general to epistemic states or to any knowledge about the cultural constraint theory itself or even to the composition of the two groups. In the general case, the agents do not perform any actions explicitly in order to produce the phenomenon.

References

1. E. Blanzieri and P. Giorgini. From collaborative filtering to implicit culture. In *Proceedings of the Workshop on Agents and Recommender Systems*, Barcelona, 2000.
2. E. Blanzieri, P. Giorgini, P. Massa, and S. Recla. Implicit culture for multi-agent interaction support. In *Proc. of the Conf. Cooperative Information Systems COOPIS*, 2001.
3. J. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proc. of the Fourteenth Conf. on Uncertainty in Artificial Intelligence*. Morgan Kaufmann, 1998.
4. P. Busetta, R. Rönnquist, A. Hodgson, and A. Lucas. Jack intelligent agents – components for intelligent agents in java. AOS TR9901, January 1999. http://www.jackagents.com/pdf/tr9901.pdf.
5. FIPA. *Foundation for Intelligent Physical Agents*. http://www.fipa.org.
6. E. Blanzieri P. Giorgini and F. Giunchiglia. Implicit culture and multi-agent systems. In *Proceedings of the International Conference on Advances in Infrastructure for Electronic Business, Science, and Education on the Internet, (SSGRR 2000)*, L'Aquila – Italy (http://www.science.unitn.it/ pgiorgio/ic), 2000.
7. A. Gokhale. Improvements to collaborative filters algorithms. Master's thesis, Worchester Ploytechnic Institute, May 1999.
8. D. Goldberg, D. Nichols, B.M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, 1992.
9. J. Herlocker, J. Konstan, A. Borchers, and J Riedl. An algorithmic framework for performing collaborative filtering. In *Conference on Research and Development in Information Retrieval*, 1999.

10. J. Hu and M. Wellman. Multiagent reinforcement learning theoretical framework and an algorithm. In *Proceedings of the ICML*, Madison, Wisciautoh, 1998. Morgan Kaufmann Publishers.
11. M. Klusch and F. Zambonelli, editors. *Cooperative Information Agents V. 5th International Workshop, CIA 2001, Modena, Italy, September 6–8, 2001. Proceedings*, volume 2182 of *LNCS*. Springer-Verlag, 2001.
12. J.A. Konstan, B.N. Miller, D. Maltz, J.L. Herlocker, L.R. Gordon, and J. Riedl. Grouplens: Applying collaborative filtering to usenet news. *Communications of the ACM*, 40(3):77–87, 1997.
13. P. McJones. Eachmovie collaborative filtering data set, dec systems research center, 1997. http://research.compaq.com/SRC/eachmovie/.
14. I. Nonaka and H. Takeuchi. *The knowledge Creating Company.* Oxford University Press, New York, 1995.
15. G. Paliouras, C. Papatheodorou, V. Karkaletsis, and C. Spyropoulos. Clustering the users of large web sites into communities. In *Proceedings of the ICML*, Stanford, 2000. Morgan Kaufmann Publishers.
16. B. Price and C. Boutilier. Implicit imitation in multiagent reinforcement learning. In *Proceedings of the ICML*, Bled, 1999. Morgan Kaufmann Publishers.
17. Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of ACM CSCW'94 Conference on Computer-Supported Cooperative Work*, 1994.
18. P. Stone and M. Veloso. Multiagent systems: A survey from amachine learning perspective. *Autonomous Robots*, 8(3):345–383, June 2000.
19. K. Takadama, T. Terano, and K. Shimohara. Can multiagents learn in organization? – analyzing organizational learning-oriented classifier system. In *IJCAI'99 Workshop on Agents Learning about, from and other Agents*, 1999.

From Physical Agents to Recommender Agents

Josep Lluís de la Rosa, Esteve del Acebo, Beatriz López, and Miquel Montaner

Universitat de Girona
Av. Lluís Santaló s/n
17071 Girona, Spain
{peplluis,blopez,mmontane}@eia.udg.es
acebo@ima.udg.es,

Abstract. In this paper, we argue in favour of the interchange of research results between physical agents and recommender systems. Certain properties, such as introspection and physical foundations of physical agents, are mapped into agents that model and make recommendations to users from the subjective, content-based point of view of products and services. How this real-to-virtual mapping of properties and behaviours can be performed is the main goal of our work, from which we expect to obtain a general concept of recommender agents which improves the state of the art in terms of performance and maintainability. Furthermore, new features of the recommender agents could be also mapped to the physical agents discipline. Some of the features explained are installed in two personalization company products: Proto-agent and Habitat-Pro.

1 Introduction

Researchers think that physical agents and recommender agents belong to two different research lines and, so far, research on both kinds of agents has been performed separately. However, we have found analogies between the two.

A physical agent is an entity whose actions and co-operation with other agents are limited and conditioned by its physical body, typically a robot [Asada97]. A recommender agent is an entity whose actions and co-operation with other agents are limited and conditioned by the combination of: the modelling of preferences of particular users, building content models and the modelling of social patterns [Maes94]. Since physical agents commit to better actions according to introspected capabilities from the physical body ([de la Rosa99], [Oller00]), in the same way recommender agents would commit to better transactions according to introspected tastes of human beings. Thus, we think that most of the results obtained in the physical world, like automatic control-oriented concepts, such as excitation, dynamics and stability of the physical bodies ,can be imported to the recommender field.

How this real-to-virtual mapping of properties and behaviours can be performed is the main goal of our work. We expect to obtain a general concept of recommender agents that is an improvement on current understanding, in terms of performance and maintainability. Furthermore, new features of the recommender agents could also be mapped to the discipline of physical agents.

M. Klusch et al. (Eds.): Intelligent Information Agents, LNAI 2586, pp. 165–178, 2003.

This paper is organised as follows: First, we outline the properties of physical agents and recommender agents in sections 2 and 3 respectively. Then, we explain the analogies between both kinds of agents in section 4. Section 5 is devoted to experimental results and finally, in section 6 we offer several conclusions.

2 Physical Agent Properties

Robotics research has been focused for years on the development of modular systems composed by perception, reasoning and actuator components. More recently, GOFAIR (Good Old Fashioned Artificial Intelligence and Robotics, [Mackworth93]) has moved towards new approaches in order to obtain an operational cognitive integration. Most of these approaches are preceded by Brooks research [Brooks91] which claimed that AI techniques did not work either in real time systems or in dynamic environments, and that the key issues in robot development are situation and embodiment.

On the one hand, situation means that robots are in the world. They do not deal with abstract situations but with the "here and now" of the environment that has a direct influence on the system behaviour. On the other hand, embodiment underlines the fact that robots have body, they directly receive world inputs, their actions are part of the world's dynamics and have immediate effects on robot sensors. Sensing and acting are highly interconnected; they cannot be separated [Asada97].

Research on physical agents is framed within this new line of Robotics. From situation and embodiment follow several properties of physical agents, among which we want to distinguish introspection, subjectivity and perception of the co-operative world.

First, introspection is perception of the agent's own physical body. Physical inputs and outputs from the environment (dynamics of the physical body) are mapped in the knowledge base of each agent in what we call physical knowledge. Such knowledge is represented by a further declarative control level and a further declarative supervision level [Müller96] and is declared by means of capacities. According to the capacities gathered though introspection, the physical agent is able to make compromises in the co-operative world. [1]

Second, subjectivity in the perception of the physical world: the actions of the physical agent are conditioned by its capacities. For example, 10 m might be a long distance for a low-speed physical agent, but the same distance could be a quite short one for a high-speed robot. Moreover, 10 m could be a long distance even for a high-speed robot if its battery is low.

---

[1] It is important to note that the term capacity is related to the dynamics of the physical body. Other static features, such as for example, being holonomic or not, is knowledge that can be programmed a priori.

And third, perception of the co-operative world: each physical agent has capacities and it is able to sense the capacities of the other agents that live in its environment; biased, of course, by its own capacities. For example, agent A has a low precision capacity and for this agent, B and C could be two co-operative agents with a higher precision in relation to its own. Perception of the co-operative world is, therefore, also subjective and so has been used in developing multi-agent physical systems that co-operate in solving tasks. Perhaps the first results have been obtained in games, like in the Robocup tournaments (see, for example, [Oller00]), and then these results have been passed on to other domains, such as teamwork, rescue, and military operations, among others.

With regard to introspection, it is clear that physical agents are complex physical systems, and they are not easy to understand for use in a computational way, based on capacities. Being physical systems, however, we can take advantage of automatic control theory to build dynamic models by using identification methods: temporal, frequential and stochastic identification.

So, we propose to extend the identification methods from control theory to physical agents theory, and therefore extend the *excitation* concept. Figure 1 shows how to identify a dynamic system from an input/output point of view, according to the current thinking ([Eykhoff74], [Ljung94], [Schoukens91], [Walter97]).

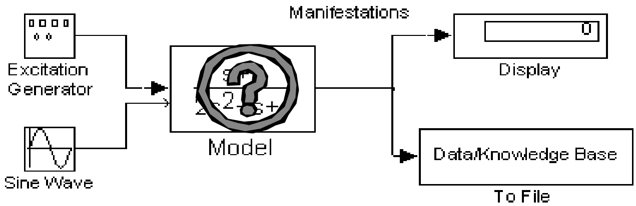

Fig. 1. Identification through excitation.

There is a theory based on stochastic systems identification that measures random input data and the consequent output of the system [Schoukens91] [Walter97]. These techniques can get towards an approximation of the model. The drawback is that if the input data is poor, that is, if the system is not persistently excited, then there is no chance of any sort of correct identification of the system. In other words, the models are poor ([Ljung94] and [Walter97]).

So, for any model, the excitation of the system is crucial. The more excited a system is, the more modes are excited and tend to manifest (output); otherwise, there may be several hidden modes that are not known. The models contain a state, that is, the system responds differently depending on the inner state, and this state depends on the history of previous inputs (convolution). The response of impulsion input is called homogeneous response, and other, more complex inputs are the heterogeneous response (see Figure 2).

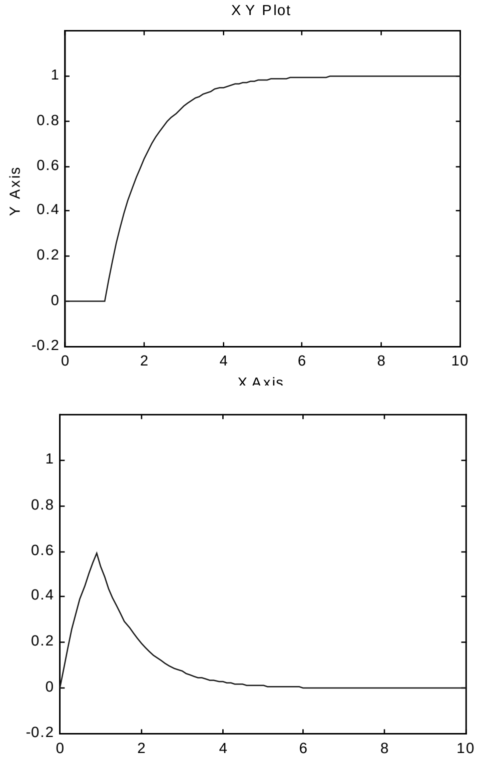

Fig. 2. Heterogeneous (excited/forced) response (left); Homogeneous (non-excited/unforced) response (right)

What if a system is highly complex? For instance, in those cases where there are many inputs and outputs. With control theory, the proposed solution is to identify every input-output pair and to obtain several models. Thus, the interactions among excitation and manifestations can be studied one by one.

Transferring these concepts to physical agents, a physical agent is a type of highly complicated system that we need to understand. So,

1. One way to understand the complexity of physical agents is to study several inputs/outputs.
2. The physical agents have to be persistently excited, that is, by several types and a lot of excitation.
3. The physical agents have strong states that make them respond in several ways to the same inputs: for example saturation, minimum phase, delays, temporal dynamical constants, relay behaviour (contradictions-“it takes longer going from New York to Barcelona than from Barcelona to New York”), etc.

One possible way of building physical agents is by taking advantage of these concepts borrowed from Systems Engineering theory and applying them when modelling such agents.

3 Recommender Agents

The introduction of Internet, World Wide Web, communications networks, and widespread computation and storage capabilities, has resulted in a global information society with a growing number of users around the world. Information, the precious raw material of the digital age, has never been so easy to obtain, process and disseminate through the Internet. Yet, with the avalanche of information at our doors, there is a rapidly increasing difficulty of finding what we want, when we need it, and in a way that better satisfies our requirements. It is often necessary to make choices without sufficient personal experience of the alternatives.

Recently, in the Artificial Intelligence community, there has been a great deal of work on how AI can help to solve the information overload problem and some research has been focused on what is called recommender systems. The main task of a recommender system is to locate documents, information sources and people related to the interests and preferences of a single person or a group of people [Sangüesa00]. This involves then, the construction of user models and the ability to anticipate and predict the user's preferences.

Intelligent agents have a set of characteristics that make them a natural choice as a basis for the construction of recommender systems. In Figure 3 we show our model of personalization and recommendation. Several users interact with a habitat (an e-commerce portal, for example) through a set of agents that represent them. Agents have a twofold mission. On the one hand, they interact among each other, and with the users they represent. On the other hand, they filter the information that arrives to the users from the habitat and the other agents.

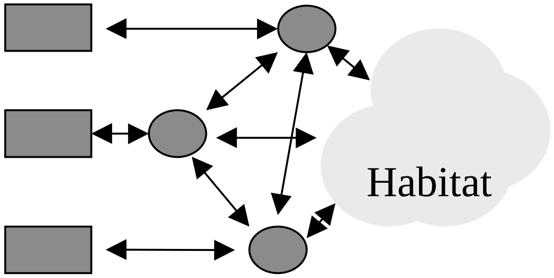

Fig. 3. Our model of personalization and recommendation using agents. Circles represent agents and squares represent users.

Agents are individuals, in the sense that each agent represents or “belongs” to a single user. Agents have knowledge about the tastes and aims of the users they represent, and are capable of learning from their interactions with the users and the environment.

In addressing the task of assisting users with recommendations, three information filtering methods are currently proposed [Montaner01]: demographic filtering, content-based filtering and collaborative filtering. Demographic filtering methods use descriptions of the people in order to learn a relationship between a single item and the type of people that like that object. Content-based filtering methods use descriptions of the content of the items to learn a relationship between a single user and the description of the items. Collaborative filtering methods use the feedback of a set of people on a set of items to make recommendations, but ignore the content of the items or the descriptions of the people. Every method has some advantages and drawbacks (see [Muntaner01]). A well-known problem of content-based filtering, for example, is the fact that items dissimilar to ones offered previously to a user are never suggested for advice. Then, in recent years, researchers claim that hybrid systems offer better performance. So, content-based filtering methods can be complemented by collaborative filtering methods that provide new items for similar users.

In collaborative filtering systems, agents with similar profiles exchange recommendations. However, when a similar agent gives unsuccessful advice, there is no way to ignore it. Over and over again this agent causes a decline in the other agent's performance. Marsh proposes the concept of trust to make our agents less vulnerable to others [Marsh94]. Trust is fundamental for any kind of action in an uncertain world; in particular it is crucial for any form of collaboration with other autonomous agents. Applying the concept of trust in the collaborative world, we can solve the problem that arises when a similar agent gives frustrated recommendations by decreasing the trust in this agent and ignoring its advice in the future. Trust is formed and updated over time through direct interactions or through information provided by other members of the society they have had experience of. Each event that can influence the degree of trust is interpreted by the agent to be either a negative or a positive experience. If the event is interpreted as a negative experience the agent will lose his trust to some degree and if it is interpreted as positive, the agent will gain trust to some degree. The degree to which the sense of trust changes, depends on the trust model used by the agent. This implies that the trusting agent performs a form of continual verification and validation of the subject of trust over time [Montaner02].

4 Properties Mapping

We have identified several properties inherent in physical agents and some others developed for recommender agents. Table 1 shows a summary of them.
What we want to do is complete the table. First by shifting properties of physical agents to the recommendation field and then by returning to the physical agents research field with new insights from recommender agents.

Table 1. Initial list of physical agent and recommender agent properties.

From Physical Agents		To Recommender Agents
Introspection (perception of its own physical body)		
Subjectivity (perception of the physical world)		
Subjectivity (perception of the co-operative world)		Trust
Identification of systems (excitation)		

We can shift properties from physical agents to recommender agents with the following in mind:

- An important point is to ensure that the agent is able to *commit* to both individual and co-operative actions with a set of other agents, within a highly changing environment [Oller99]

- Human perception performs as a selective filter whose goal is to avoid the person becoming inoperative due to information saturation. From this viewpoint, perception is a defence mechanism. But it also means that perception makes the information subjective: the person receives the things that make sense to him or her.

- We can regard customers, and people in general, as behaving like dynamic systems when selecting e-services and products from web sites. This gives us a hint of how to deal with them, so that they can be stimulated to show interest in products and services of companies and in the same way, so that they can be satisfied. Features like saturation or steady states, or random excitation are key-features imported from the automatic control domain, which have great potential for improving knowledge on every customer.

So, it is clear that:

- Recommender systems should be aware of their capacities for representing user interest, induced from the user profile. That is, being aware to what extent the agent is covering a given taste of the user and then being able to advise (i.e., commit) accordingly.
- User profiles of recommender agents, being models of the user, are subjective from the perception point of view. Thus, recommender agents should be aware of such subjectivity and exploit it accordingly.

- Excitation is a way to enrich user models when, by means of any machine learning method, we have obtained a probably stable but poor model of the user.

Now we can fill in the gaps and we get table 2.

Table 2. Properties mapping and shift of insights.

From Physical Agents		To Recommender Agents
Introspection (perception of its own physical body)	→	Recommendation based on capacities.
Subjectivity (perception of the physical world)	→	Subjectivity/Affinity
Subjectivity (perception of the co-operative world)	←	Trust
Identification of systems (excitation)	→	Excitation

Vice versa, that is, shifting insights from the recommender agent field to the physical agent field, we believe that research performed on trust can be used by physical systems to model social relations.

5 Experimental Results

Our experiments have been carried out in two stages. First we developed Proto-agent, an agent-based recommender system based on case-based reasoning. With Proto-agent we prove to some degree the shift of the identification property of physical agents to excitation in recommender systems. Currently, we are finishing Habitat-Pro™[(1)] to test introspection and subjectivity/affinity. First, we will give our results on Proto-agent and then we will outline what our approaches are to introspection and subjectivity in Habitat-Pro.

5.1 Proto-Agent: Results on Excitation

In order to test excitation, we built an environment where every proto-agent represents a person. Each agent emulates the decisions of the person in a certain field, for example, e-commerce. A proto-agent is a bare agent that contains only one good single input and single output model of a specific aspect of human behaviour on the Internet. That is, the model of human behaviour in contracting or getting goods and services. It does not yet incorporate the proactivity that a basic agent should present.

[(1)] Habitat-Pro™ is the Trade Mark of Agents Inspired Technologies, a spin-off company of the University of Girona.

The proto-agent is case-based. Cases represent excitation, from the viewpoint of Systems Engineering, that is, input and outputs regarding human behaviour. So, our starting point for modelling the proto-agent behaviour involves the following excitations:

1. The transaction history of the person in a certain area: this is the excitation feature for the state of the model.
2. The contextual information by means of click stream and contextual offers response: this is the excitation feature for the present state of the model.
3. The response to limited offers from a company: again by click stream, it shows another aspect of the present state of the model, but this is a response of the person to the excitation feature of the company.
4. The response to suggestions (offers) from other persons: again by click stream, shows still another aspect of the present state of the model, that is response of the person to the excitation feature of the community of users.

Let us start with a consistent customer. As depicted in the plot in Figure 4, we can see how the knowledge of the person stabilizes and becomes saturated at a maximum degree of performance. To reach the performance level 1 means the user is known completely. However, it is unrealistic to assume that a person is consistent and invariable over time, or that we can achieve perfect knowledge of his/her hidden modes.

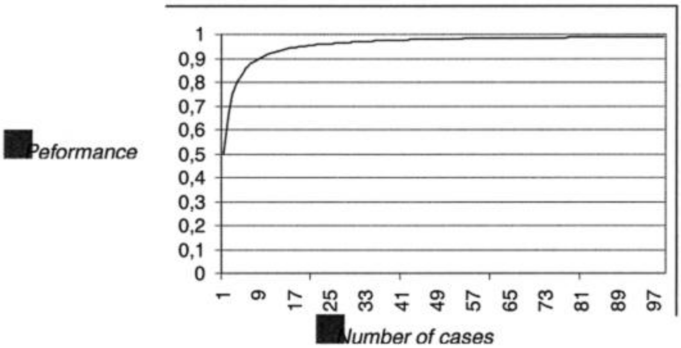

Fig. 4. Stabilisation and performance saturation for a consistent customer.

A more typical case is the following regular customer, whose plot is shown in Figure 5. The same 1st order envelope of the consistent customer appears but with much more noise and oscillation over the saturation state. The user again stabilizes and the performance does not increase any more.

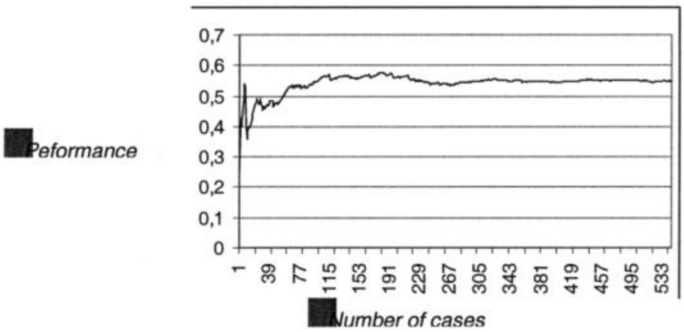

Fig. 5. Stabilisation and performance saturation for a regular customer.

However, an interesting result of case-based reasoning (CBR) is that the stabilization of the performance indicates the correct number of cases required to achieve a settled performance, which in this case is 115 cases to achieve a 50% steady state response. In order to maintain persistent excitation, we extended the CBR cycle (see Figure 6). This way, the new cases proposed to the user could be obtained from random proposal or by collaborative filtering or from other sorts of excitation which stimulate the user. If the user confirms the solution then this case is introduced into the case base.

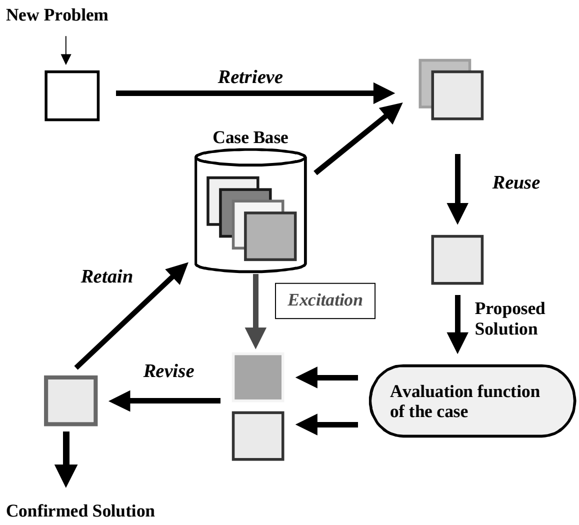

Fig. 6. New CBR cycle resulting from incorporating persistent excitation.

The results of the excitation experiment can be seen in Figure 7. The plots show that performance stabilises at 78%, an improvement of approximately 50%. All the experiments performed in our laboratory follow this pattern of between 30-60% improvement, which shows that including the excitation method from physical agents theory to the recommender agents is quite profitable, and a very promising field in which to do further research i.

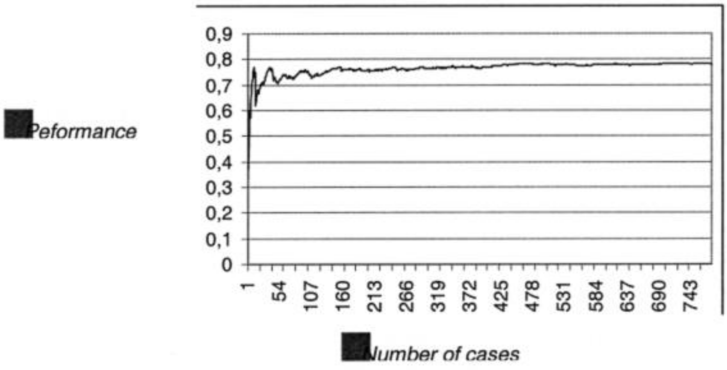

Fig. 7. Results of the new CBR cycle.

5.2 HABITAT-PRO™: Introspection and Subjectivity

The heart of Habitat-Pro™ is a Java application server together with a Java Virtual Machine (JVM). Agents "live" as servlets inside the JVM. The application server administrates the JVM, creating and destroying on request, agents and other servlets and support objects. Additionally, there is a database that stores, among other data, the user and product profiles used by the agents. (See the architecture of the system in Figure 8).

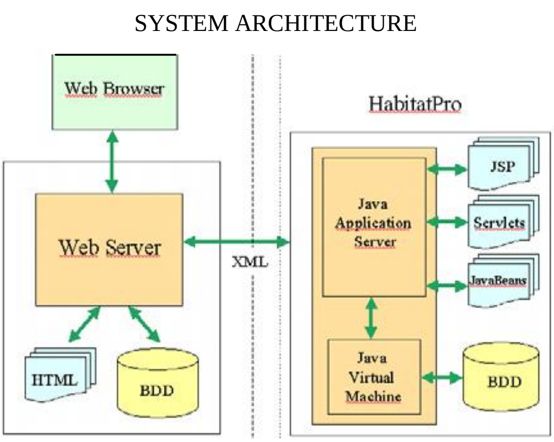

Fig. 8. HabitatPro™'s architecture. To the left, the external system (for example, an e-commerce site). To the right, the structure of the agent based personalization system Habitat-Pro.l

Introspection in Habitat-Pro

Personalization in Habitat-Pro is based on a somewhat fuzzyfied version of the well-known concept of Attribute-Value pair. We identify the concept of attribute with the one of property or characteristic of a product (for example colour or price). The values that the attributes can take (for example greenish or cheap) are, in general, subjective. The meaning of a value can vary depending on the person or agent that uses, defines or assigns it. It is necessary, then, to use sophisticated techniques to use and manipulate them.

User and product modelling in Habitat-Pro™ is carried out by defining two new concepts from the attribute-value pair concept:

- Image of a product: the set of attribute-value pairs that characterize the product.
- Image of a user: the set of attribute-value-weight triplets, codifying the preferences of the user and the strength of those preferences.

Preferences of the user are dynamic and are updated accordingly, though interaction with the physical world.

Perception of the Physical World/Model Adjustment in Habitat-Pro

Learning from experience and from perception of the physical, "outer", world is an essential characteristic of agency. In Habitat-Pro, the knowledge incorporated and used by the agents is basically concerned with the preferences and tastes of the users. Learning, therefore, consists of a progressive refinement of the image of the users.

Each time a customer buys a product, the image of the customer is refined in order to adjust it to this fact. We maintain, for each attribute and user, not only some kind of mean value, but also a weighting that essentially reflects the degree of dispersion of the values for the attribute in the set of products purchased by the client. In this way, attributes for which the customer usually acquires products with similar values have high weightings, while attributes with more dispersed values have lower ones.

The magnitudes of the updates of the customer images vary depending on different parameters such as:

- The interest shown by the customer: a customer can show a slight interest (asking for more information about it, for example) or a strong interest (buying it).
- The kind of product involved: we are more confident when adjusting the image of the customer who is purchasing an expensive or very specific item than we are when adjusting the image of a customer who is buying a pizza, for example.
- The reliability of the image of the customer: meaning essentially, the number of interactions between the customer and Habitat-pro.
- The time between image updates: people change their tastes over time. Adjustments in the images of customers are larger when the images are "old".

Perception of the Cooperative World/Trust in Habitat-Pro

Habitat-Pro addresses matters of trust in a very straightforward way. It is possible to assign confidence or trust values to the recommendations that agents make, by associating two quantities to each agent (or, more precisely, to the image of the user represented by the agent):

- The reliability of the agent: which basically measures the number of times that the customer has purchased or shown some interest in some product and, thus, the number of times that his image has been updated.
- The quality of the agent: which measures the specificity of the tastes of the customers. It's clear that it's easier to make good recommendations to a customer with a sharp image (with high weightings for the attributes) than for a customer with a soft one (with low weightings, corresponding to a user with a more random shopping pattern).

6 Conclusions

In this paper we have analysed several properties of physical agents that can enrich research into recommender systems. This is the case with introspection of physical capacities, and subjectivity from interaction with the physical world. Furthermore, identification theories from Systems Engineering can be used to represent the minimal set of inputs and outputs required by an agent in order to obtain a good model. Inversely, recommender systems can provide insights regarding interaction in the co-operative world, through trust formalisms.

We have presented the first prototypes developed to test our theories and we have shown some initial results. We think that this collaboration between the two lines of research, physical agents and recommender systems, creates a synergy from which each discipline can make bigger advances. We will continue in this line, looking for further opportunities for new knowledge transference between the two research lines.

References

[Asada97] Asada M., Kuniyoshi Y., et al. The RoboCup Physical Agent Challenge, *First RoboCup Workshop in the XV IJCAI-97 International Joint Conference on Artificial Intelligence*, pp.51–56, 1997.

[Brooks91] Brooks, R.A. "New Approaches to Robotics". Science, Vol. 253, September 1991m pp. 1227–1232, 1991.

[de la Rosa99] de la Rosa J.Ll., Garcia R., Innocenti B., Muñoz I., Figueras A., Ramon J.A., Montaner M., "Rational Dynamical Physical Agents", 3rd Robocup Workshop, Vol. on RoboCup-99: Robot Soccer World Cup III, Lecture Notes in AI Nº1395, Eds. Springer-Verlag, Estockholm (Sweden). July 31-August 6, 1999.

[Eykhoff74] Eykhoff P, Systems Identification, *John Wiley & Sons*, 1974.

[Mackworth93] Mackworth, A.K. "On Seeing Robots". In: Basu, A. And Li, X. (eds), Computer Vision: Systems, Theory and Applications. Singapore: World Scientific Press, 1:13, 1993.

[Maes94] Maes, P., "Agents that reduce work and information overload" Communications of the ACM vol. 37 no. 7, pp 30–40, July 1994.

[March94] March S. P. "Formalising trust as a computational concept". PhD Thesis, Department of Computing Science and Mathematics, University of Stirling, 1994.

[Montaner01] Montaner, M., "A Taxonomy of Personalized Agents on the Internet". Technical Report TR-UDG-2001-05-1. Departament d'Electrònica, Informàtica i Automàtica. Universitat de Girona. May, 2001.

[Montaner02] Montaner, M., López B., de la Rosa, J.L. "Developing trust in Recommender Agents". *Submitted to AAMAS 2002.*

[Müller96] Müller, J.P. . "The Design of Intelligent Agents. A Layered Approach". Lecture Notes in Artificial Intelligence, Vol. 1177. Ed. Springer-Verlag. 1996.

[Ljung94] Ljung L. and Glad T., Modeling of Dynamic Systems, *Prentice Hall International Inc., 1994*.

[Oller00] Oller A., de la Rosa J.Ll., García R, Ramon JA, Figueras A. "Micro-Robots playing soccer games: a real implementation based on a multi-agent decision-making structure", Int. Journal of Intelligent Automation and Soft Computing. Special Issue on Soccer Robotics: MIROSOT '97, Vol. 6, No. 1, January 2000.

[Oller99] Oller A, de la Rosa J.Ll., and del Acebo E., "DPA2: Architecture for Co-operative Dynamical Physical Agents", MAMAAW '99, Valencia June 1999.

[Sangüesa00] Sangüesa R., Cortés U., Faltings, B. W9 Workshop on Recommender Agents. Agents 2001.

[Schoukens91] Schoukens J. and Pintelon R., Identification of Linear Systems; A Practical Guideline to Accurate Modeling, *Pergamon Press,* 1991.

[Walter97] Walter E and Pronzato L., Identification of Parametric Models from Experimental Data, *Springer Verlag*, 1997.

Objective versus Subjective Coordination in the Engineering of Agent Systems

Andrea Omicini[1] and Sascha Ossowski[2]

[1] DEIS, Università di Bologna a Cesena
Via Rasi e Spinelli 176, 47023, Cesena, FC, Italy
aomicini@deis.unibo.it http://lia.deis.unibo.it/~ao
[2] ESCET, Universidad Rey Juan Carlos
Campus de Mostoles, Calle Tulipan s/n, E-28933 Madrid, Spain
S.Ossowski@escet.urjc.es http://www.ia.escet.urjc.es/~sossowski

Abstract. The governance of interaction is a critical issue in the engineering of agent systems. Research on *coordination* addresses this issue by providing a wide range of models, abstractions and technologies. It is often the case, however, that such a wide range of proposals could not easily find an unitary and coherent conceptual framework where all the different views and solutions can be understood and compared – and this is particularly true in the context of agent models and systems.

In this paper, we first discuss how all the many diverse approaches to agent coordination can be categorised in two main classes – the *subjective* and *objective* approaches – , depending on whether they adopt the agent's or the engineer's viewpoint, respectively. We then claim that the two approaches have a deep and different impact on the way in which agent systems are modelled and built, and show two examples rooted in different models and technologies. Finally, we advocate that both approaches play a fundamental role in the engineering of agent systems, and that any methodology for the design and development of agent systems has to exploit both objective and subjective coordination models and technologies.

1 Introduction

Multiagent systems (MAS henceforth) are software systems made up of multiple encapsulated computational entities, which are embedded in an environment and (inter-)act in an autonomous and intelligent fashion. When exactly software modules, objects, components etc. can be called agents, and as to how far and in what sense they are to behave intelligently and autonomously, is still subject to considerable debate. However, it is commonly accepted that coordination is a key characteristic of MAS and that, in turn, the capability of coordinating with others constitutes a centrepiece of agenthood.

Still, when researchers are asked to nail down their intuitive notion of what coordination in MAS is all about, agreement usually ends. To some respect this is not surprising, as the term is used in a variety of disciplines, such as

M. Klusch et al. (Eds.): Intelligent Information Agents, LNAI 2586, pp. 179–202, 2003.

Economy, Sociology and Biology, each with its own epistemological apparatus and research agenda. To make things worse, even within Computer Science the term is used quite differently in fields like robotics, concurrent programming, and mainstream software engineering. As a result, many different techniques and frameworks coexist in the agent community that claim to tackle the problem of coordination in MAS in one or another way, a situation that tends to be confusing not only to novice MAS designers. We feel that a clarification in this respect is urgently needed if, as we foresee, the development of heterogeneous and distributed systems based on the agent metaphor are to become next "big thing" in software engineering.

In this chapter, we first discuss how all the many diverse approaches to agent coordination can be categorised in two main classes, *subjective* and *objective* approaches, depending on whether they adopt the agent's or the engineer's viewpoint, respectively. We argue that the deep and different impact of these approaches on the way in which agent systems are modelled and built can provide important clues for a successful navigation through the jungle of the many, seemingly unrelated, works on agent coordination. Moreover, we advocate that both approaches play a fundamental role in the engineering of MAS for open environments such as the Internet, and that any modern methodology for the design and development of such agent-based applications will have to exploit both objective and subjective coordination models and technologies.

This chapter is organised as follows: we first review some of the most relevant coordination models, abstractions and technologies and relate them to each other. Subsequently, we introduce the notions of subjective and objective coordination, illustrate them by an example, and discuss their impact on the design of coordination mechanisms for MAS. Section 4 describes our position on how these notions may impact coordination related issues in agent-oriented software engineering. This chapter is concluded by our vision of the lines along which research on coordination in MAS may evolve.

2 Coordination: An Overview

This section provides a brief overview of current coordination models, abstractions and technologies. We first outline different attempts to *conceptualise* coordination in MAS. Then, we compare these abstractions along a variety of dimensions and discuss the role of "coordination middleware": software infrastructures that support the *instrumentation* of different coordination models.

2.1 Models of Coordination

Maybe the most widely accepted conceptualisation of coordination in the MAS field originates from Organisational Science. It defines coordination as *the management of dependencies* between organisational activities [11]. One of the many workflows in an organisation, for instance, may involve a secretary writing a letter, an official signing it, and another employee sending it to its final destination.

The interrelation among these activities is modelled as a *producer/consumer* dependency, which can be managed by inserting additional *notification* and *transportation* actions into the workflow.

It is straightforward to generalise this approach to coordination problems in multiagent systems. Obviously, the subjects whose activities need to be coordinated (sometimes called *coordinables*) are the agents. The entities between which dependencies arise (or: *objects of coordination*) are often termed quite differently, but usually come down to entities like goals, actions and plans. Depending on the characteristics of the MAS environment, a taxonomy of dependencies can be established, and a set of potential coordination actions assigned to each of them (e.g. [29]). Within this model, the *process* of coordination is to accomplish two major tasks: first, a *detection* of dependencies needs to be performed, and second, a *decision* respecting which coordination action to apply must be taken. A coordination *mechanism* shapes the way that agents perform these tasks [18].

The *result* of coordination, and its *quality*, is conceived differently at different levels of granularity. Von Martial's stance on coordination as *a way of adapting to the environment* [29], is quite well suited to understand this question from a *micro-level* (agent-centric) perspective, in particular if we are concerned with multiagent settings. If new acquaintances enter an agent's environment, coordination amounts to re-assessing its former goals, plans and actions, so as to account for the new (potential) dependencies between itself and other agents. If a STRIPS-like planning agent, for instance, is put into a multiagent environment, it will definitely have to accommodate its individual plans to the new dependencies between its own prospective actions and potential actions of others, trying to exploit possible synergies (others may free certain relevant blocks for it), and avoiding harmful dependencies (making sure that others do not unstack intentionally constructed stacks etc). At this level, the result of coordination, the agent's adapted individual plan, is the better the closer it takes the agent to the achievement of its goals in the multiagent environment.

From a *macro-level* (MAS-centric) perspective, the outcome of coordination can be conceived a "global" plan (or decision, action etc.). This may be a "joint plan" [23], if the agents reach an explicit agreement on it during the coordination process, or just the sum of the agents' individual plans (or decisions, actions etc. – sometimes called "multi-plan" [18]) as perceived by an external observer. Roughly speaking, the quality of the outcome of coordination at the macro-level can be evaluated with respect to the agents' joint goals or the desired functionality of the MAS as a whole. If no such notion can be ascribed to the MAS, other, more basic features can be used instead. A good result of coordination, for instance, is often supposed to be efficiency [23], which frequently comes down to the notion of Pareto-optimality: no agent could have increased the degree of achievements of its goals, without any other being worse off in that sense. The amount of resources necessary for coordination (e.g. the number of messages necessary) is also sometimes used as a measure of efficiency.

The dependency model of coordination appears to be particularly well suited to *represent* relevant features of a coordination problem in MAS. The TÆMS framework [5], for instance, has been used to model coordination requirements in a variety of interesting MAS domains. It is also useful to rationalise observed coordination behaviour in line with Newell's knowledge-level perspective [13]. Still, when designing coordination processes for real-world MAS, things are not as simple as the dependency model may suggest. Dependency detection may come to be a rather knowledge intensive task, which is further complicated by incomplete and potentially inconsistent local views of the agents. Moreover, making timely decisions that lead to efficient coordination actions is also everything but trivial [10]. The problem becomes even more difficult when agents pursuing partially conflicting goals come into play [18]. In all but the most simple MAS, the instrumentation of these tasks gives rise to complex patterns of interactions among agents. The set of possible interactions is often called the *interaction space* of coordination.

From a *software engineering* perspective, coordination is probably best conceived as the effort of *governing the space of interaction* [1] of a MAS. When approaching coordination from a *design* stance, the basic challenge amounts to how to make agents converge on interaction patterns that adequately (i.e. instrumentally with respect to desired MAS features) solve the dependency detection and decision tasks. A variety of approaches to tackle this problem can be found in the literature. Multiagent planning, negotiation, organisational structures, conventions, norms, reputation management, and mechanism design, are just some of them. These approaches aim at shaping the interaction space either directly, by making assumptions on agent behaviours and/or knowledge, or indirectly, by modifying the *context* of the agents in the MAS environment. The applicability of these mechanisms depends largely on the characteristics of the coordination problem at hand, as we will outline in the next section.

Finally, let us insist that conceiving coordination as a means to rule and manage interaction in MAS is by no means opposing the dependency model. Rather, it shifts the focus of attention from representation to design. We will elaborate further on this software engineering perspective on coordination in Section 4.

2.2 Characteristics of Coordination Models

It is a commonplace that, the more open a multiagent environment, the more difficult it is to instill a sufficient quality of coordination (see also the chapter by Fredriksson, Gustavsson, and Ricci in this volume [7]). For instance, in a closed environment, as assumed traditionally by Distributed Problem-Solving Systems, agent behaviour is controlled at *design-time*. As the agent designer has full control over the agents, she can implement a coordination mechanism of her choice: if certain assumptions on the agents' behaviours are necessary, these can simply be "hard-coded" into the agent programs. A popular example is the original Contract-Net Protocol (CNP) [25]: volunteering services to a contractor relies on the assumption of benevolence from the side of the bidders, which can be easily achieved when agents are designed to follow the CNP. Thus, the space of interactions is completely determined at the time the MAS is built. In closed environments, design choices are usually driven by efficiency issues.

At the other extreme, in large-scale open networks like the Internet, agent behaviour is *uncontrolled*, so that very few assumptions can be made about agents' behaviours and their frequencies. In particular, it is almost impossible to globally foresee and to influence the space of potential agent interactions. Most probably, agents will behave in a self-interested fashion, which may help to anticipate some of their actions, and may provide some clues on how to design coordination strategies at the micro-level. Also, in certain "parts" of the open system it may be possible to influence the frequencies of behaviours by spawning new agents with desired characteristics, so as to improve the quality of coordination at the macro-level (e.g. [21]). Still, very little can be done in the general case, so that the main focus of design comes to be placed on other issues, such as security, and often at a lower level of analysis.

A promising approach to efficiently instill coordination in open systems is inspired by the notion of agent as a situated entity. It falls in between the two extremes (fully controlled vs. uncontrolled), by providing for a sort of *partially controlled* agent behaviour. Since the environment where agents and MAS live is partially under human control, agent interaction can be influenced by engineering the agent environment: in particular, agent *infrastructures* are typically exploited to shape the agent environment according to the engineers' needs. For instance, *coordination infrastructures* provide agents with coordination (run-time) abstractions embodying *coordination as a service* [28], that exert a form of partial *run-time control* over agent behaviour. Coordination infrastructures are not meant to influence agent behaviour directly, but they can affect agent actions. As a trivial example, a security infrastructure enforces a set of behavioural restrictions for potential users that implicitly bounds the admissible agent interaction histories: the effects of actions of agents on the environment are thereby constrained without limiting their deliberation capabilities. Given a range of different *coordination services* made available by an infrastructure, agents can freely choose a service based on their self-interest: once they register with a coordination service, however, the infrastructure will enforce compliance with behavioural restrictions. This may be achieved by executing mobile agents on specific virtual machines, or by making agents interact through "intelligent" communication channels that govern the interaction space by filtering, modifying, or generating certain messages. Future agent-based auctions may become examples of such coordination services.

This latter form of coordination is often termed *mediated coordination* – which in general could rely on either a distinguished middle agent [9] or a coordination abstraction provided by an infrastructure (like a Linda tuple space [8], or a ReSpecT tuple centre [16]). In the literature, mediated coordination is often confused with *centralised coordination*. In fact, another important dimension of coordination models amounts to whether they can be designed in a *centralised* fashion, or need a *decentralised* instrumentation. While centralised mechanisms are appropriate for closed environments with design-time coordination, decentralised mechanisms (like peer-to-peer models) better satisfy the needs of open environments with run-time coordination. However, mediated coordination is often *multi-centric* – i.e., neither centralised, nor fully decentralised –, thus achieving a sort of welcome compromise between the engineering urges (pushing toward controlled and predictable systems) and typical features of the systems of today (emphasising openness, dynamics, and unpredictability).

Closely related to the above discussion is a concept recently introduced by Tolksdorf [27]: A coordination mechanism is said to be *coupled* if the effectiveness of an agent's coordination behaviour is based on assumptions on some (other) agent's behaviour. By contrast, *uncoupled* mechanisms impose no assumptions on agent behaviour. As truly decentralised coordination can only be achieved by a coupled mechanism, it bears the additional burden of ensuring that all involved agents will behave as expected.

Finally, when shifting our attention to the micro-level, the distinction between *quantitative* and *qualitative* models of coordination comes into play [18]. Qualitative approaches basically follow the dependency model outlined earlier, by directly representing the different "reasons" for preferring or not certain objects of coordination to others. An agent's coordination behaviour is guided by whether a certain local action (plan, goal, etc.) depends positively or negatively on the actions of others: it will choose its local and communicative actions based on the "structure" of dependencies that it shares with its acquaintances. So, in cooperative environments, it is straightforward to conceive coordination as a kind of *constraint satisfaction problem* [18]. In quantitative models, by contrast, the structure of the coordination problem is hidden in the shape of a multi-attribute utility function. An agent has control over only some of the function's attributes (i.e. some of the objects of coordination), and its utility may increase (decrease) in case there is a positive (negative) dependency with an attribute governed by another agent, but these dependencies are not explicitly modelled. Its local coordination decision problem then corresponds to a special type of *optimisation problem*: to determine a local action (plan, goal, etc.), and to induce others to choose local actions (plans, goals, etc.), so as to maximise its local utility. The quantitative approach may draw upon a well developed theoretical framework for both, cooperative settings (Operations Research) and non-cooperative settings (Game Theory), but suffers from that fact that, due to the uncertainties intrinsic to MAS domains, the utility function is only an approximation, so that its optimum need not coincide with an agent's actual best choice in the real environment. On the other hand, a coordination mechanism based on qualitative dependencies is less prone to such modelling inaccuracies, but its foundations go back to theories from social sciences (e.g. Social Psychology), that do not provide a sound formal framework to guide local decision-making.

3 Subjective *versus* Objective Coordination

Subjective and objective coordination are two different but complementary ways of conceiving coordination in MAS. We first sketch the different perspectives on coordination that give rise to these notions, and then provide an example for each of them, relying on a simple, closed environment for illustrative purposes. Finally, we discuss why both approaches are needed for the engineering of open agent systems.

3.1 Different Perspectives on Interaction

Basically, there are two ways to look at interaction: from the inside and from the outside of the interacting entities. When taking MAS into account, this comes to say that interaction within a MAS can be seen from the viewpoints of either an agent, or an external observer not directly involved in the interaction: respectively, the *subjective* and the *objective* viewpoints [24]. From the subjective viewpoint of an agent, the space of interaction roughly amounts to the observable behaviour of other agents and the evolution of the environment over time, filtered and interpreted according to the individual agent's perception and understanding. From the objective viewpoint, the space of interaction is basically given by the observable behaviour of all the agents of a MAS and the agent environment as well, and by all their *interaction histories* [30]. If coordination is taken to mean governing the interaction, then the two different viewpoints lead to two different ways of coordinating.

When looking at interaction from the agent's own point of view, coordination roughly amounts to *(i)* monitoring all interactions that are perceivable and relevant to the agent, as well as their evolution over time, and *(ii)* devising actions that could bring the overall state of the MAS (or, more generally, of the world) to (better) coincide with one of the agent's own goals. In short, this is what is called *subjective coordination*. So, in general, the acts of an agent coordinating within a MAS are driven by its own perception and understanding of the other agents' behaviour, capabilities and goals, as well as of the environment state and dynamics.

On the other hand, when looking at interaction within a MAS from outside the interacting agents, as an external observer, coordination means affecting agent interaction so as to make the resulting MAS evolution accomplish one or more of the observer's goals.[1] In short, this is what is called *objective coordination*. In general, the acts of external observers – whether they be MAS designers, developers, users, managers, or even agents working at the meta-level – are influenced not only by their perception and understanding of MAS agents and environment, but also by their a-priori knowledge of the agents' aims, capabilities and behaviour. Also, some forms of understanding and foreseeing of the global behaviour of the MAS as the result of the overall interaction amongst the agents and the environment are required to instill a coordination that is effective over time from the standpoint of the observer.

---

[1] Affecting agent interaction does not mean (directly) affecting agents. That is why objective coordination by no means contrasts with the fundamental notion of agent *autonomy* [32].

3.2 Subjective vs. Objective Coordination: An Example

This section aims at illustrating the subjective and objective viewpoints, using a reactive coordination scenario as an example. This scenario involves cognitive agents that develop short-term plans. Such a stance is appropriate for a variety of real world domains, such as agent-based decision support [4][20]. In the following, we will use a simple setting in the synchronous blocks domain [18], an extension of the well known blocks world, for illustrative purposes.

There are a table of unlimited size and four numbered *blocks*. Blocks can be placed either directly on the table or on top of another block, and there is no limit to the height of a stack of blocks. The only operations that can be performed therefore are to place a block x on the table (formally: $move(x, table)$), which requires x to be clear, or to place a block x on top of some other block y (formally: $move(x, y)$), which additionally requires y to be also clear. There is a *clock* that marks each instant of time by a tick. This scenario is synchronous, since all operations occur at a given tick, and the clock is the same for all operations. So, any operation could in principle be labelled by its tick, and any set of operations is totally ordered, since any operation can be said to occur before, after, or at the same time (tick) as any other operation. A *plan* of length k is a sequence of k operations performed successively. Instead of an operation, a plan may contain a NOP, indicating that nothing is done at a certain tick.

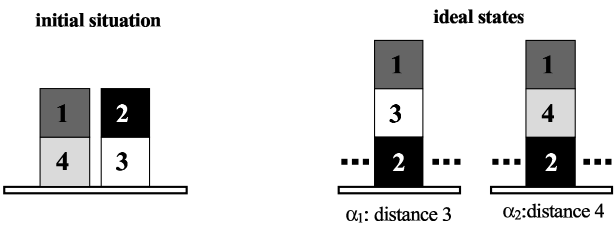

Fig. 1. A scenario in the synchronous blocks domain

Consider the scenario shown in Fig. 1. There are two agents, α_1 and α_2, both looking only two actions ahead, *i.e.* generating plans of length 2. In addition, suppose that the former is capable of moving all blocks but block 4, while the latter may manipulate all blocks but block 3: so, a plan by α_1 cannot include operations of the form $move(4, y)$, while a plan by α_2 cannot include operations of the form $move(3, y)$.

The initial situation and the states that each agent would ideally like to bring about are shown in Fig. 1. Both agents measure the "distance" to their ideal world states by the minimum number of actions they would need to perform in order to reach it. Table 1 shows some individual plans; "executability" denotes an agent's ability to enact a specific plan.

The subjective viewpoint in the example. In a single-agent environment, the agent's subjective attitude is straightforward: among the executable plans

Table 1. Some individual plans

plan	operation sequence	ability	executability
π_1	$[move(2, table), move(3, 2)]$	α_1	true
π_3	$[move(2, table), NOP]$	α_1, α_2	true
π_4	$[move(1, table), NOP]$	α_1, α_2	true
π_9	$[move(2, table), move(4, 2)]$	α_2	false
π_{10}	$[move(1, 2), move(4, 1)]$	α_2	true
π_{11}	$[move(2, 1), move(3, 2)]$	α_1	true
π_ε	$[NOP, NOP]$	α_1, α_2	true

that it can enact, it will choose the one that takes it as close as possible to its ideal state. Still, in the example α_1 and α_2 act in a shared environment, and may perform their plans simultaneously, i.e. in essence they perform a so-called *multiplan* [18]. Suppose plans are compositional, that is, the result of a multiplan μ be the "sum" of the effects of its component plans. If the component plans of a multiplan "interfere", the following rules apply:

- A multiplan μ is *not executable* if some component plans access the same block at the same tick in different ways, or if they obstruct a block that another component plan uses at a later tick.
- A multiplan μ is *executable* despite a non-executable component plan, if other component plans "complement" it, e.g. by providing a missing action.

Let us denote with (π_i,π_j) the multiplan where π_i and π_j are performed simultaneously.

Then, the multiplans (π_1,π_4), (π_4,π_9) and (π_3,π_4), for instance, lead to the world states shown in Fig. 2, respectively.

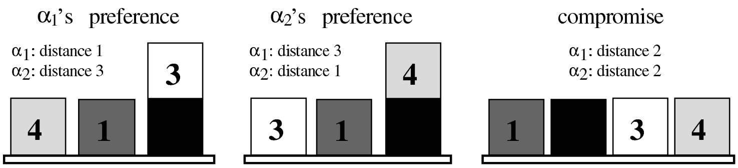

Fig. 2. Some outcomes of the execution of multiplans

Each agent just controls one component of the multiplan that is performed, so they have a (mutual) interest in coordinating their choices. It is then appropriate that they exchange a sequence of messages following some negotiation protocol, which defines the space of possible interactions for coordination. Obviously, the specific coordination interaction that actually occurs depends on the local choices of the agents when making offers and counteroffers respecting the multiplans to agree on. Supposing that agents are individually rational, these choices should depend on the agents' potential to influence each other, i.e. how far they may manipulate the outcome of their acquaintances' individual plans.

In the example, three classes of such social dependencies can be identified. First, there is a *feasibility-dependence* of agent α for a plan π with respect to α' if the latter can invalidate the plan, i.e. if it can turn down the execution of π. In the example, each agent is in a feasibility-dependence to the other for all plans shown in Table 1 except π_ϵ. Second, agent α is *negatively dependent* for a plan π with respect to α', if α' can deviate the outcome of π to a state that is less preferred by α. If α' can bring about a change in the outcome of α's plan π that α welcomes, then α is *positively dependent* on α'. In Table 1, each agent depends positively on the other for π_3 and π_4.

So, in essence, designing a coordination mechanism from a subjective point of view means

- to design a social reasoning mechanism that detects these dependence relations
- to endow agents with a decision algorithm that, taking into account the different dependence relations, guides the agent's decision making during coordination interactions (negotiation).

For instance, a social reasoning mechanism could make α_1 understand both negative and positive dependencies for multiplans with respect to α_2. Also, a decision algorithm could lead α_1 and α_2 to choose a multiplan like (π_3,π_4), producing a compromise between the ideal states of α_1 and α_2 (Fig. 2). Both represent forms of subjective coordination, where the agents reason, plan and act in order to make the global effect of agent interaction achieve their desired state of the world.

The objective viewpoint in the example. Suppose now that we are the designers of the global MAS in the synchronous blocks world. We are looking at the agents interactions from the outside, with the aim of obtaining certain global results (i.e. prefer certain classes of multiplans to others). From such an objective stance on the coordination, we need to tackle two aspects of coordination:

- a model of the outcome of coordination in the present multiagent environment is needed. What is the present space of interactions, and what will be potential instances of interactions and their outcomes?
- a mechanism to modify the space of interactions in a principled way is required. How can we "shape" the space of interactions and promote/hamper certain types of interactions, so as to influence the outcome of coordination in a desired direction?

So, based on the expected outcome of agents coordinating from their subjective viewpoints, and our mechanisms to affect the agents' interaction space and its global effects from an objective point of view, we can "tune" coordination from an objective stance.[2]

[2] Obviously, this is not to say that the objective stance always prevails over the subjective point of view. In fact, in open environments a change in objective coordination laws (e.g. tax laws) will in turn have repercussions in the agents' subjective coordination strategies (e.g. the tax payers' behaviour).

In the synchronous blocks world domain example, the first aspect is best tackled from within a *quantitative* framework. Suppose that for an agent the utility of a multiplan is given by the reduction of distance to the agent's ideal state. The utilities that each agent obtains from a multiplan can be represented in a vector. In the example, the utility vectors of multiplans (π_1,π_4), (π_4,π_9) and (π_3,π_4) for α_1 and α_2 are (2,1), (0,3) and (1,2), respectively. Agents may agree to gamble over multiplans, i.e. decide to "flip a coin" in order to choose between alternative courses of action. In this case we use the expected utility, i.e. the sum of each multiplan utility weighed by its probability. So, the utility of a mixed multiplan that enacts the above three multiplans with equal probability would be (1,2).

Still, agents are autonomous, so they are not forced to reach an agreement. Their interaction for mutual coordination may as well end up in conflict, which is equivalent to the decision not to (explicitly) coordinate their actions, and instead take the chances individually. In this case, an agent must account for the unpleasant eventuality that all its acquaintances could jointly try to harm it. So, a rational agent may choose the best individual plan that is "autonomous", i.e. whose executability cannot be turned down by others. For instance, the only plan that α_1 can execute and which is guaranteed not to become incompatible is π_ε, which α_2 counters by π_{10}, resulting in a conflict utility of -2 for α_1. Agent α_2 also needs to choose π_ε in case of disagreement, to which α_1's most malicious response is to enact π_{11}, giving rise to a conflict utility of -1 for α_2. So, the conflict utility vector in our example is $\boldsymbol{d} = (-2, -1)$.

The set S of utilities of all feasible (mixed) multiplans, along with the disagreement point, describes the so-called *bargaining scenario* [26]. In our two-agents example, the bargaining scenario can be represented in a plane, where each axis measures the utility of one agent (Fig. 3).

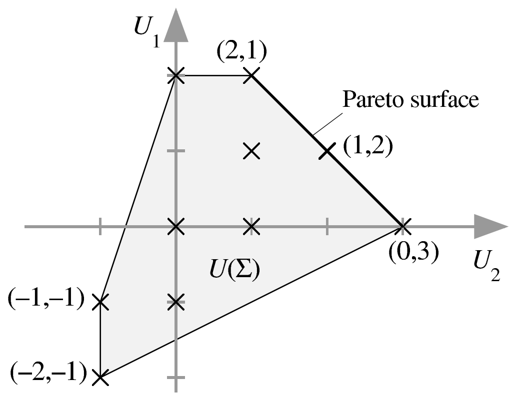

Fig. 3. Graphical representation of the example scenario

In order to model the outcome of subjective coordination among agents from a global point of view, the whole mathematical apparatus of bargaining theory is now applicable. In particular, we can apply the Nash bargaining solution [12] which states that rational agents should agree on a multiplan that maximises the product of gains from the disagreement point. In our example, this leads to a solution vector of (1,2). Consequently, our model predicts that the outcome of subjective coordination is a (mixed) multiplan whose utility is given by this solution, e.g. an agreement on the "compromise" multiplan (π_3,π_4).

Still, from a designer's point of view, the outcome of coordination-oriented interactions that are driven entirely by the subjective interests of the involved agents does not necessarily lead to the desired characteristics and functionalities of the MAS as a whole. As a consequence, a mechanism for objective coordination is required in order to suitably bias the coordination process and its outcome toward the achievement of the global MAS goals.

For instance, a simple objective coordination technique in the synchronous blocks domain relies on the ability to issue permissions or prohibitions for certain agents to perform specific actions, and assumes that agents (cannot but) comply with these prescriptions. Suppose it is forbidden for all agents to put block 1 on block 2. As a result, agent α_2 can no longer execute plan π_{10}, whereas α_1 is permitted to enact π_4. So, the worst situation that α_2 can bring about in the eyes of α_1 is to put block 4 on top of 2. The conflict utility of α_2 remains unchanged, so that the disagreement point changes in favour of α_1 to $\boldsymbol{d} = (-1, -1)$. Because of the change in $\boldsymbol{d}$, the outcome of subjective coordination interaction will switch to $(1.5, 1.5)$, which is reached by randomising equally between "compromise" (π_3,π_4) and α_1's favourite (π_1,π_4). If it was permitted for agent α_2 to unstack block 1 at the first tick (i.e. all plans that manipulate the left stack in the first step are forbidden for α_1), the worst thing that α_1 can still do is to obstruct block 2 for α_2 by putting block 4 on it in the second step, giving rise to a disagreement point which moves towards α_2. The solution utility vector then changes to $(0.5, 2.5)$, which is reached by selecting α_2's favourite (π_4,π_9) and compromise (π_3,π_4) both with probability $p = 0.5$.

Notice that the outcome of coordination has been changed without prescribing agents exactly what to do: it is just the space of possible agreements that has been modified. Still, this is not of much help as long as we do not have a means to estimate the effects of such changes in the interaction space. We would like to ensure that, as in the examples, if an agent is favoured/harmed by prescriptions and its "negotiation position" is strengthened/weakened, then this should be reflected in the agreement reached. In fact, this is always the case given the above mechanism. Issuing permissions and prohibitions shifts the disagreement point towards the favoured agents, while the property of *disagreement point monotonicity* of the Nash bargaining solution ensures that the corresponding change in the coordination outcome correlates with the movement [26]. The outcome of objective coordination can actually be computed based on a *coupled* approach using a distributed constraint optimisation algorithm [19].

3.3 Impacts on the Engineering of Agent Systems

As illustrated by the previous example, roughly speaking, subjective coordination affects the way in which individual agents behave and interact, whereas objective coordination affects the way in which interaction amongst the agent and the environment is enabled and ruled. So, while the main focus of subjective coordination is the behaviour of agents as (social) *individuals* immersed in a MAS, the emphasis of objective coordination lies more on the behaviour of a MAS as a whole.

When defining the architecture and the inner dynamics of individual agents, the subjective viewpoint on coordination is clearly the most pertinent one. How other agents' actions are represented and foreseen, how to interpret and handle shared information in the agent system, when and why to move from an agent environment to another, and so on – all these issues concern subjective coordination, and affect the way in which the agents of a MAS are designed, developed and deployed as individual entities. For instance, agents α_1 and α_2 in the example should be designed to have some planning ability, to share goals, and to be able to understand mutual dependencies to make it possible for them to reach at least the compromise outcome of Fig. 2. So, the viability of approaches adopting a subjective coordination viewpoint to the engineering of MAS strictly depends not only on the mental (reasoning, planning and deliberation) capabilities of the agents, but also on their ability to foresee the effect of their actions on the environment, the behaviour of the other agents, and the overall dynamics of the environment as well.

On the other hand, since in principle an external observer does not directly interact with the agents of a MAS, some capability to act on the space of MAS interaction without dealing directly with agents is obviously required in order to enable any form of objective coordination. Given that agents are typically situated entities, acting on the agent environment makes it possible to affect the behaviour of an agent system without having to alter the agents themselves. Objective coordination therefore deals with the agent environment: modifying the virtual machine supporting agent functioning, changing resource availability and access policies, altering the behaviour of the agent communication channel, be it virtual or physical, and so on – all these are possible ways to influence and possibly harness the behaviour of a MAS without directly intervening on individual agents. The viability of objective coordination in the engineering of agent systems depends then on the availability of suitable models of the agent environment, and on their proper embodiment within agent infrastructures. There, objective coordination would conceivably take on the form of a collection of suitably expressive coordination abstractions, provided as run-time coordination services by the agent infrastructure. In order to be able to formulate and ensure global properties of a MAS, the behaviour of coordination abstractions in response to events in the agent interaction space should be well-known and predictable.

To this end, the agents in the example could either willingly accomplish the prescriptions about blocks, or e.g. be forced to do so by a suitable infrastructure implementing security policies. For instance, blocks might embody security mechanisms, and directly handle permissions, thus bounding the space of admissible interactions, or, in other terms, shaping the interaction space of the agents. Or, blocks may be accessible only by means of virtual arms provided by the hosting infrastructure, and moves would be provided as services by the infrastructure itself. Permissions would be handled by the infrastructure, that would then implement what we call *coordination as a service*, that is, coordination provided as a service to agents by the infrastructure through a run-time coordination abstraction [28]. In both cases, agent coordination would be influenced by something external to the agents.

Also, it might be the case that permissions are explicitly available to the agent's inspection. Rules governing access to resources (blocks, in the example) might be explicitly represented and stored so as to be inspectable to agents. Agents may then read permissions, understand them, and possibly plan their course of action accordingly, thus obtaining a clear benefit by their increased knowledge of the environment resources. Inspectability of coordination rules (permissions, in the example) is where subjective and objective coordination meet. Roughly speaking, explicit representation of coordination laws, externally to agents, is an objective form of coordination, while their interiorisation and use by agents is obviously a form of subjective coordination. Intuitively, then, this example shows the benefits of blending together both subjective and objective coordination in the engineering of MAS.

On the one side, in fact, a purely subjective approach to coordination in the engineering of agent systems would endorse a mere reductionistic view, coming to say that agent systems are compositional, and their behaviour is nothing more than the sum of the individual's behaviour – an easily defeasible argument, indeed. Among the many consequences, this would require global properties of the agent system to be "distributed" amongst individuals, providing neither abstractions nor mechanisms to encapsulate such properties. As a result, the purely subjective approach would directly entail lack of support for design, development, and, even more, deployment of agent systems' global properties – which would result in substantial difficulties for incremental development, impractical run-time modification, and so on. On the other side, a purely objective approach to coordination in the engineering of agent systems would endorse a rough holistic view – where only inter-agent dependencies and interactions count, and individuals' behaviour has no relevance for global system behaviour. Among the many consequences, this would stand in stark contrast with any notion of agent autonomy, and would prevent agents from featuring any ability to affect the environment for their own individual purposes – no space for anything resembling an agent left, in short.

In the end, all the above considerations suggest that any principled approach to the engineering of agent systems should necessarily provide support for both subjective and objective models of coordination, possibly integrating them in a coherent conceptual framework, and providing at the same time a suitable support for all the phases of the engineering processes – in terms of coordination languages, development tools, and run-time environments. According to that, in the next section we discuss the role of coordination models (both subjective and objectives) in the context of agent-oriented methodologies for the engineering of software systems.

4 Coordination in Agent-Oriented Software Engineering

Coordination is a key issue for any Agent-Oriented Software Engineering methodology (AOSE). The notions of objective and subjective coordination may come to play a crucial role in this respect, in particular if we follow the notion of coordination as a service. In the sequel, we outline some basic concepts of AOSE and discuss their relation to coordination services and their supporting infrastructures. We then show how subjective and objective coordination models are put to use in the engineering of a MAS for the domain of workflow management.

4.1 Agent Abstractions in System Engineering

The ever-growing complexity of modern systems has raised the requirements on engineering models and processes up to an unprecedented level: new abstractions and practices are required to deal with open, heterogeneous, distributed and highly dynamic systems. In this perspective, Agent-Oriented Software Engineering is a new research area meant to exploit powerful agent-based metaphors and abstractions in the engineering of complex software systems [3,33]. Notions like *agent*, *agent society*, and *agent environment* constitute the basis for many of the AOSE methodologies currently being developed, such as Gaia [34] or SODA [14]. With respect to current best practice – often based on object-oriented technologies and methodologies – agent-based approaches feature two main properties: the autonomy of the main components (agents are autonomous entities encapsulating control), and the promotion of interaction to a first-class issue (agents are typically social and situated entities).

Autonomy of agents typically takes on two forms, at different levels of abstractions: at the higher level, agents are assigned to *tasks* that they pursue in an autonomous way; at the lower level, agents are autonomous and independent *loci* of control. Task accomplishment drives the flow of control inside the agents, so that a task can be regarded as the high-level abstraction to manage the low-level mechanisms of control. By moving the focus of system design up from *control dependencies* to *task dependencies*, autonomy of agents is then what allows to abstract away from control. Since one of the main sources of complexity in the design of open, distributed, highly dynamic systems is the coupling of control flows between different components, autonomy plays a key role in making agents suitable abstractions for the engineering of complex systems.

Agents are also typically acknowledged to be *social* entities, coexisting with other agents in a MAS and interacting with them in order to accomplish their own task. A group of agents interacting within a MAS is often referred to as a *society*, in particularly when their mutual interaction and communication is headed (either intentionally or not) toward the achievement of a global (*social*) functionality. From this viewpoint, then, societies of agents also serve as powerful design abstractions, to be handled as first-class entities throughout the whole engineering process. In particular, societies can be charged with tasks that could not (for either theoretical or practical reasons) be assigned to individual agents, which are then to be achieved by the overall coordinated activity of the individual member agents [14].

Finally, agents are *situated* entities, immersed in an environment containing not only other agents, but also resources and services. Resources might range from basic survival (such as an agent virtual machine) and inter-operability technologies, to complex coordination middleware. In fact, the activity of any agent (and of any agent society as well) could not be thought of or modelled without considering the environment wherein it lives – so that the agent environment calls for suitable first-class engineering abstractions on its own, modelling how each agent perceives the world around, and how it can predict its evolution over time.

Both task assignment to agents and interaction with the environment are typically mediated and represented by the notion of *role*. In fact, tasks are typically assigned to roles, that are assumed by agents either statically at design time, or dynamically at run-time. When playing a role, an agent is in charge of the corresponding task, and is entitled of all the authorisations and permissions (and limitations as well) pertaining to its role. Roles typically determine which part is played by agents within a MAS interpreted as an *organisation*, so they might be hierarchically related, or generally define some dependency between agents. In general, the interactions between the different roles have to follow specific rules for the overall organisation to work correctly and efficiently toward a shared global goal.

In all, the engineering process as promoted by agent-oriented abstractions is basically *task-oriented*: individual tasks are assigned to agents, and drive the design and development of individual agents, social tasks are assigned to groups of agents (societies) organised so as to pursue a common goal. In principle, we no longer delegate control to components (as in object-oriented and component-based practice), instead we delegate tasks, and with them, *responsibility* – while agents and societies encapsulate control. As a result, the design of agents and MAS is driven by tasks: individual tasks drive the design of individual agents, whereas social tasks drive the design of agent interaction protocols and supra-agent (social) rules.

4.2 Coordination in the Engineering of Agent System

Both autonomy and interactiveness – as novel features of agent-oriented software engineering – raise coordination issues. On the one hand, the ability of autonomously pursuing the accomplishment of a task requires agents to be able at least to represent the environment and foresee its evolution, to model and somehow predict the behaviour of the other agents and the effects of their own actions as well. This is of course the domain of subjective coordination approaches, as discussed in the previous sections. On the other hand, agent interaction within a society or with the environment asks to be first enabled, and then governed, in order to make the global agent system behave as required. These are actually interoperability and coordination issues, respectively, that straightforwardly fall under the umbrella of objective coordination models and technologies.

It is then quite natural here to emphasise the role that coordination models and technologies are going to play in the engineering of agent systems. This is particularly evident when considering the engineering of agent societies. Societies can be modelled as the sum of the individual components plus the social rules, or norms, that govern their behaviour and interaction. Once social norms are interpreted as coordination laws, coordination media can be easily acknowledged as social abstractions, around which agent societies can be modelled and built. As a result, agent societies can be easily thought of as the sum of individual agents and coordination abstractions, encapsulating norms in terms of coordination laws.

So, designing an agent society basically amounts to defining the social task, designing the individual and social roles to be assigned to individual agents as well as the corresponding interaction protocols, and then defining the norms ruling the collective behaviour. Subsequently, as argued in [14], suitable models, patterns, and mechanisms, of coordination can be chosen, that are expressive enough to capture the defined norms. For instance, in a dynamic setting, both the agents and the rules are likely to change over time: agents may join and leave the society at will, rules may vary to account for a new goal to achieve, or for a change in the environmental conditions: correspondingly, coordination abstractions should support the dynamic modification of coordination laws.

For a principled approach to the construction of agent systems, design abstractions should be supported throughout the whole engineering process, from design to development and deployment. In particular, when continuous development and incremental refinement are mandatory, design abstractions should be straightforwardly mapped upon run-time abstractions. This first calls for the availability of specialised coordination-specific IDEs to monitor and inspect the space of agent interaction, equipping developers with tools mapping abstractions into manageable metaphors – in particular, social ones [6]. Then, suitable coordination infrastructures are required, providing coordination abstractions as run-time reification of agent societies, and supporting the chosen model of coordination for social norm representation and enforcement. For instance, dynamic modification of coordination laws should obviously not only be foreseen by the model, but also suitably supported at execution time.

4.3 A Case Study: Agent Coordination in Workflow Management

A typical example where both subjective and objective coordination models and technologies come at hand is workflow management. In [22], an example of Workflow Management System (WfMS) in a Virtual Enterprise (VE) is presented and discussed in detail: workflow is first shown to be amenable for an interpretation as an agent coordination problem, then a WfMS is built by exploiting the TuCSoN coordination technology. In the following, we summarise the example as well as the main issues it raises, showing how the adoption of suitable models and technologies for agent coordination can make the engineering of complex systems easier, particularly when they promote the fruitful coexistence of both subjective and objective approaches.

The virtual bookshop. In [22], a *virtual bookshop* is presented, that is, a VE that aggregates several companies of different sorts to sell books through the Internet. In this scenario, four sorts of companies were identified: the *bookseller* (who provides the books), the *carrier* (who delivers books from sellers to customers), the *interbank service* (which executes payment transactions), and the *Internet service provider* (the Web portal for customers). In the example, two workflows were defined, that imply the specification and execution of classic workflow rules, such as managing a sequence of activities, an AND-Splitting process (coordinating activities started at the same time and executed in parallel) and an AND-Joining process (requiring synchronisation of parallel activities):

– the purchase of a single book from a specific bookseller, and
– the purchase of a set of books from different booksellers.

The first workflow describes the sequence of activities involved in the purchase of a specific book. First, order information is gathered from the Web site used by the customer – any of the sites provided by any Internet service provider participating in the VE. Then, the activity to get the book ready at the chosen bookseller's starts. When the book is ready, the dispatching activity is executed, to have the book delivered from the bookshop to the customer by a carrier. Finally, after the book has been delivered to the customer, the payment transaction activity is executed, involving the intervention of an interbank service to transfer money from the customer to the VE.

The second workflow involves the purchase of a set of books from (possibly) different booksellers. After order information is gathered, a number of book acquiring activities are executed in parallel, each aiming at getting a book from a specific bookseller. When all the books are ready at the involved booksellers', the dispatching activity can start as in the first case (for the sake of simplicity, the same carrier is assumed to be used to deliver all books).

WfMS as an agent system. The main element of a workflow is the *task*, as the elementary unit of work inside a workflow. A workflow *schema* is the collection of tasks that collectively achieve the goal(s) of a *process*. Tasks are interrelated into a flow structure via *connectors*, such as split and join elements, which define the execution dependencies among tasks (the order in which tasks can be executed). A *case* is an execution of a workflow schema, i.e. an instance of the corresponding (business) process.

According to the standard WfMC approach [31], workflow management is modelled in terms of *workflow engines* (workflow servers or coordinators, where coordination takes place) and *workflow participants* – thus straightforwardly inducing a clear separation between coordinating and coordinated activities. Workflow tasks are executed autonomously by (artificial or human) agents, coordinated by the workflow engines: in particular, agents can be used to represent the workflow participants whose activity can be automatised. Each workflow is then modelled as an agent society, involving individual agents autonomously pursuing workflow tasks, while the society, built around the workflow engine, aims at bringing the whole workflow to a successful end.

In the example, the main agents involved in the workflows are:

– *interface agents*, responsible for collecting information about customers and orders. These agents also interact with customers during order execution, informing them about order status and possible problems.
– *buyer agents*, responsible for ordering books from the booksellers and getting them ready for delivery.
– *delivering agents*, responsible for delivering the books to the customers, provided that the books are ready at the bookseller's. To do so, they inform the carrier of each new dispatch to do, and monitor the delivery until the book is in the customer's hands.
– *payment agents*, responsible for getting the payments done. These agents interact with an interbank service for transferring money from customers to the virtual bookshop.

Workflow engines govern the activity of the workflow participants, encapsulating the workflow rules. So, coordination media can be used as workflow engines to automatise coordination, embodying the workflow rules in terms of coordination laws: in this way, the workflow rules can be encapsulated outside agents and somehow superimposed on them, so that agents are entitled to autonomously pursue their task with no need to be aware of the global workflow. In particular, in the example, logic *tuple centres* are used as coordination media [16], that is, tuple spaces whose behaviour in response to communication events can be specified through ReSpecT logic programs – so that a ReSpecT specification of a tuple centre actually defines the laws of coordination that the tuple centre embodies [15].

The agent/engine interaction occurs either when an agent declares to be ready to execute a specific task (according to its role), or when an agent returns the result of the task execution (or communicates the related problems). In our example, interaction takes the form of insertion and retrieval of tuples in `vbs` tuple centres, exploiting Linda-like coordination primitives [8]: agents check for new tasks to execute by trying to retrieve (by means of an `inp` operation) `ready_to_do` tuples, and provide task results by inserting (by means of an `out`) `task_success` or `task_failure` tuples.

Mapping a workflow engine onto a tuple centre essentially means to define the event model and the process model [2]. The event model defines what kinds of internal and external workflow events must be considered. Internal events are the ones defining the normal flow of activities within the workflow, such as temporal events related to the starting and the termination of a task. External events are events that can have an impact on the regular evolution of a running case, but are outside the control of the WfMS. In the example, both internal and external events are captured and represented by *reifying* them as descriptive tuples in the tuple centre. For instance, when a case `CaseID` is terminated, the tuple `case_done(CaseID)` is inserted in the tuple centre.

In turn, the process model describes the behavioural aspects of a workflow specification, from its initial state to one of its final states. In our example, the coordination activities of a workflow engine are naturally mapped upon the behaviour of a tuple centre. In particular, part of the rules, implemented

Table 2. ReSpecT code for the workflow meta-rules

```
reaction(out(case_to_start(CaseName, Info)),(
  in_r(case_to_start(CaseName, Info)),
  in_r(case_id_counter(ID)),
  NextID is ID + 1,
  out_r(case_id_counter(NextID)),
  out_r(case_started(ID,CaseName, Info)),
  out_r(case_state(ID, executing, _)),
  out_r(next_task(ID, CaseName, case_start,_)))).

reaction(inp(ready_to_do(TaskType, _, _)),(
  pre,
  in_r(task_to_do(CaseID, TaskType, Info)),
  in_r(task_id_counter(TaskID)),
  NextID is TaskID + 1,
  out_r(task_id_counter(NextID)),
  current_agent(ExecutorID),
  out_r(ready_to_do(TaskType, Info, TaskID)),
  out_r(task_started(TaskID, CaseID, TaskType, Info, ExecutorID)))).

reaction(out(task_success(TaskID, Info)),(
  in_r(task_success(TaskID, Info)),
  out_r(task_done(TaskID, Info)))).
reaction(out_r(task_done(TaskID, Info)),(
  rd_r(task_started(TaskID, CaseID, TaskType, Info, _)),
  rd_r(case_started(CaseID, CaseName, _)),
  rd_r(case_state(CaseID, executing, _)),
  out_r(next_task(CaseID, CaseName, TaskType, Info)))).
reaction(out_r(next_task(CaseID, CaseName, TaskType, Info)),(
  in_r(next_task(CaseID, CaseName, TaskType, Info)))).

reaction(out_r(case_done(CaseID)),(
  in_r(case_state(CaseID, executing, _)))).
```

as ReSpecT reactions, constitute a basic core of meta-workflow rules, acting as the instructions of a *workflow virtual machine*: they are independent of the specific workflow, and are used to define standard workflow behaviour triggered by internal workflow events, making it possible to define the basic skeleton of the flow structure. These (meta-)rules interpret/execute other rules (trigger other ReSpecT reactions in our case), which are specific to the given workflow type and constitute the remaining part of the rules defining the process model. Taken from [22], Table 2 shows the ReSpecT code for the workflow meta-rules, while Table 3 shows the ReSpecT code for the single book purchase workflow.

Features. Adopting coordination media like tuple centres as the workflow engines provides, in the first place, all the benefits of objective coordination approaches, and, in the second place, a most effective support for subjective coordination techniques.

Due to mediated interaction, workflow participants are loosely coupled, and may interact with no need to know each other, or to coexist in space or in time. Then, workflow rules are encapsulated within suitable abstractions outside the agents – the tuple centres –, so that workflow participants are enabled to autonomously participate in the workflow with no need to be aware of the global workflow rules or dynamics. This also means that agents acting as workflow par-

Table 3. ReSpecT code for the single book purchase workflow example

```
reaction(out_r(next_task(CaseID, single_book_purchase, case_start,_)),(
  rd_r(case_started(CaseID,single_book_purchase,info(_,Book,Seller,_))),
  out_r(task_to_do(CaseID, buy_the_book, info(Book,Seller))))).

reaction(out_r(next_task(CaseID, single_book_purchase, buy_the_book,
                           BookReceipt)),(
  rd_r(case_started(CaseID,single_book_purchase,info(Customer,_,Seller,Carrier))),
  out_r(task_to_do(CaseID, dispatch_the_book,
                      info(BookReceipt,Carrier,Seller,Customer))))).

reaction(out_r(next_task(CaseID, single_book_purchase, dispatch_the_book,
                           CustomerReceipt)),(
  rd_r(case_started(CaseID,single_book_purchase,info(Customer,Book,_,_))),
  out_r(task_to_do(CaseID, make_payment, info(Customer,Book))))).

reaction(out_r(next_task(CaseID, single_book_purchase, make_payment,
                           BankReceipt)),(
  out_r(case_done(CaseID)))).
```

ticipants are amenable to independent design and development, and, dually, that changes in the workflow rules do not necessarily require changes to all agents. Finally, both the state of the interaction and the laws of coordination are explicitly represented as logic tuples in the tuple centre (respectively, ordinary tuples, and meta-level ReSpecT tuples), that can then be inspected and dynamically modified by both engineers and (intelligent) agents, at least in principle.

On the other hand, explicit representation and inspectability of both the communication state and the coordination state are the features of objective coordination models (like ReSpecT tuple centres) that build a bridge toward subjective approaches, thereby making them particularly effective. In fact, since agents can perceive the state of the interaction, an intelligent cooperative agent could in principle monitor the state of the workflow and act to improve the overall workflow performance or results, either starting an activity by itself, or by properly stimulating other cooperative (but possibly not-so-intelligent) agents. Also, since coordination laws are inspectable and dynamically modifiable, an intelligent supervisor agent could in principle reason about the state of the workflow and its rules, and possibly change them according to current needs – for instance, when external conditions have turned the previously established workflow ineffective – thus paving the way for dynamically adaptive systems.

All in all, this example clearly suggests that the concerted exploitation of both objective and subjective approaches in the coordination of agent systems could provide a well defined path to the construction of dynamic, adaptive, and intelligent systems.

5 Discussion

In this chapter, we have given a brief overview over current conceptualisations, models and support infrastructures for coordination in MAS. Among the many dimensions of coordination models, the duality between the objective and the subjective viewpoint on coordination has been identified as particularly important for the construction of modern MAS, an idea that has been illustrated using a simple multiagent blocks world domain as an example. We have stressed the fact that such an approach leads quite naturally to the idea of conceiving coordination as a service that multiagent coordination infrastructures may provide. Finally, we have argued that this stance can be smoothly integrated into current Agent Oriented Software Engineering methodologies, and provided evidence for that claim by an example from the domain of agent-based workflow management.

The question of how to design effective coordination mechanisms for open environments is a major research issue in the agent community. A popular approach to this problem is to complement mechanisms for closed systems with techniques that enforce initially uncontrolled agents to comply with the required "behavioural norms". Some self-enforcing approaches have been borrowed from Game Theory and Economics (e.g. Vickrey Auctions), but their applicability to many real-world domains appears to be limited. In this respect, work on social control and models of trust is promising, but much work still needs to be done in that direction. An effective instrumentation of findings from Evolutionary Game Theory and Social Simulation to instill desired global properties, e.g. by controlling the frequencies of certain behaviours in a open agent society, is still further ahead.

At the time being, the notion of *coordination as a service* appears most promising to us. Once agents have freely chosen a particular coordination service, the compliance with its "behavioural norm" can be enforced by the infrastructure of the service itself, e.g. through dedicated virtual machines or "intelligent" communication channels. Still, this idea requires advances on several fronts. Most importantly, more powerful service description languages are needed, that allow agents to dynamically choose between services based on their self-interest. In addition, what kind of coordination knowledge should be used in these services, and how to model, represent, and enact it within coordination media is still subject to debate. Finally, some considerations on how current AOSE methodologies could be extended in a principled manner to better capture the design abstraction of coordination services and its mapping to adequate run-time abstractions may lead to interesting results. In this regard, the chapter by Fredriksson, Gustavsson, and Ricci in the present volume [7] presents a current effort to develop a comprehensive methodology for the provision of sustainable coordination in open systems.

References

1. Nadia Busi, Paolo Ciancarini, Roberto Gorrieri, and Gianluigi Zavattaro. Coordination models: A guided tour. In Omicini et al. [17], chapter 1, pages 6–24.
2. Fabio Casati, Silvana Castano, Mariagrazia Fugini, Isabel Mirabel, and Barbara Pernici. Using patterns to design rules in workflows. *IEEE Transactions on Software Engineering*, 26(8):760–785, August 2000.
3. Paolo Ciancarini and Michael J. Wooldridge, editors. *Agent-Oriented Software Engineering*, volume 1957 of *Lecture Notes on Computer Science*. Springer-Verlag, January 2001. 1st International Workshop (AOSE 2000), Limerick, Ireland, 10 June 2000, Revised Papers.

4. José Cuena and Sascha Ossowski. Distributed models for decision support. In Gerhard Weiss, editor, *Multiagent Systems – A Modern Approach to Distributed Artificial Intelligence*, pages 459–504. MIT Press, 1999.
5. Keith Decker. TÆMS: A framework for environment centered analysis and design of coordination mechanisms. In Gregory M.P. O'Hare and Nicholas R. Jennings, editors, *Foundations of Distributed Artificial Intelligence*, pages 429–448. John Wiley and Sons, 1996.
6. Enrico Denti, Andrea Omicini, and Alessandro Ricci. Coordination tools for MAS development and deployment. *Applied Artificial Intelligence*, 16(10), November 2002.
7. Martin Fredriksson, Rune Gustavsson, and Alessandro Ricci. Sustainable coordination. Springer-Verlag, 2002. In this volume.
8. David Gelernter. Generative communication in Linda. *ACM Transactions on Programming Languages and Systems*, 7(1):80–112, 1985.
9. Matthias Klusch and Katia Sycara. Brokering and matchmaking for coordination of agent societies: A survey. In Omicini et al. [17], chapter 8, pages 197–224.
10. Victor R. Lesser. Reflections on the nature of multi-agent coordination and its implications for an agent architecture. *Autonomous Agents and Multi-Agent Systems*, 1(1):89–111, 1998.
11. Thomas Malone and Kevin Crowston. The interdisciplinary study of coordination. *ACM Computing Surveys*, 26(1):87–119, 1994.
12. John F. Nash. The bargaining problem. *Econometrica*, 28:152–155, 1950.
13. Allen Newell. Reflections on the knowledge level. *Artificial Intelligence*, 59:31—38, 1993.
14. Andrea Omicini. SODA: Societies and infrastructures in the analysis and design of agent-based systems. In Ciancarini and Wooldridge [3], pages 185–193. 1st International Workshop (AOSE 2000), Limerick, Ireland, 10 June 2000, Revised Papers.
15. Andrea Omicini and Enrico Denti. Formal ReSpecT. In Agostino Dovier, Maria Chiara Meo, and Andrea Omicini, editors, *Declarative Programming – Selected Papers from AGP'00*, volume 48 of *Electronic Notes in Theoretical Computer Science*, pages 179–196. Elsevier Science B. V., 2001.
16. Andrea Omicini and Enrico Denti. From tuple spaces to tuple centres. *Science of Computer Programming*, 41(3):277–294, November 2001.
17. Andrea Omicini, Franco Zambonelli, Matthias Klusch, and Robert Tolksdorf, editors. *Coordination of Internet Agents: Models, Technologies, and Applications.* Springer-Verlag, March 2001.
18. Sascha Ossowski. *Coordination in Artificial Agent Societies – Social Structure and its Implications for Autonomus Problem-solving Agents*, volume 1535 of *LNAI.* Springer, Berlin, 1999.
19. Sascha Ossowski. Constraint-based coordination of autonomous agents. *Electronic Notes in Theoretical Computer Science*, 48, 2001.
20. Sascha Ossowski, Josefa Z. Hernández, Carlos A. Iglesias, and Alberto Fernández. Engineering agent systems for decision support. In Paolo Petta, Robert Tolksdorf, and Franco Zambonelli, editors, *Engineering Societies in an Agent World III.* Springer-Verlag, 2002.
21. Sunju Park, Edmund H. Durfee, and William P. Birmingham. Emergent properties of a market-based digital library with strategic agents. *Autonomous Agents and Multiagent Systems*, 5:33–51, 2000.

22. Alessandro Ricci, Andrea Omicini, and Enrico Denti. Virtual enterprises and workflow management as agent coordination issues. *International Journal of Cooperative Information Systems*, 11(3/4):355–380, September/December 2002. Cooperative Information Agents: Best Papers of CIA 2001.
23. Jeffrey S. Rosenschein and Gilad Zlotkin. *Rules of Encounter – Designing Conventions for Automated Negotiation Among Computers*. MIT Press, 1994.
24. Michael Schumacher. *Objective Coordination in Multi-Agent System Engineering – Design and Implementation*, volume 2039 of *LNAI*. Springer-Verlag, April 2001.
25. Reid G. Smith. The Contract Net Protocol: High level communication and control in distributed problem solver. *IEEE Transactions on Computers*, 29(12):1104–1113, 1980.
26. William L. Thomson. Cooperative models of bargaining. In R. Aumann and S. Hart, editors, *Handbook of Game Theory*, pages 1238–1284. North-Holland, 1994.
27. Robert Tolksdorf. Models of coordination. In Andrea Omicini, Franco Zambonelli, and Robert Tolksdorf, editors, *Engineering Societies in an Agent World*. Springer-Verlag, 2000.
28. Mirko Viroli and Andrea Omicini. Coordination as a service: Ontological and formal foundation. In Antonio Brogi and Jean-Marie Jacquet, editors, *Foundations of Coordination Languages and Software Architectures – Papers from FOCLASA'02*, volume 68 of *Electronic Notes in Theoretical Computer Science*. Elsevier Science B. V., 2002.
29. Frank von Martial. *Co-ordinating Plans of Autonomous Agents*, volume 661 of *Lecture Notes on Artificial Intelligence*. Springer-Verlag, Berlin, 1992.
30. Peter Wegner. Why interaction is more powerful than computing. *Communications of the ACM*, 40(5):80–91, May 1997.
31. Workflow Management Coalition home page. `http://www.wfmc.org/`.
32. Michael J. Wooldridge and Nicholas R. Jennings. Intelligent agents: Theory and practice. *The Knowledge Engineering Review*, 10(2):115–152, 1995.
33. Michael J. Wooldridge, Gerhard Weiss, and Paolo Ciancarini, editors. *Agent-Oriented Software Engineering II*, volume 2222 of *Lecture Notes on Computer Science*. Springer-Verlag, January 2002. 2nd International Workshop (AOSE 2001), Montreal, Canada, 29 May 2001, Revised Papers and Invited Contributions.
34. Franco Zambonelli, Nicholas R. Jennings, Andrea Omicini, and Michael J. Wooldridge. Agent-oriented software engineering for Internet applications. In Omicini et al. [17], chapter 13, pages 326–346.

Sustainable Coordination

Martin Fredriksson[1], Rune Gustavsson[1], and Alessandro Ricci[2]

[1] Department of Software Engineering and Computer Science
Blekinge Institute of Technology
Box 520 – Ronneby, Sweden
{martin.fredriksson|rune.gustavsson}@bth.se
[2] DEIS, Università di Bologna a Cesena
Via Rasi e Spinelli, 176 – 47023 Cesena, FC, Italy
aricci@deis.unibo.it

Abstract. Coordination, accounting for the global coherence of system behaviour, is a fundamental aspect of complex multi-agent systems. As such, coordination in multi-agent systems provides a suitable level of abstraction to deal with system organisation and control. However, current coordination approaches in multi-agent systems are not always fully equipped to model and support the global coherence of open computational systems, i.e., multi-agent systems that are situated in complex and dynamic physical environments. We therefore emphasise the critical roles of observation and construction to sustain coordination in open systems. We present the methodological framework VOCS (*Visions of open computational systems*), exemplified in terms of a naval multi-agent system scenario (TWOSOME) and the tools explicitly developed and used in construction and observation of this system, SOLACE and DISCERN.

1 Introduction

A common vision and natural area of applicability for multi-agent systems is that of massively distributed and evolving computational environments, where the involved network of systems is considered to be interconnected in an autonomic and conjunctive manner [38]. In this scenario, network nodes are simply powered on and then automatically integrated into the surrounding fabric of interaction. The general message is that a prominent class of future multi-agent systems will rely on such conjunctive fabrics of interaction that are populated by personally tailored autonomic and conjunctive services, i.e. service-oriented multi-agent systems [15]. We account for such environments in terms of *open computational systems*[1] [14], i.e., physical environments where each agent — human and software alike — has the capability to access and interact with part or all of a networked and computationally empowered information system. From

[1] We later return to a more specific characterisation of open computational systems. In the current context, however, the concept should be considered as a complement to service-oriented multi-agent systems with a focus on mechanisms for agent discovery, integration, and interaction.

M. Klusch et al. (Eds.): Intelligent Information Agents, LNAI 2586, pp. 203–233, 2003.

a general perspective, open computational systems can be conceived as cooperating ensembles of individual agents and services, striving to achieve different missions. Consequently, *coordination* is a fundamental aspect, accounting for the local as well as global coherence of system behaviour.

We consider multi-agent systems to be a suitable level of abstraction when dealing with the complexity of coordination in open computational systems. However, we also find that current approaches to coordination in multi-agent systems are not rich enough to model and support the aspects of system coordination needed. Typically, this situation occurs when we aim at real-time observation, analysis, and instrumentation of complex system behaviour in open and dynamic environments, i.e., when frequent context-switching occurs. We therefore focus on a systemic perspective on modelling and support of coordination in open computational systems, i.e., *sustainable coordination.*

1.1 Background

In terms of recent classifications of coordination approaches in multi-agent systems [24,36], open computational systems should be considered to account for the exploitation of coordination mechanisms spanning from *subjective* to *objective* approaches (see the chapter of the present volume by Omicini and Ossowski for an in-depth discussion). The need to thoroughly understand the dualistic relationship between these two approaches has previously also emerged in the complex coordination scenarios of computer-supported cooperative work and workflow management systems [32,35,2,10]. On the one hand, when it comes to complexity of subjective coordination, the involved processes and required mechanisms are imposed exclusively upon the individual agents involved, i.e., each computational entity must reason

"...about its local actions and the (anticipated) actions of others to try and ensure the community acts in a coherent manner ..." ([17], p.187)

Approaches of subjective coordination are realized by means of high-level agent communication languages and conversation protocols that are supported by the individual agents themselves. On the other hand, in the case of objective coordination, the support for coordination is delegated to computational entities other than those participating in the coordinated behaviour. Such external coordination entities are provided either in the form of mediators [18] or by means of coordination media [6], i.e., *services.* In open computational systems, systemic aspects, such as dynamics and scalability, are of the essence. In contrast, subjective coordination by means of agent communication languages and protocols such as FIPA ACL or KQML [9,19], does not appropriately support the notion of openness [30,37]. In particular, these protocols are by definition syntax-based and therefore aim at local coherence of system behaviour. Such an approach towards coordination typically stems from a system perspective where a priori established patterns of interaction are assumed. However, in open computational systems coordination protocols must account for the global

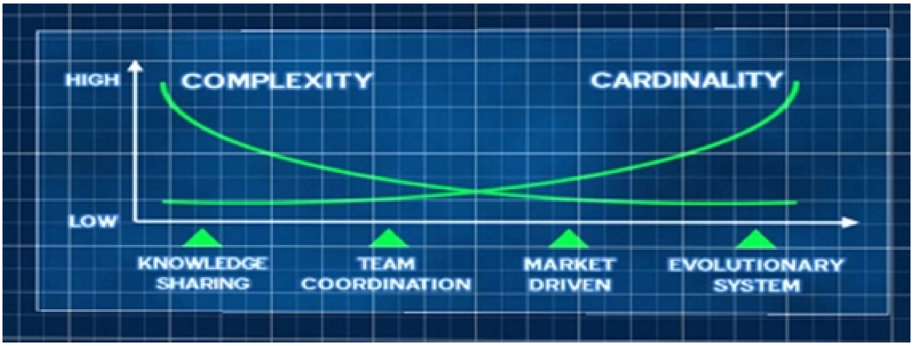

Fig. 1. *Complexity* vs. *Cardinality.* The two primitive dimensions involved in sustainable coordination, complexity and cardinality. Coordination approaches typically address systems comprised of small number of complex reasoning agents or a large number of simple actuators.

coherence of system behaviour, i.e., support a sustainable behaviour as a result from an unpredictable sequence of events in the surrounding environment.

Subjective approaches, building coordination exclusively on top of the reasoning capability of the individual agents and the semantics of their agent communication language, explicitly address issues of dynamics in closed systems [3]. These approaches often constrain the possible system population to involve only high-level logic-based agents applied in the context of social interaction, and then usually account for only relatively small sets of such complex agents. Such variants are therefore not applicable to system coordination in the context of medium and large domains [31]. In contrast, subjective approaches that rely on marked-based coordination [43] are better suited to scale with system complexity and heterogeneity [11,16,44]. However, these approaches are not always suitable to support required complex coordination activities and articulation of work, e.g., in the case of distributed workflow management.

On the opposite side of the spectrum we find objective coordination approaches, such as tuple based models [25], which, shifting the coordination burden (or part of it) from agents to some kind of service or artifact provided by the infrastructure, make it easier to deal with system complexity and heterogeneity. That is, objective approaches support coordination also for agents not necessarily characterised by high-level mental attitude or skills and explicitly account for scalability and engineering of system coordination by means of coordination media/artifacts. At the same time, however, objective coordination approaches contribute to increased system complexity in terms of the resources needed to support coordination and the required quality of service (e.g., fault tolerance or reliability).

1.2 Problem Statement

Subjective and objective coordination approaches each address their particular dimension of system characteristics, i.e., complexity in reasoning and cardinality of system population. Contemporary areas of applicability have already been shown in industrial manufacturing, pervasive computing, and *military operations* [4,28,30,31,21]. But when it comes to practical application scenarios of open computational systems, the two types of approaches taken individually cannot account for a sustainable behaviour in the dynamic environments involved. In response to this situation, so-called stigmergy coordination has been proposed [29] as a move towards a systemic characterisation of system behaviour, i.e., synthetic systems [31] where global coherence is of the essence. As is the case of open computational systems, coordination in synthetic systems plays a pivotal role. System features and capabilities are not considered to be embodied by any single agent but rather to emerge by the coordinated activities of a dynamic ensemble of entities. However, in order to survive in such dynamic environments, a particular entity necessarily requires the complex and autonomic capabilities of analysis and instrumentation. In effect, we consider sustainable coordination as dependent on modelling and support of dynamics in three interrelated constructs of (synthetic) open computational systems, namely *structures*, *patterns*, and *processes*: Dynamic sets of contexts, coordination mechanisms, and agents, affect the mediation structures, interaction patterns, and observable processes of a given system as a result from the temporary populations of agents taking part in the joint coordination of given missions.

From the perspective of open computational systems, the characteristics above call for the support of scalable coordination [11]. In particular, coordination activities in open computational systems are strongly *context-dependent* [5,23], being situated in unpredictable environments and influenced by the inherent dynamics, i.e., evolution of mediation structures, interaction patterns, and observable processes. These system characteristics and requirements of sustainable coordination are not addressed by individual coordination protocols [34] or models [6,27] available, and therefore require the integration of multiple approaches. The practical application scenarios found in open computational systems account for different levels of abstraction, both concerning the cardinality of system populations and the complexity in reasoning [11](see Fig. 1). Given the overall issue of sustainable coordination, we aim at exploring a methodological framework that explicitly addresses context-driven system behaviour. That is, we focus on the possibility of any agent to dynamically take the appropriate actions; not constrained by its relation to current coordination mechanisms but primarily as a result from observing its temporal context. To that end, we consider sustainable coordination to play a pivotal role in the successful deployment of multi-agent systems comprised by human as well as computational actors.

1.3 Approach

The primary issue addressed in this paper is to elicit in what way we can understand and respond to challenges and opportunities in the coordination of complex multi-agent systems, where sustainable behaviour in dynamic environments is of the essence. To this end, we introduce a methodological framework called *Visions of open computational systems* (VOCS) and focus on the following continuous interplay between articulation, construction, observation, and instrumentation:

- *Articulation.* The required agent population and the roles involved need to be articulated with respect to some particular mission and current context, i.e., according to our current situation, understanding, and intentions. This population is made up of (human and computational) actors and coordinators that must achieve a sustainable system behaviour in their evolving environment by means of interaction and cognitive capabilities.
- *Construction.* Domain specific system qualities are to be synthesised by means of autonomous agents and coordination artifacts. As the quality of the system is determined by its capability to take an evolving environment into account, multiple coordination services are usually needed by the involved agents. That is, the coordination mechanism required by a set of agents in one context does not necessarily match the mechanism required in another.
- *Observation.* In order to fulfil some particular role, an agent needs to take actions according to the evolving environment. This capability of an agent manifests itself in the activity of system analysis. Agents need the capability to observe their evolving environment and understand the current context they are situated in, so as to execute the most appropriate actions. This requires that the qualities of the system domain can be measured and quantified in a dynamic manner and thereafter put to effective use to change the agent's current context.
- *Instrumentation.* When one or more agents come to the conclusion that their current context is heading for an unacceptable behaviour, the system needs to be instrumented. This in turn requires that the process of articulation, construction, and observation is performed all over again as a matter of system adaptation. Consequently, a set of agents could therefore require different coordination mechanisms at different times.

The issue of sustainable coordination calls for a comprehensive methodology that accounts for both theoretical and practical aspects of open computational systems. Our proposal is based on our experience of engineering a complex multi-agent system as well as the basic tools used in the continuous articulation, construction, observation, and instrumentation of that particular system. The remainder of this chapter is organised as follows. In Section 2 our methodological approach toward sustainable coordination is discussed in detail, introducing crucial aspects of sustainable coordination. In Section 3 we present a layered perspective of open computational systems. This perspective is then used to characterise a practical application scenario (TWOSOME) of open computational systems currently under development at our research laboratory[2]. In Section 4

[2] For more information concerning the *Societies of computation laboratory* (SOCLAB), please visit the following URL: `http://www.soclab.bth.se`.

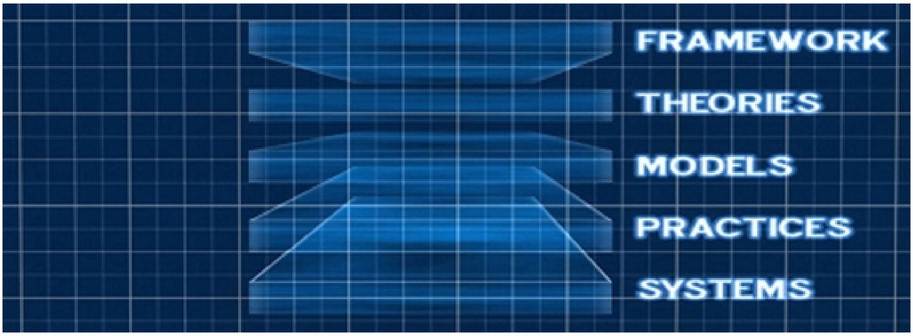

Fig. 2. *Methodology.* Sustainable coordination in open computational systems requires a comprehensive methodology. We need to address not only the meta-theoretical framework but also the basic theories and their corresponding models as well as the synthesis, analysis, and instrumentation of system behaviour by means of practical methods and tools.

we describe two different tools that were developed in response to requirements of system construction (SOLACE) and observation/instrumentation (DISCERN) of sustainable coordination; these tools were also used to produce all figures of this chapter, with the exception of Fig. 10. Finally, in Section 5 we summarise the material presented and point to currently planned research and development activities.

2 Sustainable Coordination in Theory

To understand the causes and effects of a particular system's coordinated behaviour in terms of articulation, construction, observation, and instrumentation, a methodological approach is required. In this endeavour it is useful to introduce different levels of abstraction and applicability. More precisely, we argue that it is useful to distinguish between the perspectives of (methodological) *frameworks*, *theories*, and *models*. A methodology collecting components from these three perspectives should be able to also support the *system* analyst and engineer with appropriate guidelines and tools for their work *practice* (see Fig. 2).

We will next characterise the focus of our methodological approach in terms of framework, theories, models, practices, and systems. We then take a closer look at the the continuous feedback loop of articulation, construction, observation, and instrumentation introduced above.

2.1 Visions of Open Computational Systems

A methodology can be considered to be comprehensive if the application of methods and tools, i.e., its practices, fully encompass the involved theories and established models of a particular system class and its corresponding framework.

In the following, we introduce the general outline of our comprehensive methodology for articulation, observation, construction, and instrumentation of open computational systems, namely *Visions of open computational systems* (VOCS).

– *Framework.* From a methodological perspective, frameworks attempt to identify the universal elements and concepts of some particular problem domain under study. That is, frameworks provide the meta-theoretic language to be used in the development of theories and establishment of models. Frameworks help to organise diagnostic and prescriptive inquiry and consequently indicate the list of common variables that can be used in analysis and instrumentation of a particular system's behaviour. The elements and concepts contained in a framework help the analyst to identify central questions that need to be addressed in developing scientific theories. For the purposes of system synthesis, elements and concepts of a framework can be classified in terms of structure (the configuration of relations and interactions that determine a system's behaviour), patterns (the abstract embodiment of a system's structure), and processes (the continuous evolution of a system's structure and patterns).
– *Theories.* Based on a particular methodological framework, theories enable system analysts to specify which elements and concepts of the framework are of particular relevance for certain questions, but also to identify certain assumptions about these elements and their possible interdependencies. Thus, within the boundaries of a particular framework, theories involve certain (strong or weak) assumptions that are applied when analysts inquire and diagnose a particular phenomenon to explain the involved processes and predict their outcomes. Typically, several theories are compatible with any given framework; in contrast, it can be difficult to determine true alternatives.
– *Models.* To develop and use models, explicit and precise assumptions about a limited set of parameters and variables are required. Logic, mathematics, game theory, experimentation, simulation, and other means are used to systematically explore the consequences of these assumptions for a limited set of outcomes. Most theories are compatible with multiple models. For example, several models may be compatible with a particular theory of system behaviour.
– *Practices.* The notion of (work) practices supports the analysis, instrumentation, and synthesis of system behaviour with respect to the theories and models at hand. Articulation of sustainable coordination presupposes, for instance, tools for observation and prediction of the existing system. However, practices that support inspection and prediction of coordination in multi-agent systems are by and large missing at the moment.
– *Systems.* Within a particular methodological framework, theories and models are developed using the practical tools and methods provided. All these theoretical aspects of a methodology aim to comply with scientific and engineering principles. To uphold such rigour, it is essential that there exist a

Fig. 3. *Sustainability.* A particular domain is constituted by the interplay of contexts, agents, and coordination in interplay. The mission of a domain defines the globally coherent invariance criteria to be sustained and regulated by the involved system and coordination mechanisms.

priori systems to actually analyse and instrument by means of observation and construction of particular phenomena.

2.2 Framework

The abstract perspective of coordination provided in the context of computer-supported cooperative work [33] is a good starting point for the focus on sustainable coordination of our methodological framework. As a scoping mechanism we introduce the concept of *domains*. The cognitive restriction of domains to some particular system allows us to introduce structures, patterns, and processes that are rich enough to address the issue of sustainable coordination in a particular context. We consider the main components of a domain to be a *context*, a dynamic set of *agents*, and a dynamic set of mechanisms for *coordination*. The overall purpose of a particular domain is captured by the concept of a *mission* (see Fig. 3).

We are interested in domains — i.e., applications — where the behaviour achieving the mission is supported by an information system; this means that some agents are human actors while the others are computational entities. The challenge then is to build and maintain an information system that supports the agents in achieving their missions in a particular domain. In short, we see coordination at the heart of open computational systems and, hence, as a key component of any methodology that aim at supporting the creation and maintenance of agents and services in a particular domain. The general issue of open computational systems is how to sustain coordination of a particular system with respect to its fixed domain and evolving context. Given a number of coordination mechanisms, the set of purposeful missions is constrained. The capabilities of the agents involved must meet the requirements of the coordination mechanisms and the current context in order to fulfil the mission criteria. We argue that an appropriate approach to resolve this issue is to equip all agents with the capabilities to continuously *articulate*, *construct*, *observe*, and *instrument* their contribution to fulfil some particular mission.

Articulation. Coordination in computer-supported cooperative work explicitly accounts for articulation of work, as the set of activities required to manage the distributed and coordinated nature of work [33]. Typically, articulation involves setting up the preconditions for carrying out a task or tiding up after completion of a task. As reported in [34],

"*... in cooperative work settings characterized by complex task interdependencies, the articulation of distributed activities requires specialized artifacts which, in the context of conventions and procedures, are instrumental in reducing the complexity of articulation work and in alleviating the need of ad hoc deliberation and negotiation ...*" (p. 163)

A specific type of articulation is due to change of context in a dynamic world. A typical cycle of articulation begins with a mission (see Fig. 3). Given the involved agents and domain of the mission, a suitable set of coordination patterns are determined (by means of negotiations and acknowledged by contracts that regulate the instantiations of coordination patterns). The articulation work continues by instantiating a suitable subset of coordination mechanisms together with observation criteria monitoring the execution of these mechanisms. The monitoring manages smooth changes of coordination, as regulated by the contracts, and discovers eventual breakdowns in a coordination mechanism. In the latter case, articulation work enters an exception state. Proper behaviour at exception states is crucial for user acceptance. Normally, a mission comes to and end (as agreed upon in the contracts) and the articulation work enters the closing and cleanup phase of the cycle. Appropriate articulation during the cycle depends on adequate observations. This is even more important if we also have to consider change of context during some particular mission. Such changes would typically trigger exception states, as discussed above. Furthermore, we might also need to add new types of observations during a mission. We therefore need to have possibilities to make instrumentations of a system in real time. We return to issues of observation and instrumentation after the following discussion of construction.

Construction. In Section 1.1 we introduced the distinction between subjective and objective approaches to coordination. In the context of computer-supported cooperative work there are recent efforts that aim to conceptually integrate these two strategies [8,20]:

"*... on one hand a strategy aiming at coordination technology reducing the complexity of coordinating cooperative activities by regulating the coordinative interactions, and on the other hand a strategy that aims at radically flexible means of interaction ... (leaving users) to cope with the complexity of coordinating their activities...*" ([35], p. 1)

In earlier approaches toward systems for computer-supported cooperative work the information system was primarily only regarded as a medium that enabled users to communicate and interact with each other. Today, such

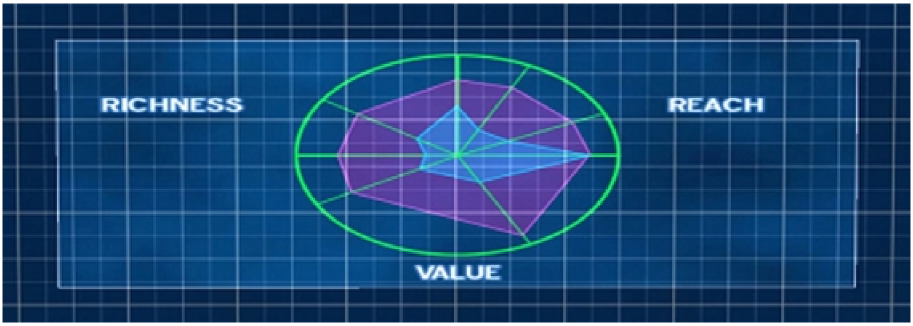

Fig. 4. *Assessment.* Analysis and assessment of a particular agent's context is constrained by certain dimensions (e.g., richness, reach, and value) that are characteristic of some specific domain under study (e.g., economics of information). The outer area illustrates sustainability criteria on a number of dimensions. Accepted technologies should always provide values for the different dimensions plotted in the inner area (delimited by the oval), thereby documenting they meet the given sustainability criteria.

systems also provide effective and malleable coordination services, in the form of basic mechanisms to be composed and exploited in order to support any collaborative activities of interest [35]. In particular, some approaches provide specific languages to build and dynamically coordinate artifacts and protocols[3] [8,20]. Given these computer-supported cooperative work perspectives, two main considerations arise for sustainable coordination of open computational systems. Firstly, information systems supporting open computational systems can most appropriately be exploited as the infrastructure, not only enabling user interaction, but possibly also providing coordination as a service [39]. Secondly, in the case of open computational systems the computer-supported cooperative work perspective must be enhanced toward the exploitation of the information system and related basic coordination mechanisms not only by (human) social actors, but also and mostly by (artificial) agents. Our methodological framework must, consequently, drive the selection of the relevant coordination mechanisms, according to the specific nature of the open computational systems.

Observation. The introduction of information systems and communication technologies has dramatically changed the perceived quality of information and its effective impact on system coordination. In literature, the concepts of information richness and reach are introduced in order to explain how information systems and the involved technologies change the way we perceive quality of information [12] and, consequently, its effect on sustainable coordination. Information richness is considered to be an aggregated measure of the quality of

[3] A similar approach can be found in some objective coordination infrastructures in the multi-agent systems context, such as TuCSoN [25], which provides coordination media called tuple centres whose behaviour can be programmed/forged dynamically by means of a special purpose logic-based language.

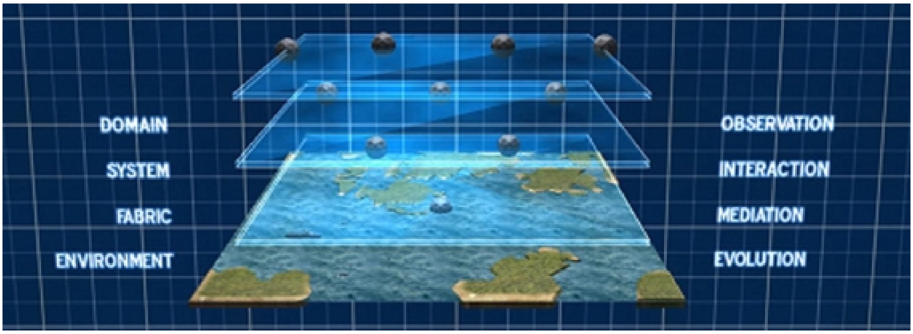

Fig. 5. *Open computational systems.* The sustainable and coordinated behaviour of open computational systems can be considered in terms of four distinct layers that exhibit certain key features: evolution, mediation, interaction, and observation.

information; information reach is the aggregate measure of the degree to which information is shared. In essence, what is called for is the possibility to quantify relevant system qualities and, by means of analysis, to instrument the systems accordingly. In order to compare technologies or to define sustainability criteria of coordination in open computational systems, we introduce so called *Kiveat* diagrams (see Fig. 4). Qualities are depicted as different radii in the diagram and some particular quality is considered to increase with the length of the radii. Such a diagram allows us to compare qualities of different technologies. In particular, we can also depict qualities that we believe contribute to systemic properties such as sustainability, i.e., the outer area in Fig. 4. The capability of achieving a mission in some particular domain by means of cooperation requires that we can achieve a sustainable level of such coordination criteria.

Instrumentation. Current approaches towards dynamical and complex information systems exhibits a mixture of domain- and network-centric focuses. Correspondingly, technologies such as service-oriented and peer-to-peer architectures emphasise new models of applications and connectivity. These architectures address the shortcomings of monolithic applications by introducing a model where dynamically assembled applications can be derived from some set of services, e.g., general purpose agents governed by coordination mediators. In order to create new and more powerful services, each service is therefore considered as available for reuse and recombination. A key concept in our methodology for sustainable coordination of open computational systems therefore is that of instrumentation. Instrumentation captures the dualistic relation between construction and observation of some particular system's behaviour.

3 Open Computational Systems

Contemporary research and development activities in complex multi-agent systems primarily focus on coordination of systems where interaction patterns and protocols are closed in nature, i.e., articulation is implicitly assumed to be accounted for by means of static coordination protocols or mediators. Practical application areas of such coordination approaches typically involve auctions, scheduling, and facilitation. However, in the case of multi-agent systems where coordination aims for an open and context-driven system behaviour, the presence of unpredictable events in the agents' environment introduces us to a quite different situation. Here, we find computational systems that most appropriately could be described as being of an evolving nature, i.e. the synthetic system behaviour cannot be fully known to any observer at any point in time. We characterise this class of multi-agent systems in terms of open computational systems. The behaviour of such systems is time dependent, due to the possible influence of external factors that cannot be predicted. If we aim at a sustainable system behaviour driven by frequent context switches, we therefore need to explicitly address the notion of continuous articulation, construction, observation, and instrumentation. In other words, we need to better understand the notion of sustainable coordination and in what way such a system quality can be analysed, instrumented, and synthesised in real-time. We argue that an increase in such knowledge could be helped by a separation of concerns, i.e., to introduce separate aspects of, as well as the support for, key features in coordinated system behaviour.

The material of this section is structured as follows. Firstly, we characterise the primitive aspects of open computational systems. In particular, we account for these aspects in terms of four layers of increasing abstraction and specialisation: *environment*, *fabric*, *system*, and *domain*. Secondly, challenges and opportunities in network-centric operations are summarised, by means of an open computational system called Twosome. This particular system accounts for the complex dynamics in information warfare environments. Finally, we return to the issue of continuous articulation, construction, observation, and instrumentation of coordination in real world application domains and identify *sustainability* as a key concept to be dealt with in any methodological approach toward open computational systems.

3.1 A Layered Perspective

An important property of complex multi-agent systems is that relations and interactions between entities change over time, due to unpredictable events in the surrounding environment. When a system exhibits such a property we often tend to denote it as an *open* system. However, such a characterisation tends to focus on a subjective use of the notion of openness. That is, a system may be considered as open with respect to one set of criteria but at the same time closed with respect to some other. Since an objective characterisation of openness would serve us better in identifying challenges and opportunities of sustainable coordination, we have devoted the following material to a meta-discourse of open computational systems. We introduce a layered perspective in order to account for an increasing level of abstraction in what we consider to be the primary causes and effects of some particular system's qualitative behaviour, i.e., the continuous evolution of structures, patterns, and processes. To that purpose, we will use the four primitive dimensions of environment, fabric, system, and domain, as well as their respective key features, i.e., *evolution*, *mediation*, *interaction*, and *observation* (see Fig. 5).

Environment. A key aspect of open computational systems is their characteristic property of being open and dynamic in nature. We consider such systems as situated in environments where new agents and services can enter and leave the environment at will. In this environment entities, computational or not, observe and interact with each other. This environment is also the origin of unpredictable events affecting sustainable behaviours and, hence, coordination of any system situated in this context:, agents and services cannot only enter and leave a system at will, they can also be forced to change state as a result from external stimuli that cannot be predicted at design time. Agents and services can also be forcefully removed from the environment as a result from unknown events. In a sense, an agent that proliferates in an open computational system is by all means subject to a continuous struggle for survival and evolution.

Fabric. The fabric of interaction is an integral part of an open computational system's environment. Any set of agents and coordination artifacts require some form of support for mediation. We consider the fabric of an open computational system to explicitly support the (local) existence of agents and services, as well as the dynamic coupling of a dynamic set of fabric nodes. Each fabric node must support the construction and observation of cognitive constructs as well as the interaction between agents. In summary, the fabric of an open computational system is considered in terms of the artifacts involved as well as their connectivity and required support for mediation, interaction, and observation. However, the fabric's support for mediation must never involve the actual coordination of agents and services. This feature is solely imposed on domain specific constructs at the system and domain levels of open computational systems.

System. An integral part of the interaction fabric, there exist a dynamic set of computational entities. As such, each entity, e.g., agent or service, at the system level exhibit an autonomous behaviour by means of observing the surrounding environment and interacts accordingly. In doing so, a computational entity requires the explicit support for mediation provided by the fabric of interaction. Due to the occurrence of frequent context switching in open computational systems, the entities that interact and form some particular behaviour also require the capability to dynamically create cognitive constructs — i.e., models — of themselves and their particular behaviour. These constructs are used by other entities in the continuous process of articulation, construction, observation, and instrumentation.

Domain. From a holistic perspective, the domain layer is one of the most important perspectives of an open computational system. By means of a dynamic set of cognitive constructs — i.e., constructed and instrumented by the computational entities at the system level — the activity of observation is primarily

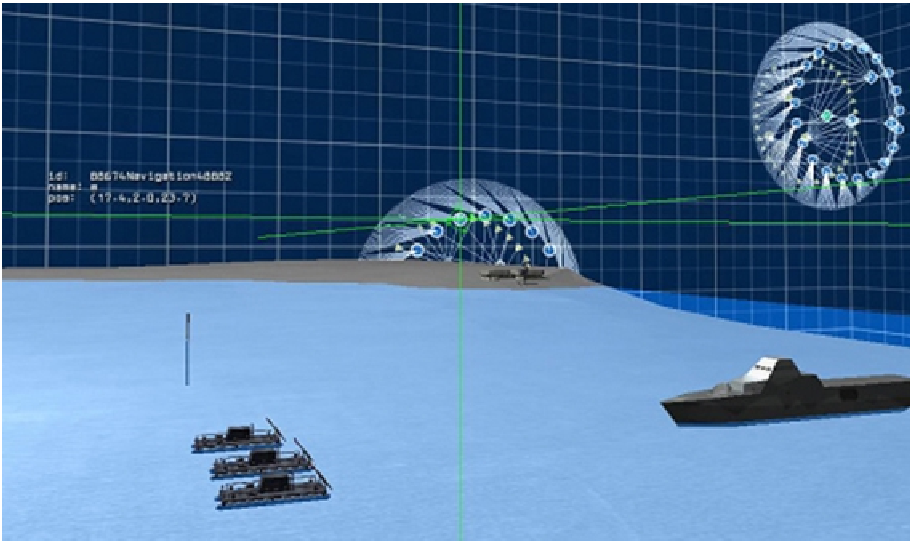

Fig. 6. TWOSOME. Two systems in interaction, i.e., attacker and defender, require sustainable coordination. In particular, due to unpredictable events in the (physical) environment, multiple coordination mechanisms must be provided to agents belonging to different cognitive domains but the same physical environment.

dependent on an agent's capability to discover potential entities to interact with in some articulated mission. It is by means of observation of these cognitive constructs that an agent implicitly is capable of measuring characteristic qualities in a system. In essence, the domain layer of open computational systems is supported by each node in the fabric of interaction, but articulated, constructed, observed, and instrumented by the dynamic set of entities residing at the system level.

3.2 Trustworthy and Sustainable Operations in Marine Environments

Research and development of information systems for defence and warfare have changed most dramatically during the last decade; from weapons of mass destruction to sustainable systems of coordinated and computationally empowered entities, i.e. network-centric warfare. From a holistic perspective, the involved systems are comprised by a wide range of agents and services: sensor and actuator systems, detection and weaponry systems, as well as communication, decision, and support systems. To that end, the behaviour of each system component, as well as their synthetic behaviour, has to be dependable and trustworthy in operations under dynamic and hostile conditions, and — perhaps even more challenging — the notion of system lifespan has to be considered in terms of decades. Consequently, at the core of a prototype system for *Trustworthy and* *sustainable operations in marine environments* (TWOSOME)[4] was the development of a multi-agent system, where interacting and coordinated entities and services temporarily come together in a physical setting in order to perform a particular assignment under dynamic and hostile conditions. The system developed is subject to a validation of qualities such as information fusion, adaptation, shared awareness and decision making, i.e., coordination in military missions and operations.

TWOSOME involves two interrelated application domains — the viewpoints of attackers and defenders (see Fig. 6) — and missions, i.e., creating and removing physical threats at some particular environment location. The various agents involved are positioned in a physical environment and therefore, by means of their autonomous and cooperative behaviour, create an evolving state of affairs that is impossible to handle by means of a single coordination mechanism. An attacker — a smart mine — is positioned in the environment and set to detonate whenever the presence of a particular vessel is identified in the surroundings. Correspondingly, three coordinated defenders — autonomous vessels emanating acoustic and magnetic signatures — are assigned the role of sweeping different environment locations and aim at removing potential attackers by means of making mines detonate in a harmless way. The role of a team of defenders is to outsmart the mine by means of providing fake signatures of real vessels. In TWOSOME, the issue of sustainable coordination is explicitly accounted for by the three defenders by means of continuously executing the articulation, construction, observation, and instrumentation feedback loop and thereby maintaining a particular signature at hand. In reality, this involves the continuous adaptation and reconfiguration of roles. In accordance with the layered perspective of open computational systems, the following material introduces an outline of TWOSOME's characteristic environment, fabric, and system properties that account for the concept of *deception in network-centric warfare*.

Environment. The physical bounding space of the scenario has an imaginary location and topology, but the physical dimensions of the environment are set to 1 * 1 * 1 nautical miles. Within this delimited volume a number of constructs form the physical and marine environment where the scenario takes place (see Fig. 6):

- *Mainland.* The scenario environment involves mainland at two different locations. The two landmasses form a natural harbour in the physical bounding space, in which naval vessels can start and end arbitrary operations. In particular, an operations centre is localised at one of the landmasses. It gathers intelligence and acts upon its currently available information history.
- *Island.* Localised between the two landmasses, the environment includes an island that brings about two separate navigation channels. The depths of the channels are defined by the sea bed topology, in relation to the sea surface. The channels offer the (possibly) safe passages for different marine vessels.

[4] For more information concerning the TWOSOME project, please visit the following URL: `http://www.soclab.bth.se/systems/twosome.html`.

Fabric. As previously described, the fabric of an open computational system is considered in terms of the physical artifacts involved as well as their connectivity and computational capabilities. In TWOSOME's fabric of interaction the following types of artifacts are involved: *operations centre*, *transporter*, *attacker*, *defender*, and *sensor*.

- *Operations centre.* The operations centre can gather intelligence and, based on newly acquired information, decide that creating or removing a threat — e.g., a smart mine — at a given environment location would be of strategic importance. Consequently, an operations centre can create and delegate tasks to other autonomous physical entities and then continue with the coordination of other operations.
- *Transporter.* To carry out multiple tasks in parallel, they have to be assigned to entities capable of taking on the required roles. When such a capability involves transportation, the opportunity arises to create and delegate tasks to be carried out at remote locations as well as simultaneously assigning additional tasks to a designated transporter. Mission efficiency in general is positively affected if multi-purpose vessels capable of transportation are involved and can be trusted to effectively carry out multiple tasks by means of delegation to other autonomous physical entities with communication and computation capabilities. In the present scenario, multi-purpose vessels are operationalised by means of a corvette vessel capable of carrying out a defence scenario, i.e., to transport a group of defenders from one location to another and subsequently delegate a defence order to a designated transported entities.
- *Attacker.* In the scenario, the role of an attacker is operationalised by means of a mine with the sole purpose of detonating when certain acoustic and magnetic signatures are identified in its surrounding environment. A mine is equipped with sensors that recognise the occurrence of particular vessel types. By means of combining currently available sensor information, a mine can make informed decisions whether or not to detonate, given its current environment state history and identified vessel signatures.
- *Defender.* This artifact is a small autonomous vessel that emanates different acoustic and magnetic signatures in order to exhibit characteristics similar to those of general naval vessel properties, e.g., propeller cavitations and engine acoustics. In doing so, the defence vessel has the ability to trigger the detonation mechanisms of artifacts such as mines. This autonomous capability of the vessels can, by means of communication and coordination technology, be combined into complex signatures. Multiple vessels can achieve higher mission efficiency than stand-alone vessels, given that they are able to autonomously reconfigure themselves according to changing environment conditions and contexts.
- *Sensor.* One of the most important artifact classes in open systems, i.e., when it comes to the capability of agents to act in an autonomous and sustainable manner, is that of sensors. The sole purpose of sensor entities is to acquire information concerning their surrounding environment and pass it forward in an appropriate format to those entities that are dependent on such information in order to fulfil their particular mission.

System. As indicated previously, the fabric of an open computational system is what constitutes the interaction and coordination medium. From the layered perspective of open computational systems, the next level of abstraction involves the actual system in question, i.e., where agent behaviour and coordination mechanisms are located and integrated with each other. In the particular case of TWOSOME, an attacker (smart mine) is created somewhere in the physical environment and thereby triggers observations by three different sensors. This information is then passed on to an operations centre that sends out a transporter carrying a number of defenders, i.e., autonomous vessels emanating deceiving acoustic and magnetic signatures. At some particular location, estimated by the operations centre based on the information given, the transporter deploys the defenders and travels back to its origin. It is at this point that we can start to understand the opportunities and challenges with sustainable coordination.

Each defender has the capability to interact, navigate, and deceive. Sets of defenders can produce a combined signature that is difficult to distinguish from that produced by real vessels threatened by an attacker. The coordinated structure of a number of defenders is therefore very sensitive to perturbations in the surrounding environment. Since the quality of the coordinated team of defenders corresponds to their effectiveness in deceiving some potential attacker — i.e., making it detonate and thereby removing the threat — it is essential that each defender produces an appropriate complex signature at the appropriate location. In order to do so, each agent must be able to observe and instrument its current context with respect to the given mission and the available set of coordination mechanisms. For example, if a particular defender's capability of producing some specific signature suddenly is removed, the whole team needs to reconfigure and reuse its currently available services. This state of affairs requires that the whole process of coordination — i.e., articulation, construction, observation, and instrumentation — has to be performed all over again. Obviously, since a switch in context has occurred, going through the coordination process again implies that the system's overall quality has changed. The question is how to sustain the coordination of this quality.

3.3 The Problem Revisited

Coming back to the main issue addressed in this paper, it stands clear that both subjective as well as objective approaches to multi-agent systems coordination implicitly assume that the articulation of some particular mission is an integral component of the system in question and, consequently, must not change during its life time. However, as accounted for by TWOSOME, there are indeed situations — i.e., context switches — that require the global coherence of some particular system's mission and quality as a whole to be continuously observed and instrumented. Consequently, when some complex multi-agent system is situated in an evolving physical environment, it is of fundamental importance that the role of articulation is made explicit. This also implies that multiple coordination mechanisms are required as a matter of the context switching taking place in the particular system under study. In summary, if we consider sustainable coordination to be a matter of continuous articulation, construction, observation, and

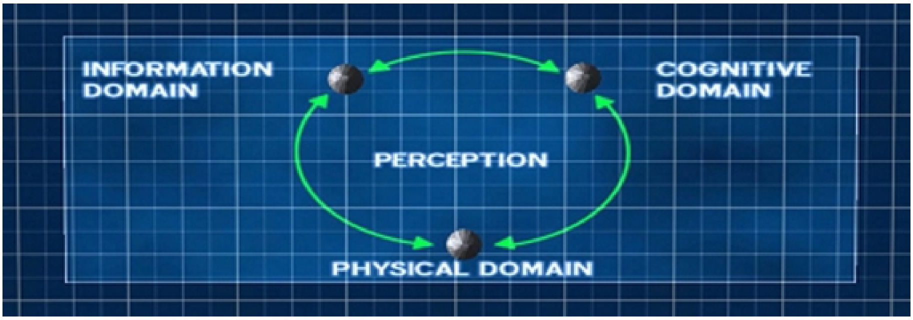

Fig. 7. *Coordination.* By means of perception, a particular agent must be able to observe its physical environment in order to acquire the qualitative information that characterises its current context. Such information can then be used by the agent, by means of its cognitive domain, in analysis and instrumentation of the physical environment.

instrumentation of open computational systems, we need to address this type of continuous system adaptation in terms of a comprehensive and methodological approach.

4 Sustainable Coordination in Practice

A comprehensive methodology should, necessarily, not only provide us with a suitable framework and corresponding theories and models, but also guide us in development and applicability of methods and tools, i.e., when undertaking a full attempt to resolve domain specific issues. Since coordination of system behaviour is one of the primary dimensions of constructing multi-agent systems [6, 45], the previous considerations have a deep impact on any methodology involved and, in particular, on the practical tools we make use of in applying models of sustainable coordination. Indeed, the impact of a systemic perspective on the required methodologies and tools currently conceived for multi-agent systems construction is very strong [13,46,29] and, when it comes to articulation of sustainable coordination (see Fig. 7), calls for the explicit account of *construction*, *observation*, and *instrumentation*:

– *Construction.* In terms of certain (articulated) qualitative dimensions, a dynamic set of agents must be able to interact with each other and fulfil a given mission. In order to do so, the particular mission involved must necessarily be possible to configure, by means of articulation. Given an initial articulation, the involved set of agents requires multiple mechanisms for coordination and survival in a continuously evolving environment.
– *Observation.* Given a particular mission and articulation, each agent requires the capability to observe and analyse its current context. This requirement is imposed on a cognitive entity by means of the capability to turn domain specific qualities into quantities, i.e., observing and then evaluating dynamically acquired measurements from the environment.

– *Instrumentation.* When an agent has the ability to observe and analyse its current context, it is fully equipped not only to interact with other agents in a meaningful way, but also to take part in a jointly articulated and coordinated mission in dynamic environments. By means of articulation, construction, and observation an agent can successfully instrument its own context and cognitive domain as a matter of interacting with other agents or the coordination mechanisms involved.

In summary, we consider the continuous feedback loop of articulation, construction, observation, and instrumentation to be the cornerstone of sustainable coordination of behaviour in open computational systems. Related ideas and frameworks have been proposed in the theoretical areas of interaction [40] and empirical computer science [41,42], and are suggested to become the fundamental scientific and practical approach in understanding, as well as engineering, the structures, patterns, and processes involved in open computational systems. In the case of open computational systems and sustainable coordination, we have implemented two methodological tools to support the continuous articulation, construction, observation, and instrumentation of system behaviour: SOLACE and DISCERN.

This section is structured as follows. Firstly, we outline the required practical support in construction of open computational systems. In particular, we have developed a distributed architecture called SOLACE that accounts for the construction of interaction fabric, system behaviour, and cognitive domains (see Fig. 5). Secondly, our current efforts in development of support for observation and instrumentation are summarised in terms of a distributed architecture called DISCERN. This particular architecture addresses sustainable coordination in open computational systems in terms of real time quantification of domain specific qualities as well as instrumentation of individual agents and their behaviour. However, it should be noted that sustainable coordination of a particular mission, i.e., the continuous process of articulation, construction, observation, and instrumentation, necessarily has to be accounted for by the involved system itself; the practical methods and tools presented only address the support of the involved structures, patterns, and processes.

4.1 A Service-Oriented Layered Architecture for Communicating Entities

The most characteristic property of open computational systems can be described as a natural support for construction and observation. In order to explicitly account for the issues of construction and observation of system behaviour, we introduce a formal model for construction that is grounded in process algebra, with the operators of join (integration) and connect (interaction). This approach enables us to observe the fundamental constructs involved in sustainable coordination and, consequently, in what way instrumentation of system behaviour most appropriately can be supported in terms of continuous articulation, construction, and observation. As such, construction is considered to be a matter of integration and interaction, i.e., the ability of an agent to enter or leave a

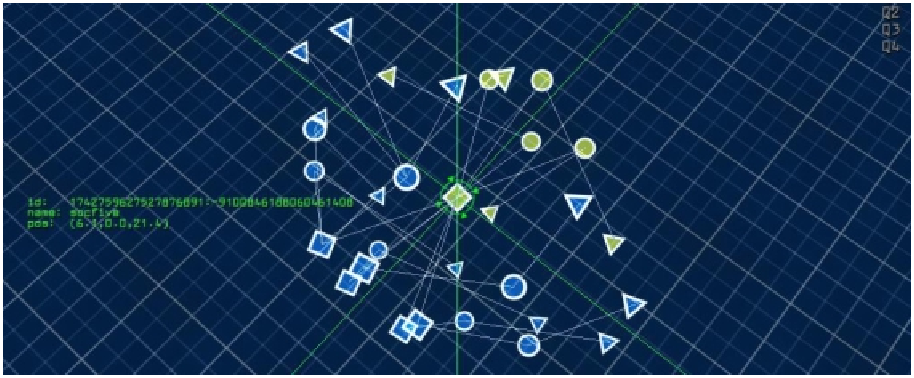

Fig. 8. *Construction.* In SOLACE, empirical studies of open computational systems are supported by means of (multiple) qualitative perspectives, i.e., online models of structures in open computational systems. We can therefore add and join structures in an algebraic setting, as described in Section 4.1

particular location in some system and the ability of agents in the system to interact with each other, by means of integration.

Support for Construction. In order to support the observation of different construction aspects, we model the distinct agent activities of *integration* and *interaction* by means of process algebra techniques. Our theory of conjunction is based on Milner's calculus of communicating systems [22]. The basic notions and properties of such a systems calculus are *processes*, *ports*, and *operators*:

1. We have a numerable set of processes, $\{a,\ b,\ c,\ \ldots\}$, with discrete internal states and a finite, yet dynamic, set of external ports. A process with an empty set of external ports is considered to be of a closed nature. In any other situation, the process is considered to be open, by virtue of possible interactions with its surrounding environment.
2. Each port is typed as either export or import, with a signature identifying the type of stimuli that can be sent or received.
3. There are two operators in the process algebra, $+$ (add) and $|$ (join). The join of two processes a and b is possible if and only if there is an import-export pair of ports, one belonging to a and the other to b, with matching signatures.
4. In the particular situation where two processes are joined, a corresponding import-export pair of ports becomes an internal (invisible) port. Message passing will occur if and only if the send-receive actions are enabled by the corresponding internal states.
5. The operators of add and join allow us to construct compound algebraic expressions that denotes new processes: $d = (a + b \mid c) \mid (b + c)$. Furthermore, there is a set of algebraic laws that allow us to expand and compare algebraic expressions. Examples of such laws include the distributive law: $a \mid (b + c) = a \mid b + a \mid c$.

6. The expressions $a + b$ and $a \mid b$ are considered to have the following properties: $Ports(a + b) = Ports(a) \cup Ports(b)$ and $Ports(a \mid b) = Ports(a) \cup Ports(b) - \{connected\ ports\}$.
7. By expanding expressions we can derive the behaviour of complex processes in terms of stimuli exchange between primitive processes.

In our setting of open computational systems, the processes of communicating systems correspond to agents, or services, and the concept of ports determines type and specification of a particular service. The operators of add and join correspond to, respectively, integration and interaction. An agent or service s where $Ports(s) = \emptyset$ should therefore be considered as closed and, consequently, not observable or attainable from its environment. In effect, we consider the behaviour of any primitive or complex service as amenable to analysis by means of its continuous interactions and integrations with the surrounding environment. However, the appropriate representation and explanation of system behaviour in this context is of a very challenging nature. Even though an initial identification of the agents and services involved in some coordinated behaviour is momentarily attainable, it should be noted that subsequent interaction and integration of the system continuously originate from unknown sources. In a real setting, the add and join operators of processes can only be effectuated when contextual constraints are fulfilled. This means that the processes or agents and services have descriptions attached to be assessed which determine whether or not an operator is allowed. We thus have:

$$desc(a) = functional\_desc(a) \cup quality\_desc(a)$$

where $functional\_desc(a)$ denotes a description of the functionality of the service a and $quality\_desc(a)$ is a description of relevant quality attributes of the service. When it comes to the join operator, a crucial issue of service conjunction is the preservation of essential qualities, e.g., to fulfil sustainability criteria. The add operator allow us to choose between different services in order to fulfil the crucial criteria of a join operation. Returning to the setting of TWOSOME, we have three defenders: a, b, and c, that can generate different signatures. Together, they can generate a complex signature of a particular vessel and thus trigger a smart mine to detonate without harming the threatened vessel. However, during the operation some of the defenders can lose some of their signature capabilities, so that the defenders no longer can outsmart the mine. The defence team d can be expressed by the following algebraic expression:

$$d = (a + b + c) \mid (a + b + c) \mid (a + b + c) \tag{1}$$

Initially we can expand d as

$$d = a\,b\,c \tag{2}$$

Indicating that the ordered sequence of $a\,b\,c$ can simulate the signature of the intended vessel. Let us assume that a emanates the signature of a vessel engine. Let us also assume that after a while the defence discovers that a

not longer can provide the desired signature. The defence thus has to recover the intended meaning of d by making another expansion of the expression (1). Let us assume that the following expansion of d now fulfils the requirements:

(3) $$d = c\,a\,b$$

The defenders a, b, and c can now regroup as in (3) by coordinated actions. In summary, algebraic expressions, corresponding to agents and services, can aid us in analysis and reasoning about some particular real time system quality (see Fig. 8), e.g., sustainable system coordination as well as during maintenance and repair.

Architecture for Construction. We have developed a distributed *Service-oriented layered architecture for communicating entities* (SOLACE), that supports the construction of open computational systems. This architecture explicitly supports the three abstraction layers of fabric, system, and domains (see Section 3.1 *Layered perspective*). Support of the environment perspective has been omitted in SOLACE because the architecture as such is only supposed to support the abstract perspectives characterising constructs that are an integral part of the physical environment. That is, we consider the architecture to be an integral part of the physical environment and not vice versa. Our architecture therefore supports the layered perspective of open computational systems in terms of *fabric*, *system*, and *domain*.

– *Fabric.* In terms of current implementations of service-oriented architectures, the fabric layer of open computational systems must explicitly address scalability in its support for mediation. In doing so, SOLACE provides for the automatic and localised coupling and decoupling of fabric nodes. This is done by means of multiple classes of broadcast protocols that, at regular intervals, advertise a particular node's existence in the surrounding environment. Nodes already existing in its surrounding environment receive these broadcasts and acknowledge the presence of the signalling node. SOLACE thereby provides system entities with support for mediation, interaction, and observation, irrespective of their physical locality.
– *System.* A number of computational entities exist as an integral part of each node in the fabric of open computational systems that, by means of mediated interaction, form complex and dynamic system behaviour. By means of the fabric, SOLACE supports each entity — i.e., agent or service — in the continuous feedback loop of articulation, construction, observation, and instrumentation. As soon as an entity is launched and temporarily resides on a particular fabric node, it is automatically coupled with and accessible by its neighbouring system entities. The architecture therefore supports all entities with the essential mechanisms for discovery, addressing, and routing. These mechanisms provide all agents with the support needed in sustainable coordination of an open computational system.
– *Domain.* Current implementations of service-oriented architectures more or less lack the support for multiple domain perspectives of a particular system, i.e., separation of concerns. Our methodological approach toward open

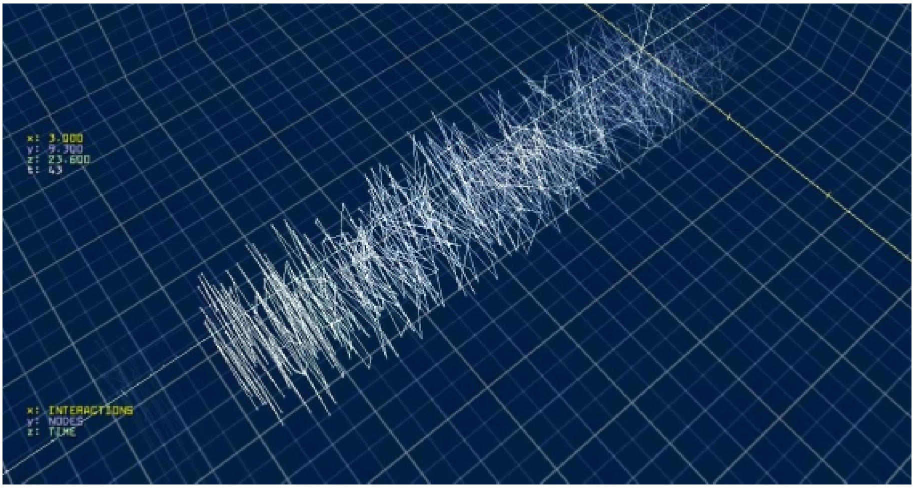

Fig. 9. *Observation.* In DISCERN, empirical studies of open computational systems are supported by means of (multiple) quantitative perspectives, i.e., online models of patterns in open computational systems. A specific observation must necessarily encompass important quality dimensions as well as their quantification.

computational systems introduces the separate perspectives and features of environment, fabric, system, and domain. Supported by the fabric, a dynamic set of cognitive constructs — i.e., constructed and instrumented by the computational entities at the system level — can be observed by means of the mediation support offered by SOLACE. The cognitive constructs supported correspond to some particular agent's domain adherence and characteristics. It is by means of observing a particular agent's cognitive constructs that other agents in the fabric are implicitly capable of measuring characteristic qualities in the system. To that end, SOLACE supports the domain layer as a matter of construction, observation, and instrumentation of multiple cognitive constructs. In particular, the architecture supports this continuous process in terms of implementing our theory of conjunction.

4.2 A Distributed Interaction System for Complex Entity Relation Networks

The concepts and models of multi-agent systems explicitly address systems for which interaction between autonomous agents is a characteristic property. It is therefore important that we understand in what way these models are applied in the actual construction of multi-agent systems, and, equally important, that we understand in what way a multi-agent system can be observed in terms of the same models. An understanding of this strong relationship between the models used in construction and observation is pivotal since it is the fundamental basis for an informed activity of instrumentation. When methodological approaches towards construction and observation originate from implicit assumptions we consider the applicability and common benefit of those models to be negatively affected.

Support for Observation and Instrumentation. Observation of behaviour in complex systems is an important methodological approach to multi-agent systems. As we can only understand what we can observe and measure, we focus on modelling of the autonomous behaviour that gives rise to system behaviour. Dimensions of a complex and dynamical system are identified and assessed with respect to the dimensions of a particular model. An important assumption of this approach is therefore that certain relevant dimensions are accessible from a system that exists a priori. These particular dimensions alone are considered to reflect the relevant behaviour in a system. Examples of similar approaches can be found in control theory, where system identification and analysis are used as a tool in observation and explanation of behaviour in complex systems. We argue, however, that the basic assumptions of accessibility and support concerning a fixed set of dimensions do not hold if a system is considered to be of an open nature. That is, if agents can enter or leave a system's proximity at will, the idea of certain conceptual dimensions alone reflecting the relevant behaviour of a system is no longer true. In general, the reason for this can be ascribed to the problem of representing the dynamical nature of integrated systems [7]. If at any given point in time we cannot be certain of a system's structure, interaction patterns, or evolutionary processes, then how can we ever be certain that we have captured its relevant dimensions in terms of an *a priori* developed model? In particular, how can we assure that a particular coordination protocol or mediator does not actually hamper us in sustaining some dynamic quality of interest? After all, sustainable coordination must never involve implicit and static constraints on system adaptation. Here we refer to Fig. 2 and the discussion on articulation in dynamic contexts related to that figure. There, we noted that the quality and adequacy of observation and instrumentation has to be complemented by means of support for articulation and construction. However, since we are dealing with a continuous feedback loop of articulation, construction observation, and instrumentation, an appropriate question at this point is to consider in what (practical) way we can support observation and instrumentation.

Architecture for Observation and Instrumentation. In practice, the activity of observation aims at acquiring certain quantifications of some domain's characteristic qualities. Acquiring these measurements can be done by means of certain instruments. Also, we consider these quantifications to be the basis in an actual instrumentation of some particular system under study. We therefore have implemented a *Distributed interaction system for complex entity relation networks* (DISCERN) that explicitly addresses the dynamic and real time observation and instrumentation of open computational systems residing on SOLACE, e.g., TWOSOME. DISCERN enables the observation of *qualities* and *quantities* in multiple domains of open computational systems.

– *Qualities.* As described in Section 3.1, we consider a layered perspective of qualities in open computational systems (see Fig. 5), in terms of environment, fabric, system, and domain. Discern is therefore applied in the observation of domain specific qualities with respect to these four perspectives. However, even though Solace does not address the notion of a physical environment, this is of utmost importance in Discern in order to immediately provide a human agent and observer with the current context. In a similar manner, the fabric, system, and domain layers of open computational systems are represented in Discern by means of a volumetric space that is navigable.
– *Quantities.* In addition to the possibility of a human agent to observe and instrument the qualities of open computational systems, Discern also provides support of quantification. That is, by means of the constructs accounted for by Solace, any quality available in some accessible open computational system can dynamically be quantified and further analysed in Discern (see Fig. 9). Examples of the sustainable coordination of such qualities and quantities are currently studied in Twosome, in terms of information fusion, adaptation, awareness, and policies. Since Discern supports real time observation and instrumentation of open computational systems, the quantities acquired in some particular context can be used in further studies on automated analysis and in the continuous feedback loop of articulation, construction, observation, and instrumentation.

5 Summary and Concluding Remarks

In Section 1 we highlighted current important trends in distributed computing, i.e., ambient intelligent systems or autonomic computing. We also stated that a multi-agent system approach provides a suitable high-level (society level) abstraction of such systems. Of specific importance is that this approach explicitly positions coordination as its focus. Proponents of ambient intelligent systems and autonomic computing (as well as proponents of web services) stress that most applications (or services) in future complex systems will rely on suitable combinations and adaptations of existing components. Hence, we propose that application development — i.e., based on coordination patterns — will be predominant in methodologies for future distributed systems. Given the central role of coordination in complex distributed systems, we firstly argued that coordination related to articulation and instrumentation is an essential mechanism in support for flexible and dynamic behaviour. Secondly, we introduced the systemic concept of sustainable coordination to account for control of emergent behaviours or global coherence in complex systems.

We have illustrated our ideas and our methodology by means of an example from network-centric operations, Twosome, that has been constructed, observed, and instrumented with the distributed architectures of Solace and Discern. We introduced Kiveat diagrams as a tool to express important concepts and their relations. The diagram in Fig. 11 illustrates the information concepts of information richness versus reach as well as customer value (cf. Fig. 4). The quality dimensions in the figure are of particular relevance in the application domain of Twosome [1]. We have also indicated the minimal values of the qualities that have to be maintained during the whole lifetime and co-evolution of the mission capability package related to the Twosome scenario. That is, we have defined sustainability criteria as a boundary curve in the corresponding Kiveat diagram. Obviously we have to address two important issues. Firstly, we cannot have qualities in the diagram if we cannot measure or compare them to each other across different contexts (i.e., situations). Secondly, we need tools for observation and instrumentation of a particular system's characteristic qualities during the lifetime of its mission.

Concerning measurability of qualities in a Kiveat diagram (as in Fig. 10), the intended reading is that the quality increases (improves) moving outwards along its radius, starting from the origin. The measures could be different for different qualities. E.g., the 'sensing' quality of Fig. 10 could be measured in bits/sec, i.e., a numeric value. On the other hand, while a quality such as 'shared awareness' could be measured in terms of the number of people having the same screen view at any given time, another — and more interesting — measure could be the coherence in a group assessing a given mission situation. In the latter situation, we do not immediately obtain a numeric value, but given proper training and education a team arguably could evaluate the strengths of different technologies with respect to that quality. As a matter of fact there is an (unproven) statement in military circles that systems supporting network-centric operations have higher values on all qualities in Fig. 10 than systems supporting platform-centric operations. In short: Each quality in a Kiveat diagram has a non-empty set of measures associated to it, where the value of each measurement is in a ordered set. The measurements themselves could be provided by instrumentations in the system (Fig. 11) or by observations by users of the systems and/or evaluations by teams of users. The individual (or group) value of the information system at hand (e.g., Twosome) is given by a function of the qualities in the value sector of the Kiveat diagram. In the case given in Fig. 10 we thus have

$$Value_{user} = f_{context}\ (knowledge,\ awareness,\ understanding,\ decisions)$$

with values given by the situation at hand. The challenge then is to determine a suitable function $f_{context}$. When such a function exists, we can compare its value across different information systems and come up with a methodology to increase (or optimise) the value, while maintaining the overall sustainability criteria. In the Twosome scenario there is a specific interpretation of the shared awareness criteria. That is, the defending team should have a shared awareness that the attacker is supposed to be outsmarted at all times. To maintain this criterion the defending team must be capable of observing qualities in order to detect malfunctions of the team's available resources (see Section 3.2 and Section 4.1).

In Section 1.1 we introduced our methodological framework (Vocs), focusing on the feedback loop of articulation, construction, observation, and instrumentation. We consider sustainable coordination as dependent on modelling and

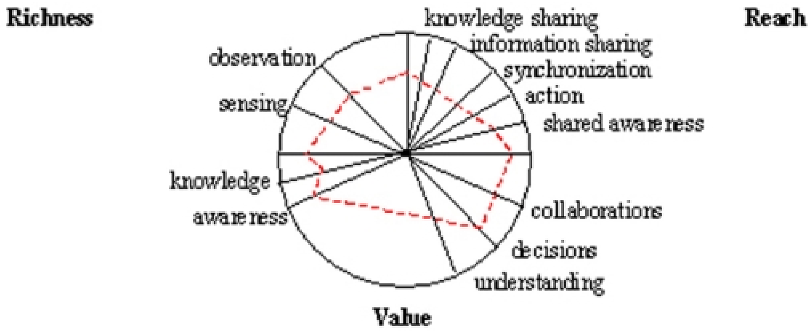

Fig. 10. *Sustainability.* A Kiveat diagram illustrating key quality attributes related to information richness and reach as well as agent value. The figure also illustrates minimal values of the quality attributes to uphold sustainability criteria during the life time of an open computational system.

support of dynamics in three interrelated constructs of (synthetic) open computational systems, namely structures, patterns, and processes (see Section 1.2). The roles of our distributed architectures, i.e., SOLACE and DISCERN, is captured by the Kiveat diagrams depicted in Fig. 10 and Fig. 11.

In the paper we introduce four distinct layers; environment, fabric, system, and domain, of open computational systems to support key system features, e.g., evolution, mediation, interaction, and observation (Fig. 5). DISCERN allows us to inspect the system from different points of view, but it also provides the capability of real time instrumentation of open computational systems (see Fig. 8 and Fig. 9).

To conclude, we have illustrated a novel approach towards design, implementation, and maintenance of complex distributed multi-agent systems. We have furthermore demonstrated our methodological approach applied to a complex real world application, TWOSOME. Sustainable coordination is at the core of our methodology and we have indicated how we can observe and maintain sustainability criteria by means of DISCERN. Obviously, much work remains to be done in terms of verifying and extending our ideas. Firstly, we will do so by assessing TWOSOME and other upcoming applications. Secondly, we will enhance SOLACE by introducing mission packages comprised of multiple coordination services, e.g., coordination patterns related to computational markets and resource management and objective coordination patterns such as TuCSoN. Thirdly, we will introduce complex computational entities that are capable of context-sensitive domain reasoning, e.g., about system stability and maintenance.

Acknowledgements. The authors would like to express their gratitude towards the members of the *Societies of Computation* (SOC) research group in Ronneby/Sweden as well as the *Dipartimento di Elettronica, Informatica e Sistemistica* (DEIS) research group in Cesena/Italy for their valuable comments on drafts of this chapter. Furthermore, we would like to thank all the personnel at

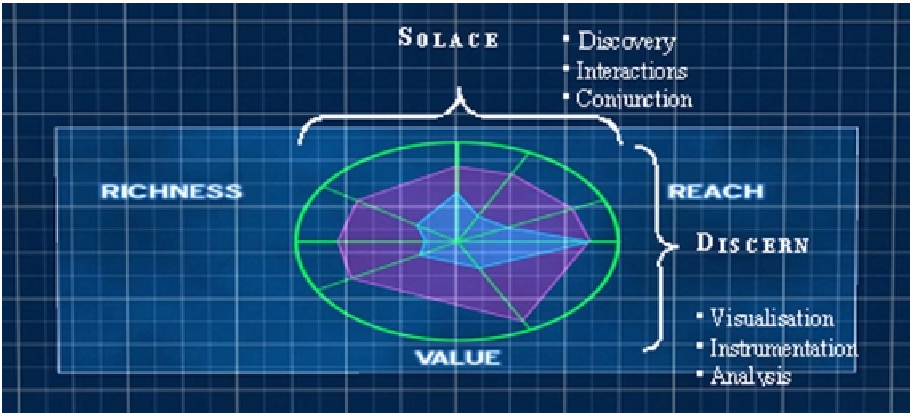

Fig. 11. *Architectures.* A Kiveat diagram illustrating the main characteristics and support provided by SOLACE and DISCERN.

the *Societies of computation laboratory* (SOCLAB) in Ronneby/Sweden, for their extensive work on developing the practical tools (SOLACE and DISCERN) and system (TWOSOME) outlined herein.

References

1. D. Alberts, J. Garstka, R. Hayes, and D. Signori. *Understanding Information Age Warfare*. CCRP Publication Series, 2001.
2. J. Bardram. Designing for the dynamics of cooperative work activities. In *Proceedings of the ACM 1998 conference on Computer Supported Cooperative Work*, pages 179–188. ACM Press, 1998.
3. F. Bergenti and A. Ricci. Three approaches in coordination of MAS. In *Proceedings of the 2002 ACM Symposium on Applied Computing (SAC'02)*. ACM Press, 2002. Track on Coordination Models, Languages and Applications.
4. S. Bussman. Self-organising manifacturing control: An industrial application of agent-technology. In E. Andre and S. Sen, editors, *Proceedings pf the 4th IEEE International Conference on Multiagent Systems*, pages 87–94, Boston (MA), July 2000. IEEE Press.
5. G. Cabri, L. Leonardi, and F. Zambonelli. Context-dependency in internet-agent coordination. In A. Omicini, R. Tolksdorf, and F. Zambonelli, editors, *Engineering Societies in the Agents World*, volume 1972 of *LNAI*, pages 51–63. Springer-Verlag, Dec. 2000.
6. P. Ciancarini, A. Omicini, and F. Zambonelli. Multiagent system engineering: the coordination viewpoint. In N. R. Jennings and Y. Lespérance, editors, *Intelligent Agents VI — Agent Theories, Architectures, and Languages*, volume 1767 of *LNAI*, pages 250–259. Springer-Verlag, Feb. 2000.
7. P. Cilliers. Problem with representation. In P. Cilliers, editor, *Complexity and Postmodernism: Understanding Complex Systems*, pages 58–88. Routledge, 1998.
8. M. Cortes. A coordination language for building collaborative applications. *International Journal of Computer Supported Cooperative Work (CSCW)*, 9(1):5–31, 2000.

9. S. R. Cost, Y. Labrou, and T. Finin. Coordinating agents using agent communication languages conversations. In Omicini et al. [26], chapter 7, pages 183–196.
10. U. Dayal, M. Hsu, and L. Rivka. Business process coordination: State of the art, trends and open issues. In M. G. Apers, P. Atzeni, S. Ceri, S. Paraboschi, K. Ramamohanarao, and R. T. Snodgrass, editors, *Proceedings of the 27th VLDB Conference*, pages 3–13. Morgan Kaufmann, Sept. 2001.
11. E. H. Durfee. Scaling up agent coordination strategies. *IEEE Computer*, 34(7):39–46, July 2001.
12. P. Evans and T. Wurstel. Strategy and the new economics of information. *Harward Business Review*, pages 71–82, Sept. 1997.
13. M. Fredriksson and R. Gustavsson. A methodological perspective on engineering of agent societies. In A. Omicini, P. Petta, and R. Tolksdorf, editors, *Engineering Societies in the Agents World II*, volume 2203 of *LNAI*, pages 10–24. Springer-Verlag, Dec. 2001. 2nd International Workshop (ESAW'01), Prague, Czech Republic, 7 July 2001, Revised Papers.
14. M. Fredriksson and R. Gustavsson. Theory and practice of behavior in open computational systems. In R. Trappl, editor, *Cybernetics and Systems 2002*, Vienna, Austria, 2002. Austrian Society for Cybernetic Studies. 16th European Meeting on Cybernetics and System Research (EMCSR 2002), 2–5 Apr. 2002, Vienna, Austria, Proceedings.
15. R. Gustavsson and M. Fredriksson. Coordination and control in computational ecosystems: A vision of the future. In Omicini et al. [26], chapter 18, pages 443–469.
16. B. A. Huberman and T. Hogg. The emergence of computational ecologies. In L. Nadel and D. Stein, editors, *Lectures in Complex Systems*, pages 185–205. Addison-Wesley, 1993.
17. N. Jennings. Coordination techniques for distributed artificial intelligence. In G. M. P. O'Hare and N. R. Jennings, editors, *Foundations of Distributed Artificial Intelligence*, pages 187–210. Wiley, 1996.
18. M. Klusch and K. Sycara. Brokering and matchmaking for coordination of agent societies: A survey. In Omicini et al. [26], chapter 8, pages 197–224.
19. Y. Labrou, T. Finin, and Y. Peng. Agent communication languages: The current landscape. *IEEE Intelligent Systems*, 14(2):45–52, March/April 1999.
20. D. Li and R. Muntz. Coca: collaborative objects coordination architecture. In *Proceedings of the ACM 1998 conference on Computer Supported Cooperative Work*, pages 179–188. ACM Press, 1998.
21. M. Mamei, L. Leonardi, M. Mahan, and F. Zambonelli. Motion coordination for ubiquitous agents. In P. Finin and F.Perich, editors, *Workshop on Ubiquitous Agents on embedded, wearable, and mobile devices – First International Joint Conference on Autonomous agents and multiagent systems (AAMAS 2002)*, 2002.
22. R. Milner. *A Calculus of Communicating Systems*, volume 92 of *Lecture Notes in Computer Science*. Springer-Verlag, 1980.
23. A. Omicini. Towards a notion of agent coordination context. In D. Marinescu and C. Lee, editors, *Process Coordination and Ubiquitous Computing*, pages 187–200. CRC Press, 2002.
24. A. Omicini and S. Ossowski. Objective versus subjective coordination in the engineering of agent systems. LNCS, 2002. In this volume.
25. A. Omicini and F. Zambonelli. Coordination for Internet application development. *Autonomous Agents and Multi-Agent Systems*, 2(3):251–269, Sept. 1999. Special Issue: Coordination Mechanisms for Web Agents.

26. A. Omicini, F. Zambonelli, M. Klusch, and R. Tolksdorf, editors. *Coordination of Internet Agents: Models, Technologies, and Applications*. Springer-Verlag, March 2001.
27. G. A. Papadopoulos. Models and technologies for the coordination of Internet agents: A survey. In Omicini et al. [26], chapter 2, pages 25–56.
28. H. V. D. Parunak, S. Brueckner, and J. Sauter. Digital pheromone mechanisms for coordination of unmanned vehicles. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems AAMAS'02*, pages 449–450. ACM Press, 2002.
29. V. D. Parunak. 'Go To The Ant': Engineering principles from natural agent systems. *Annals of Operations Research*, 75:69–101, 1997.
30. V. D. Parunak, S. Brueckener, J. Sauter, and R. Matthews. Distinguishing environmental properties and agent dynamics: A case study in abstraction and alternate modelling technologies. In A. Omicini, R. Tolksdorf, and F. Zambonelli, editors, *Engineering Societies in the Agents World*, volume 1972 of *LNAI*. Springer-Verlag, Dec. 2000.
31. V. D. Parunak, J. Sauter, and S. Clask. Toward the specification and design of industrial synthetic ecosystems. In M. Singh, A. Rao, and M. Wooldridge, editors, *Intelligent Agents IV. Agent Theories, Architectures, and Languages*, volume 1365 of *LNAI*. Springer-Verlag, 1998.
32. A. Ricci, A. Omicini, and E. Denti. Activity theory as a framework for mas coordination. In P. Petta, R. Tolksdorf, and F. Zambonelli, editors, *Engineering Societies in the Agents World III*, LNAI. Springer-Verlag, Dec. 2002.
33. K. Schmidt and L. Bannon. Taking CSCW seriously: Supporting articulation work. *International Journal of Computer Supported Cooperative Work (CSCW)*, 1(1):7–40, 1992.
34. K. Schmidt and C. Simone. Coordination mechanisms: Towards a conceptual foundation of cscw systems design. *International Journal of Computer Supported Cooperative Work (CSCW)*, 5(2–3):155–200, 1996.
35. K. Schmidt and C. Simone. Mind the gap! towards a unified view of CSCW. In *The Fourth International Conference on the Design of Cooperative Systems COOP 2000*, May 2000.
36. M. Schumacher. *Objective Coordination in Multi-Agent System Engineering – Design and Implementation*, volume 2039 of *LNAI*. Springer-Verlag, Apr. 2001.
37. M. Singh. Agent communication languages: Rethinking the principles. *IEEE Computer*, 31(12):40–47, 1998.
38. D. Tennenhouse. Pro-active computing. *Communication of ACM*, 43(5):43–50, May 2000.
39. M. Viroli and A. Omicini. Coordination as a service: Ontological and formal foundation. In A. Brogi and J.-M. Jacquet, editors, *Foundations of Coordination Languages and Software Architectures – Papers from FOCLASA'02*, volume 68 of *Electronic Notes in Theoretical Computer Science*. Elsevier Science B. V., 2002.
40. P. Wegner. Why interaction is more powerful than algorithms. *Communication of ACM*, 40(5):80–91, May 1997.
41. P. Wegner. Interactive foundations of computing. *Theoretical Computer Science*, 192(2), Feb. 1998.
42. P. Wegner. Toward empirical computer science. *The Monist*, 82(1), Jan. 1999.
43. M. P. Wellman. A market-oriented programming environment and its application to distributed multicommodity flow problems. *Journal of Artificial Intelligence Research*, 1:1–23, 1993.

44. M. P. Wellman and P. R. Wurman. Market-aware agents for a multiagent world. *Robotics and Autonomous Systems*, 24:115–125, 1998.
45. F. Zambonelli, N. R. Jennings, A. Omicini, and M. J. Wooldridge. Agent-oriented software engineering for Internet applications. In Omicini et al. [26], chapter 13, pages 326–346.
46. F. Zambonelli and V. D. Parunak. Signs of a revolution in computer science and software engineering. In P. Petta, R. Tolksdorf, and F. Zambonelli, editors, *Working Notes of the third International Workshop on Engineering Societies in the Agent World (ESAW '02)*, Sept. 2002. Technical Report TR-2002-35, Austrian Research Institute for Artificial Intelligence.

The Pragmatics of Software Agents: Analysis and Design of Agent Communication Languages*

Juan Manuel Serrano, Sascha Ossowski, and Alberto Fernández

Artificial Intelligence Group
School of Engineering
University Rey Juan Carlos
{jserrano,sossowski,afernand}@escet.urjc.es
http://www.ia.escet.urjc.es

Abstract. Modern ACLs, such as FIPA ACL, provide standardised catalogues of performatives and protocols, designed as general purpose languages to ensure interoperability among agent systems. However, recent work reports a need for new ad-hoc sets of performatives and protocols in certain contexts, showing that FIPA ACL does not support adequately all relevant types of interactions. In this paper we first present a formal model that relates performatives and protocols to the organisation of MAS. Then, a principled method for the *design* of the ACL of a particular MAS is developed, which accounts for both, reusability and expressiveness. Finally, we illustrate our approach by an example in the domain of agent-based decision support for bus fleet management.

1 Introduction

Since Wittgenstein couched the term "language game" to emphasise the functional and constitutive role that language plays in society [30], the pragmatics of natural languages have been an important topic of research in a variety of disciplines. The concept of language game attempts to show that language is not only a *semantic* device, mainly used to denote things and properties of the outside world, but primarily a *social instrument*. From this perspective, language is an essential means to let people participate in the different types of social interaction, or language games, that are "played" in their society.

Speech act theory [1,21] made the link between social activity and language use more specific, by creating the concept of "illocutionary act", which denotes the kind of particular social action performed *in* uttering some sentence, e.g. informing, promising, warning, and so on. There are many different types of lexical, morphological and syntactical devices, which allow marking the illocutionary force of some message. Among the most important is the lexicon of *speech act verbs*, which represents the *particular* way in which a particular natural language categorises the universe of speech acts [29, p. 10]. Thus, the lexicon of

* Research sponsored by MCyT, project TIC2000-1370-C04-01, and by CAM, project 07T/0011/2000

M. Klusch et al. (Eds.): Intelligent Information Agents, LNAI 2586, pp. 234–273, 2003.

English speech act verbs [29] contains performatives such as *promise* or *warn*, which do not denote culture-independent universal kinds of speech acts, but specific lexical encodings of "language games", or modes of social interaction, that this culture considers particularly relevant [28].

An artificial counterpart of natural language speech act verb lexical is the catalogue of performatives included as part of the Agent Communication Language (ACL) standardised by the Foundation of Intelligent Physical Agents (FIPA). The performatives included in this catalogue, termed FIPA Communicative Act Library (FIPA CAL), denote the types of illocutionary actions (or communicative actions, CAs) which the community of designers interested in the development of Multiagent Systems (MAS) considers especially important. The FIPA CAL includes performatives such as *inform*, *query* or *request*, which agents use to participate in social interactions concerning the exchange of information and the performance of actions, the two most common kinds of interactions in MAS. FIPA ACL also standardises a set of interaction protocols, the FIPA Interaction Protocol Library (IPL), which identify common patterns of agent interaction on the basis of the performatives included in the FIPA CAL.

The field of Discourse Analysis [27] also aims at making the link between language and society more explicit, by exploring the interface between *social organisation* and discourse structure. It particularly emphasises the influence of organisational and institutional concepts, such as *roles*, *groups*, *power*, *ideology*, etc. on the use and structure of language. For instance, language users invariably engage in social interaction as players of particular social *roles*, and members of particular *social groups* or communities.

Organisational concepts and models are not new to the MAS community [6, 31]. Up to some respect, these models already acknowledge the link between organisational concepts and ACLs: most of them make explicit reference to CAs as an essential part of the interaction within an organisation. However, their primary concern is usually the organisational model (OM), whereas the communication language plays a subordinate, facilitating role. One of the few exceptions is the work by Esteva [5,2], where a close relationship between roles and illocutionary particles is established. However, a conceptual framework which provides a comprehensive view of the link between the ACL and the Organisational Model of MAS is yet to be developed.

So, it seems promising to investigate the interplay between the design of the OM and the design of the ACL for some particular MAS, especially with respect to the role that a standard agent communication language may play. A step in this direction has been taken by Pitt and Mamdani [16]. They argue that the use of a standard ACL, such as FIPA ACL, as a common language for the development of any particular MAS should be disregarded, as not all social interactions are adequately supported, a claim backed by the many proposals of new performatives aimed at supporting more explicit types of interaction than the ones addressed by FIPA ACL, such as argumentation-based negotiation[26], team formation[4] or decision support [23]. Instead, Pitt and Mamdani suggest to extend a core ACL in line with the requirements imposed by the particular MAS to be developed.

The purpose of this paper is both fundamental and practical. On the one hand, we aim at providing a comprehensive conceptual framework which defines an integrated view of the OM and the ACL components of a MAS. The UML-based [17] Role/Interaction/Communicative Action (RICA) metamodel formalises this conceptual framework, and serves as a theoretical basis for the analysis of the FIPA ACL from an organisational point of view. On the other hand, setting out from FIPA ACL as a core language, we tackle the practical problem of how to design an ACL for a particular MAS in a principled manner, covering its Communicative Acts as well as its Interaction Protocols, while complying with both, expressiveness and interoperability requirements.

The paper is structured as follows: In Section 2 we introduce the RICA metamodel, and show how the different elements of the FIPA ACL can be structured according to this model. Section 3 describes a RICA-based MAS ACL design method, which is illustrated in Section 4 in the domain of decision support for bus fleet management. Finally, we compare our approach to related work, and discuss future lines of research.

2 An Organisational Perspective on ACLs

2.1 The Role/Interaction/Communicative Action Model

In order to model the link between communicational and organisational components of MAS, the conceptual framework needs to be identified first. Respecting organisational terms, we draw from Ferber and Gutknecht's Agent/Group/Role (AGR) metamodel [6,7], that uses *Agent*, *Group* and *Role* as organisational key concepts. *Group Structures* are composed of a number of roles, which participate in different *Interactions Types.*

Setting out from the work of Esteva et al. [5], we will relate these concepts to terms from ACLs such as *Message*, *Communicative Action* (CA) and *Protocol*, giving rise to our RICA (Role/Interaction/CA) metamodel. Figure 1 shows this model in terms of a UML class diagram [17], where ACL concepts are indicated by white boxes. In this context, the *Communicative Action* class shall be understood as the illocutionary action performed by an *Agent* when sending some *Message* with a given performative particle. In the sequel, we will describe the most relevant concepts and associations, and illustrate them based on a well-known problem from agent-oriented software engineering: the design of an agent-system for the management of an international conference [32].

Role/Role and Class/Role associations. It is customary in the literature on organisation in MAS to conceive the concept of *role* in terms of the functionality (or behaviour) of some type or *class* of agents [6,31,5]. Moreover, agents belonging to a certain class often play several roles during their life-cycle, either at the same time, or consecutively.

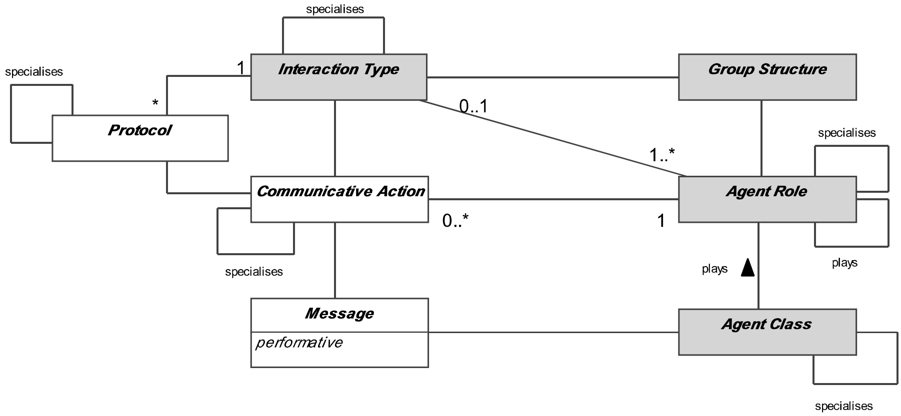

Fig. 1. UML diagram of the RICA meta-model

There are two major relationships involving classes of agents: *generalisation/specialisation* of classes, as well as a *plays* relation with respect to roles. For instance, in a MAS for conference management, we may distinguish two major classes of agents: *Researchers* and *Senior Researchers*. The latter can be considered as a specialisation of the former. Senior researchers decide which role(s) each of them will play for conference management (e.g. the *PC Chair* and *PC Member*). They are also in charge of finding other senior researchers to play the role of *Reviewers*. *Researchers* are allowed to play the role of (potential) authors of the conference. As the *Senior Researcher* class constitutes a specialisation of the *Researcher* class, it inherits all the roles attached to the latter, thus allowing senior researchers to write papers as well.

The RICA metamodel follows [10] in establishing two kinds of relationships between roles: roles can be generalised/specialised, and roles may play other additional roles. With respect to the former relation, different classes of reviewers might be considered, taking into account their area of expertise (e.g. *ACL Reviewer*, *OM Reviewer*, etc). Respecting the latter, authors may play the role of *accepted authors*. Figure 2 summarises the class and role hierarchies of the conference example.

In the object-oriented systems literature (e.g. [10]), the difference between classes and roles is conceptualised in terms of the dynamic vs. static nature of the modelled entity. Class hierarchies allow designers to statically classify entities into a set of disjoint types, i.e. entities that have been classified into one class are not expected to change that class during their lifetime. In our example, agents are classified either as senior researchers, or as researchers for a certain conference. This classification will not change during the management of the conference [1].

[1] The static vs. dynamic nature of some entity may depend on the context of the application. For instance, a *Researcher* can be considered a *role*, rather than a *class* of agents, in the context of an application dealing with careers or professions.

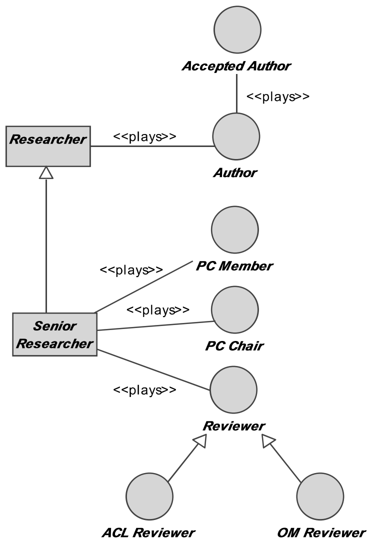

Fig. 2. UML class model for the conference example

On the contrary, a senior researcher may come to play either the role of the PC Chair or the PC member; she may perform the roles of an ACL and OM reviewer at the same time, etc. In much the same way, if their submission is successful, authors may come to play the role of *accepted authors*. With respect to the generalisation relationship between classes of reviewers, there might be no agreement at all on which should be the right classification of expertise areas, but once a classification is accepted, the corresponding generalisation relationship is static.

Communicative Action/Role association. In general, both the communicative and non-communicative behaviour (or *private* actions), together with a set of rights or permissions, are necessary to fully specify a role [31]. The relative importance of the communicative and non-communicative behaviour for the characterisation of roles is conceived differently in different models. As we are concerned with the interplay between organisational and communicative concepts, our standpoint in this paper is similar to [5], where agent interaction is purely dialogical and, consequently, roles are exclusively defined in terms of a set of illocutionary actions. So, although (non-communicative) interaction with the environment is undoubtedly important, in the sequel we focus on the communicative competence of agents. In the conference management example, *submissions* are types of CAs performed by authors. PC members are required to *ask* reviewers for the evaluation of papers, *suggest* or *recommend* whether some submitted paper should be accepted or not, etc. Finally, the PC chair has the responsibility of *accepting* or *rejecting* them.

We may wonder as to how far the illocutionary actions identified for the roles of our example are really *characteristic* to them. It is a commonplace that suggestions and recommendations are not only issued by a PC member, but may be realised by different roles in other application domains as well (e.g. an intelligent tutor may recommend the students certain guidelines for preparing an exam). This suggests the existence of a more abstract role, the *advisor*, which the intelligent tutor, the PC member, and many other roles may *play*. Thus, although suggestions and recommendations are performed by many different roles, these CAs and some others will only be *directly* performed by the advisory role (see section 4.2 for a more detailed analysis of this role).

If a role is a specialisation of another, the former obviously inherits the attributes of the latter, and in particular all the CAs that characterise the super-role. The sub-role may add new illocutionary actions to the catalogue of the super-role, or may specialise some of these CAs, just as class methods may be overloaded within subclasses in object-oriented systems. Typically, this specialisation of CAs is carried out by overloading the propositional content, as it is the case of *query-if* with respect to *request*[2]. It may also happen that the illocutionary catalogue of the sub-role is identical to the super-role. Then, the introduction of the new role might be justified by additional rights or permissions, private functionality, particular protocols, etc. In our example, the illocutionary actions performed by reviewers of a specific research area coincide with the ones of a general reviewer. Still, it is convenient to distinguish between these roles based on non-communicative aspects of their ontologies (in particular, on the domain knowledge necessary to play a role).

In a nutshell, the one-to-many Role/CA association of the RICA metamodel refers to the relation between a CA and its *characteristic* role:

> The *characteristic* role of some type of CA is the most *generic* role which is entitled to *directly* perform it (i.e. without playing another role).

It should be noted that the characteristic role of a CA is *unique*. By contrast, a given role can usually count on several characteristic CAs. Sometimes, a role may not have any characteristic CA at all, when its communicative behaviour can be described in terms of the pragmatic competence of its super-classes, or when its set of CAs is covered by the other roles that it is able to perform.

Role/Interaction association. The one-to-many association *Role/Interaction* is another key relation of the RICA meta-model. The term *interaction* is used here in the same way as the term *protocol* is used in [31], i.e. to refer to the kind of social interaction in which some group of roles participates, specified in terms of the *purpose* of that interaction, rather than in terms of the prototypical pattern of message exchange. The word *protocol* is reserved for this latter sense. We may

[2] In the FIPA semantic framework, the specialisation of CAs can be achieved by adding further restrictions not only to the propositional content of an illocutionary action, but also to its feasibility precondition and rational effect [23].

say that illocutionary actions are to roles, what (types of) social interactions are to groups of roles.

It is also customary in the organisational literature to acknowledge that roles may participate in several types of interaction [6]. In our example, the PC chair and PC members may interact in order to evaluate the submitted papers, based upon the collected reviews. We may call this type of interaction *Paper Decision Making*. However, agents playing both kinds of roles will also participate in interactions about the *assignment of papers* to the different reviewers.

To account for this multiple cardinality we make use of the *plays*-relation between roles. The PC chair is able to participate in both types of interaction because she performs two different roles: in the paper decision making, she plays the role of *Paper Responsible*, i.e. she is responsible for taking the decision of accepting or rejecting some paper. In the interaction about paper assignment she will play the role of the *Review Coordinator*. Equivalently, PC members will participate in both types of interaction playing the roles of *Paper Advisor* and *Review Coordinee*. From this point of view, we may conceive complex roles, such as the PC member and the PC Chair, to participate *indirectly* in all types of social interaction which are characteristic to the roles they may play at some moment.

The role/social interaction relationship of the RICA metamodel, accounts for the *only* characteristic social interaction in which some role *may* directly participate (in case that a role is only playing some additional roles, there may even be no such interaction). On the other hand, we have seen that there are usually several characteristic roles for a certain social interaction. Still, although more than one agent is required to enact a social interaction, these agents may play the same role, so the lower cardinality at the role-side of the association is one.

Note that the specialisation relationship between roles also induces a specialisation relationship between the corresponding types of social interaction. For instance, a *Paper Advisor* can be modelled as a specialisation of the *Decision Making (DM) Advisor* role. Consequently, the *Paper Decision Making* interaction will be a specialisation of the *Decision Making* interaction, in which several DM advisors and one *DM Responsible* agent participate. This latter role can be considered as a generalisation of the *Paper Responsible* role. Figure 3 summarises the role/social interaction association of the conference example by means of an UML class diagram[3].

Social Interaction/Protocol association. By introducing a one-to-many relation between *protocols* and *social interactions*, we account for the fact that *several* protocols may relate to the same topic. Moreover, our definition acknowledges that, in case that agents with strong reasoning capabilities are able to dynamically plan for their future communicative actions, there will be no need for a protocol.

[3] if a use case diagram was chosen instead, the actor would represent the different roles while the use cases would model the types of social interactions.

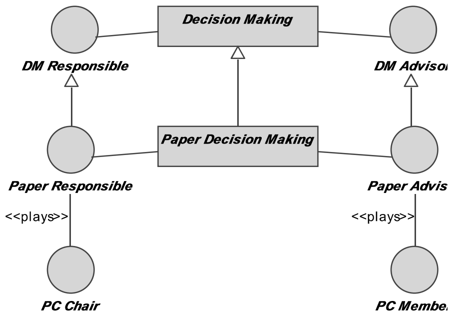

Fig. 3. UML class model illustrating the Role/Social Interaction association

The types of CAs available to define the protocols of some social interaction are limited to the ones associated to the roles participating in that interaction. These CAs can be further classified into those which are characteristic to the participating role, and those inherited from super-roles. Note that protocols should not be defined for a type of interaction whose roles do not have characteristic communicative actions. For instance, as the catalogue of communicative acts of the roles *Paper Responsible* and *Paper Advisor* do not contain any new performative besides the ones inherited from the *DM Advisor* (e.g. suggests, recommend, warn, etc.) and the *DM Responsible* (e.g. accept, reject, etc.) roles, the particular protocols that we may devise for the *Paper Decision Making* interaction should be defined at the more abstract level of the *Decision Making* interaction.

Again, given that social interactions may be specialised, the protocols attached to the super-interaction will be inherited by the sub-interaction, which might add new protocols in case that its roles define new characteristic communicative acts. Moreover, in line with the overloading of communicative actions, interaction protocols from a super-interaction may be specialised in the sub-interaction. Some examples of this kind of relationship between protocols will be discussed in the next subsection.

The Organisational Structure of an ACL. As Table 1 shows, setting out from the RICA meta-model, a classification scheme for the performative lexicon and the protocols of a given ACL can be derived. Table 1 is divided into three major rows: row one includes the types of interaction supported by the ACL, while rows two and three classify roles, together with their characteristic CAs, and protocols accordingly. Note that the number of roles need not be the same among different interactions. Columns are guaranteed to be disjunct, as in the RICA meta-model performatives and protocols are characteristic of a single type of interaction, and performatives are characteristic of a single role.

Table 1. Classification Scheme for a particular ACL

Agent Communication Language		
Interaction Type 1	Interaction Type 2	...
Communicative Act Library		
Role 1-1	Role 2-1	...
performative 1-1-1	...	...
performative 1-1-2	...	...
...	...	...
Role 1-N	Role 2-M	...
performative 1-N-1	...	...
...	...	...
Interaction Protocol Library		
protocol 1-1	*protocol 2-1*	...
...	...	...

2.2 The Organisational Structure of the FIPA ACL

In this section we analyse the OM underlying FIPA ACL on the basis of the aforementioned findings, thereby providing the basis for the ACL design method to be outlined in Section 3.

FIPA ACL Organisational Structure. From the point of view of pragmatics, the Communicative Act library FIPA CAL and the library of protocols FIPA IPL are the most relevant components of FIPA ACL. In the FIPA standard, the former is not structured in any relevant sense, while the latter just refers to very generic roles such as *Initiator* and *Participant* [4]. In the sequel we will first structure these components according to the classification scheme of Table 1, so as to subsequently infer generalisation relationships based on the implicit OM that underlies FIPA ACL.

The semantics of FIPA CAL suggests that five major types of interactions are supported, which we have called *Message Exchange*, *Action Performing I*, *Information Exchange*, *Action Performing II* and *Brokering*. The 22 communicative actions of the FIPA CAL[5], and 11 protocols of the FIPA IPL, are classified accordingly. The CAs can be further structured into two sets corresponding to the two roles which participate in each type of social interaction.

As an example, consider the interaction *action-performing*, involving a requester and a requestee. The former aims at achieving some goal, while the latter role is supposed to volunteer its services to the requester. The CAs characteristic to the requester agent are *request*, *request-when*, *requestwhenever* and *cancel*. The first one is the most common, and refers to some action a which the requester intends to be performed by the requestee. The second and third can be regarded as specialisations of the *request* action. The last one, *cancel*, allows the requester to tell the requestee that it changed its mind, and does not any

[4] The only exception are the protocols associated to brokering interactions, where the labels *Broker* and *Recruiter* are used instead.

[5] The communicative actions are described as $f(p_1, ..., p_n)$, where f is the illocutionary force of the performative and p_i are the components of the propositional content.

Table 2. Organisational Structure of the FIPA CAL and FIPA IPL

FIPA ACL				
Message Exchange	**Action Performing I**	**Information Exchange**	**Action Performing II**	**Brokering**
FIPA CAL				
Communicator	**Requester I**	**Information Seeker**	**Requester II**	**Brokering Requester**
inform(ϕ) *confirm*(ϕ) *disconfirm*(ϕ) *not-understood*(a, ϕ)	*request*(a) *request-when*($<j, act>, \phi$) *request-whenever*($<j, act>, \phi$) *cancel*(a)	*query-if*(ϕ) *query-ref*($Ref\ x\ \delta(x)$) *subscribe*($Ref\ x\ \delta(x)$)	*cfp*($<j, act>, Ref\ x\ \phi(x)$) *accept-proposal*($<j, act>, \phi$) *reject-proposal*($<j, act>, \phi, \psi$)	*propagate*($Ref\ x\ \delta(x), <i, cact>, \phi$) *proxy*($Ref\ x\ \delta(x), <j, cact>, \phi$)
	Requestee I	**Information Provider**	**Requestee II**	**Broker**
	agree($<i, act>, \phi$) *refuse*($<i, act>, \phi$) *failure*(a, ϕ)	*inform-if*(ϕ) *inform-ref*($Ref\ x\ \delta(x)$)	*propose*($<i, act>, \phi$)	
FIPA IPL				
	FIPA-Request-Protocol *FIPA-Request-When-Protocol*	*FIPA-Query-Protocol* *FIPA-Subscribe-Protocol*	*FIPA-Propose-Protocol* *FIPA-ContractNet-Protocol* *FIPA-IteratedContractNet-Protocol* *FIPA-English-Auction-Protocol* *FIPA-Dutch-Auction-Protocol*	*FIPA-Brokering-Protocol* *FIPA-Recruiting-Protocol*

longer want *a* to be performed. By means of *agree* and *refuse* CAs, the requestee can make public its (positive or negative, respectively) intention to collaborate. Finally, the *failure* CA allows the requestee to communicate the impossibility of performing the requested action.

As stated before, protocols can be associated to interactions based on the CAs that support them. In this way, for instance, both the FIPA-Query-Protocol and FIPA-Subscribe-Protocol correspond to patterns of *information exchange* interactions, as their most relevant performatives – *query-if, query-ref* and *subscribe* – are characteristic to that interaction type.

Besides this classification, the RICA model also accounts for an analysis of the different relationships of generalisation and specialisation that hold between the roles already identified above. In order to identify the role hierarchy, we first focus on the performatives that are used in the definition of non-primitive performatives of a role, and then on the performatives of other roles that are used in its protocols. So, for instance, the characteristic performatives of the *Information Seeker*, (*query-if*, *query-ref* and *subscribe*), are all defined in terms of performatives that are characteristic of the *Requester* role (*request* and *request-whenever*). Moreover, the *cancel* performative is used in the *FIPA-Subscribe-Protocol*, which shows that this performative, which is characteristic to the requester, is also characteristic to the information seeker. Thus, the information seeker can be considered a specialisation of the requester role. The results of this analysis are summarised as a UML class diagram in Figure 4 using the usual abbreviated stereotype for roles.

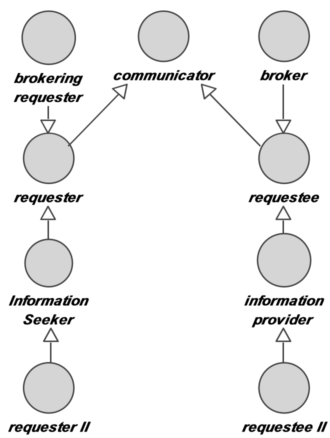

Fig. 4. Role model Implicit in the FIPA CAL

FIPA IPL Restructuring. As discussed in section 2.1, the RICA meta-model implies a specialisation relationship among protocols corresponding to generic types of social interactions. This section proposes a restructuring of the set of protocols included in FIPA IPL, which takes advantage of this specialisation mechanism. The proposal will be illustrated taking into account the *FIPA-Query-Protocol* and *FIPA-Request-Protocol*, corresponding to the *Information Exchange* and *Action Performing I* interaction types, respectively.

The generalisation relationship described in figure 4, between the *Information Seeker* and the *Requester*, shows that the exchange of information can be conceived as a particular case of interaction about action performing, where the action to be performed is to inform the requester about some fact. Therefore, information exchange protocols like the *FIPA-Query-Protocol*, could be specified based on a specialisation of the *FIPA-Request-Protocol*. Moreover, action performing interactions, and any other interaction type, are particular cases of message exchange. Thus, the *FIPA-Request-Protocol* could be defined as a refinement of some protocol for message exchange (not defined by FIPA).

In the sequel we will define these protocols in terms of their only characteristic types of CAs, which are described in table 2. UML State diagrams have been chosen to represent the interaction protocols. Moreover, the protocols are defined in accordance with [2], where state transitions are triggered by messages from both roles, and states are considered to be *interaction states*, not *agent states*. The labels used for each role participating in the interaction, are defined in a simple UML *note* element. Figures 5, 6 and 7 depict the corresponding protocols.

Message Exchange Protocol. As described in figure 5, any kind of interaction may be in two states: a normal one in which interaction unfolds smoothly, and one in which normal interaction has been interrupted due to a *not understood* message. We have termed the latter case the *clarification* state. From here, the protocol allows either to return to the normal state, or requires the termination of the interaction.

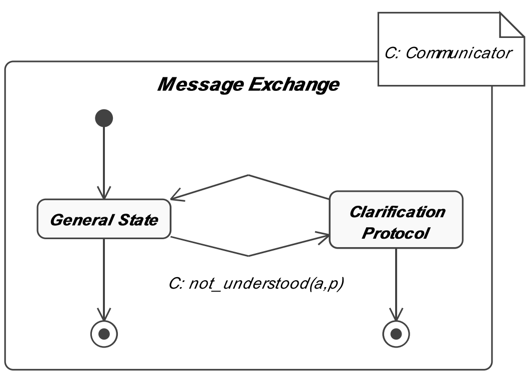

Fig. 5. Message-Exchange Interaction Protocol

Action Performing Protocol. The Action Performing interaction protocol also relies on two different states (see figure 6). The first state is entered whenever the requester issues a request, and represents the situation in which this request is being evaluated by the requestee. Upon an agreement of the requestee, the interaction enters a new state where the requester is waiting for the performance of that action.

There are two major differences between this protocol and the FIPA-Request-Protocol. First, it is not necessary to explicitly send an *agree* message in case that the requested action is to be performed immediately. Second, once the action has been performed, it is not obligatory to inform the requester that the action was successfully or unsuccessfully performed, if the action performance was made publicly. In that case, the interaction finishes without the need for any additional messages.

It should be noted that the state diagram shown in figure 6 does not represent the whole protocol of the action performing interaction. Rather, this state diagram should be considered as a specialisation of the *Message Exchange* state named *General State*. Thus, the whole protocol for action performing includes the *Clarification Protocol* state as well. In this way, the specialisation of protocols proceeds by adding new states and transitions to the generic protocol, similarly to [12,15].

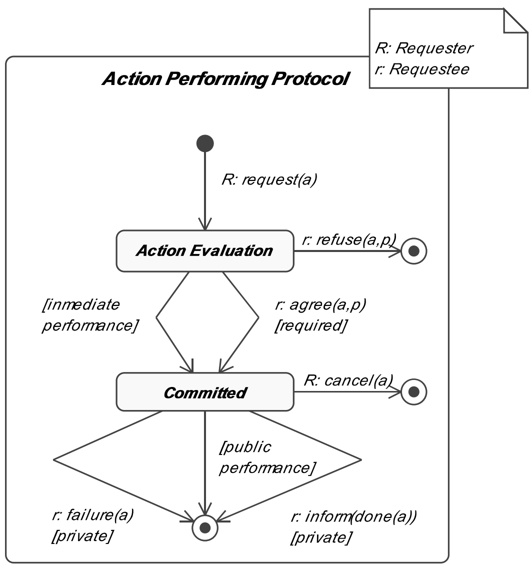

Fig. 6. Action Performing Interaction Protocol

Information Exchange Protocol. Figure 7 describes the *FIPA-Query-Protocol* as a specialisation of the Action-Performing-Protocol described previously. It relies on the same substates as the former, but adds a number of new transitions. The first one, from the initial state to the *action evaluation* state, shows that the interaction may start with a *query-ref* message. As this type of CA is a kind of request, this transition overrides the *request* message which initiates action performing interactions. Second, the protocol may finish with the requestee *informing* the requester about some fact. This transition specialises the Action-Performing-Protocol transition corresponding to a public performance of the requested action. No new transitions from the *action evaluation* to the *committed* states are needed, besides those found in the generic protocol.

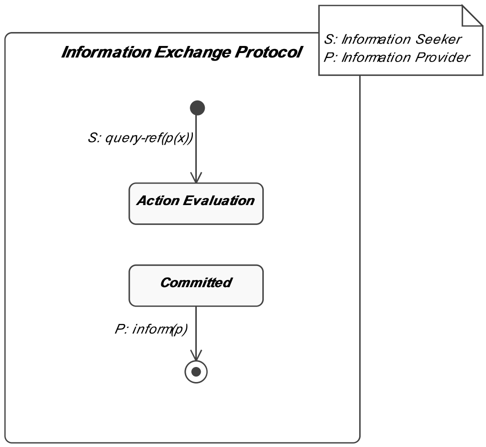

Fig. 7. Information Exchange Interaction Protocol

3 A Design Method for the ACL of a Multiagent System

Once the close relation between OMs and an ACLs is formalised, the question arises as to how an ACL for a *particular* Multiagent System (MAS ACL) can be defined. This section outlines our proposal for a design method for MAS ACLs. However, before describing the different steps to follow, and their relation to the RICA metamodel, the essential characteristics that any such approach must meet are stated.

3.1 Design Requirements

The development of any ACL design method should be guided by the following design goals: to allow for expressive interactions within the domain, to account for reusability of standard ACLs (particularly, FIPA ACL), to ensure interoperability, and to be consistent with other components of the MAS architecture:

- By *expressiveness* we essentially mean the closeness of the ACL performatives to the particular characteristics of the domain, as exemplified in the linguistic intuitions of the MAS designers and users. For instance, the behaviour of an advisory agent can either be described in terms of the FIPA performative *inform*, or by means of new more expressive performatives such as *suggest*, *recommend*, etc.
- Concerning *reusability*, the standard performatives and protocols included in FIPA ACL should be reused whenever they adequately describe the agents' communicative behaviour. When defining extensions to the FIPA ACL for particular applications, a potential reuse of performatives and protocols pertaining to existing extensions should be supported.
- *Interoperability* ensures that agents developed by different designers can effectively interact with each other. This requires that performatives used in the exchanged messages are adequately defined, in order to avoid misinterpretations.
- In the context of this paper, consistency with the MAS architecture is understood with respect to the close relationship between organisational and communicative concepts, as argued in section 2. In this sense, the design of the MAS CAL should take into account the organisational model of the application, making explicit the impact of organisational notions and structures on ACL design decisions.

3.2 Design Method

Figure 8 summarises our design proposal in terms of an UML activity diagram [17]. The first input of the design process is an initial version of the MAS OM, which specifies the domain-dependent types of interaction and roles played by the different classes of agents in the MAS. The whole design process is driven by this initial OM. The other major input is the existing FIPA ACL standard (including its two major components: the FIPA CAL and FIPA IPL). The third input is a natural language (NL) catalogue of speech act verbs, such as [29]. As a result, the MAS CAL and MAS IPL are obtained, structured in terms of the types of interaction and roles included in a *refined* version of the initial OM. In the sequel we outline the different steps of our design method.

(1) Collecting Sample Dialogues

The first step is to collect natural language sample dialogues for the different domain-dependent types of interaction, in which the different classes of agents participate holding particular roles. Thus the collection of sample dialogues will be structured in terms of the initial OM. These dialogues may be transcripts of real conversations between human agents playing those roles (as for instance, the dialogue between some human tutor and her student, in case of development of a tutoring agent), or artificially designed dialogues showing prototypical patterns of interaction.

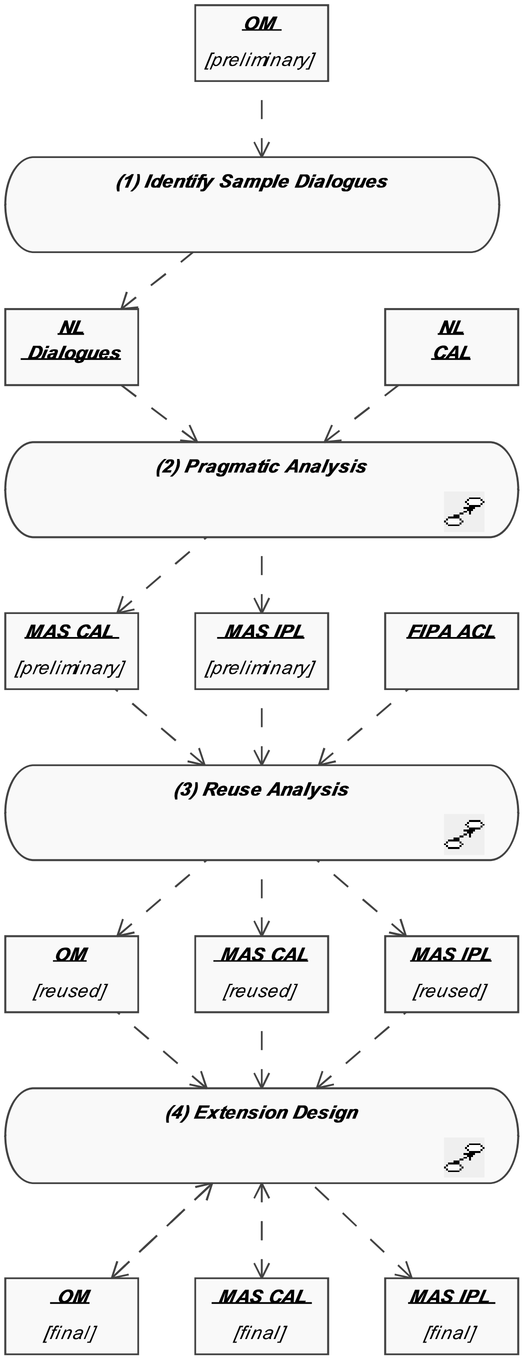

Fig. 8. Design Task Specification

(2) Pragmatic Analysis

The second step is a pragmatic analysis of the collection of sample dialogues, attempting to identify both, the types of CAs that best describe the communicative behaviour of agents in the dialogues, together with their patterns of message exchange:

– At this stage, it is convenient to express the communicative actions by means of a set of natural language speech act verbs or nouns (or even general expressions, if there is no adequate lexical item to denote that illocutionary action), so as to satisfy the *expressiveness* requirement. In order to do so, a reference to natural language dictionaries, such as Wierzbicka's dictionary of English speech act verbs [29], may be helpful [6].
– For the design of the preliminary MAS IPL, it is necessary to identify the patterns of message exchange within the dialogues, taking into account the illocutionary actions identified previously. UML sequence diagrams [17] provide an adequate framework to express these patterns.

Thus, a first natural-language version of both, the MAS CAL and the MAS IPL, is obtained from this step, structured according to the domain-dependent types of interaction and roles of the initial OM.

(3) Reuse Analysis

In order to cope with the *reusability* requirement, the standard FIPA CAL and FIPA IPL are matched against the preliminary versions of the MAS CAL and MAS IPL, respectively. The goal is to identify those standard performatives and/or interaction protocols which do not show significant differences with respect to some of the natural-language speech act verbs and patterns of message exchanges included in the preliminary versions of the MAS CAL and MAS IPL.

As a side-effect, a first refinement of the OM is produced, as certain roles performing some types of CAs in the initial OM are modelled as specialisations of the roles implicit in the FIPA ACL (see section 2.2).

(4) Extension Design

The previous step may have uncovered limitations of the FIPA ACL to account for some kinds of illocutionary expressions and/or dialogue patterns that are characteristic for the particular MAS. Figure 9 shows the subactivities required to design new MAS CAL components that cope with this problem.

[6] In this book, Wierzbicka aims at an analysis of the English-categorisation of the universe of speech acts. She provides definitions for around 200 English speech act verbs, which are classified in groups according to their similarity.

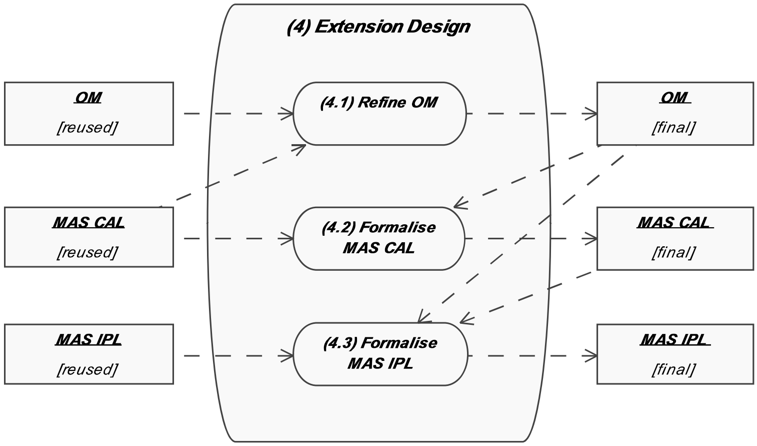

Fig. 9. Subactivities required to design the extended components of the MAS ACL

(4.1) Refinement of the Organisational Model

The "problematic" illocutionary expressions and dialogue patterns often correspond to some type of interaction which is not supported by FIPA ACL[7]. As a result, a new extension of the corresponding part of the initial organisational model is performed. This refinement consists of abstracting generic roles (and their particular interaction types) from the domain-dependent ones, by analysing their characteristic illocutionary expressions included in the preliminary MAS CAL.

This role-abstracting mechanism, grounded on speech act analysis, allows to foster *reusability* of the extensions performed to the FIPA CAL. Nevertheless, the reuse may not always be possible, so that we might come up with ad-hoc types of interaction. This possibility is a reflection of our trade-off between expressiveness and reusability requirements.

(4.2) Communicative Act Library Design

In this second sub-step, the preliminary natural illocutionary expressions not covered by the FIPA CAL, are formalised into a set of artificial CAs, retaining much of the meaning of their natural language counterparts, but adapting them to the context of artificial agents. To account for the *interoperability* requirement, two languages are proposed in order to precisely formalise the performative's meaning: the *Natural Semantic Metalanguage* (NSM) [29], and the *FIPA Semantic Language* (SL) used in the FIPA CAL [9].

[7] Although this must not be the case. For instance, in the context of decision-support systems, the responsible role may ask the decision-support agent for alternative actions, which could achieve some given goal. This communicative action, similar in some respects to the FIPA `cfp` action, could be part of the requester II role of the FIPA ACL [23].

The NSM *language.* The NSM language is the metalanguage used in [29] to state the meaning of the English verbs. It consists of a set of nearly 60 English primitive terms, although the speech act verb definitions of [29] rely on a slightly more complex basic vocabulary. The illocutionary force of a speech act verb is stated by its decomposition into an *ordered* sequence of semantic components. The two most important components are the 'dictum' and the 'illocutionary purpose'. The former represents the "overt content of the utterance", and is encoded in the frame: "I say: ...". The latter represents the "speaker's (purported) intention in making that utterance", and is included in the frame "I say this because ...". The other components are different types of "assumptions, emotions, thoughts and intentions". A prototypical definition in the NSM language is sketched as follows:

Definition 1 *Performative*

I think ...
I say: ...
I say this because ...

This categorisation shows important analogies with the work of Searle and Vanderveken [22]: The illocutionary purpose corresponds to the illocutionary point. The propositional content and the sincerity conditions can normally be found in the dictum. The other components may be conceived as different preparatory conditions (including the mode of achievement).

The FIPA SL *Language.* The FIPA SL language provides the semantic framework underlying the formalisation of CAs. It sets out from a first-order language, including modal operators of different mental attitudes (belief, uncertainty and choice) and actions (feasible, done)[19]. Uncertainty is such that the following set of formulae are mutually exclusive [18]: $\{B_i\neg\phi, U_i\neg\phi, U_i\phi, B_i\phi\}$.

With respect to the modelling of actions, their meaning is given in terms of a set of formulae, which is structured in two major parts. The first one establishes the *rational effect* (RE) of the action, i.e. the reasons to choose it. The second one states the *feasibility preconditions* (FP) which must be fulfilled if the action is to be executed. In the case of communicative actions, the RE corresponds to the perlocutionary effect of the CA, and the FPs can be further decomposed into *ability preconditions* and *context-relevance* preconditions [18]. The following scheme shows the general model of the definition for some given *performative* in FIPA SL:

$$
\begin{aligned}
&< i, Performative(j, \text{``}propositional\ content\text{''}) > \\
&\quad FP: \text{``}ability\ preconditions\text{''} \wedge \\
&\qquad\quad \text{``}context\ relevance\ preconditions\text{''} \\
&\quad RE: \text{``}rational\ effect\text{''}
\end{aligned}
$$

Compared to the Searle and Vanderveken categories, the rational effect and the feasibility preconditions correspond to the illocutionary point and preparatory conditions, respectively. The ability precondition can be conceived as a generalisation of the sincerity condition [18].

(4.3) Interaction Protocol Library Design

Based upon the refined organisational model and the final MAS CAL, the MAS IPL is designed, so as to account for those message exchanges that are not adequately supported by the protocols included in the FIPA IPL. These protocols may (but need not) correspond to new types of social interactions (not supported by FIPA ACL). In any case, IPs are specified exploiting the specialisation mechanism of protocols described in section 2.1. UML state diagrams appear to be an appropriate specification language.

In summary, the final MAS ACL is obtained from two consecutive refinements of an initial natural-language MAS ACL. The first refinement allows for the reuse of some components of the FIPA ACL, whereas the second aims at the formalisation of the components for which reuse is not possible. Note that the whole design process is driven by the organisational model of the MAS. Accordingly, the types of social interaction supported by the MAS ACL can be divided into those reused from the implicit organisational model of the FIPA ACL, and those which are not supported by the standard. The latter can be further classified into potentially reusable and idiosyncratic ones.

Table 3. Structure of the MAS ACL

MAS ACL				
FIPA Inter. Type 1	...	Inter. Type 2	...	...
CAL				
FIPA Role 1-1	...	Role 2-1	...	...
FIPA performative 1-1-1 *[NEW-performative 1-1]* ...	...	*performative 2-1-1* ...	...	...
FIPA Role 1-2	...	Role 2-2	...	...
performative 1-N-1 ...	...	...	...	...
IPL				
protocol 1-1 *[NEW-Protocol-1]* ...	...	*protocol 2-1* ...	...	...
FIPAACLReuse		PotentiallyReusable		Idiosyncratic

4 Example: The ACL of a Bus Fleet Management System

This section illustrates the design method described previously with an example in the domain of decision support for bus fleet management. An introduction to the problem domain is given first. Subsequently, the application of the different steps of our design method is described. Finally, the resulting bus fleet management MAS ACL is summarised.

4.1 Bus Fleet Management Domain

Advances in telematic infrastructures are currently producing revolution in almost all modalities of transportation. On-board information systems help drivers to find their way through an ever enlarging road network. The growing number of loop detectors under the road pavement provide traffic management authorities with an increasingly complete picture of the current traffic state, enabling them to take road traffic management decisions at a more strategic level [11]. Dedicated short range communication devices make it possible to dynamically grant traffic preference to emergency services (opening "corridors" for ambulances etc.).

Public transport has not been diverted from this trend. In particular, buses in modern fleets are equipped with GPS devices, providing information to control operators in a Bus Fleet Management Centre (BFMC) that enables them to determine the current location of buses, estimate the time of arrival at bus stops etc., so as to take the necessary actions to maintain an appropriate quality of this public transportation service.

A typical task of BFMC operators is to detect problems by comparing the timetable of all lines and buses with the current data. The most frequent incidents that can be detected in this way fall into the following categories: *individual delay* (one of the buses in a line is delayed), *generalised delay* (several buses are delayed), *advance* (a bus arrives at a stop before the expected time), *individual saturation* (some people cannot take a bus because it is full), *generalised saturation* (all buses on a line are full) or the *breakdown* of a bus. When such incidents occur, operators need to devise a management plan, containing a set of control action for the drivers, with the aim of minimising the impact of these problems on the overall quality of service. Examples of control actions include *increase/reduce speed* (of an individual bus), change *timetable regulation* (each bus must arrive to the stop at a fixed time) to *frequency regulation* (a bus must arrive a stop every x minutes) or vice versa, *timetable rotation* in a line (each bus in a line adopts the scheduled timetable of its successor), *reinforce* a line with a bus from another line, etc. It is obvious that the less time the operator wastes in taking such management decisions the less will be the impact of the incident on the transportation service from the point of view of the passengers.

Due to the quantity of data that operators receive and the complexity of their reasoning, a Decision Support System (DSS) is necessary to ensure a real-time response of sufficient quality. Modern agent-based DSS can be conceived as Intelligent Decision-making Assistants (IDEAS) [13] that render support to their human operators during the various stages of their decision-making process by means of flexible dialogues. It is essential to notice in this respect that a DSS is not a substitute of the operator, but a tool that helps her to better understand the meaning of the data, and to explore potential consequences of her control actions. The final decision (and the responsibility for it) stays with the operator. As a consequence, it is important that the DSS be able to *explain* its reasoning in the course of a decision support conversation, so as to increase the operators' confidence in the control proposals.

In the sequel we will illustrate our ACL design method by the example of an agent-based DSS for bus fleet management.

4.2 Design of the Bus Fleet Management ACL

In order to design the *Bus Fleet Management CAL*, we depart from a simplified organisational model of the application. Figure 10 shows this model, where the bus fleet operator and bus fleet management system classes are identified. These agents play several roles as they participate in three different types of interaction: bus fleet *information exchange*, bus fleet *control* (interactions on the orders given to bus drivers), and bus fleet management *advice* for the operator.

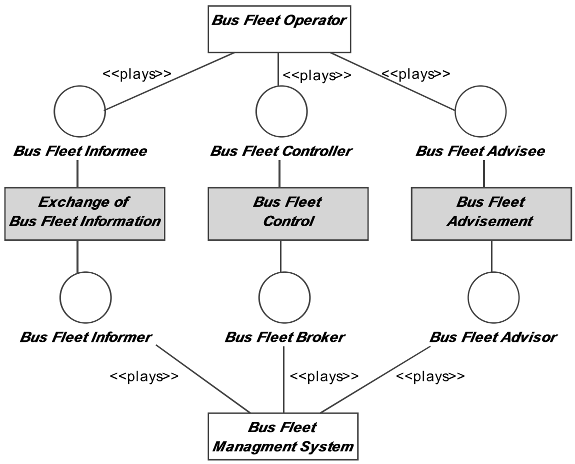

Fig. 10. Initial Organisational Model for the Bus Fleet Management Application

(Step 1) Collecting Sample Dialogues

Table 4 shows a typical natural language dialogue between a bus fleet operator (BFO) and the bus fleet management system (BFMS), whose topic is the detection of a delay problem in a line and the decision on control actions to resolve it. It covers the three types of interaction in which BFMS and BFO participate, according to the initial organisational model shown in figure 10 (distinct levels of gray indicate the corresponding subdialogues, numbered I, II and III, respectively).

The interaction starts with a short exchange of information about the situation of the bus fleet, which is centred around the delay of some bus (I). Next, a sub-conversation concerning a potential control action is initiated by the operator (II), which is temporarily suspended as the BFMS takes on the role of an advisor, trying to raise the BFO's concern on the suitability of the proposed action. An advisory interaction respecting the management of the bus fleet proceeds, where new actions to solve the problem are put forward by the BFMS (III). Then, the conversation focuses again on information matters, concerning the meaning (explanation) of the management information and the impact of

Table 4. Sample Dialogue between Bus Fleet Management System (S) and Operator (O)

(1)	O-	Which is the situation of bus 'B28'?
(2)	S-	Bus 'B28' is delayed
(3)	O-	Bus 'B28' must increase the speed
(4)	S-	That action surely won't resolve the problem in line 'L3'
(5)	O-	Can you explain why?
(6)	S-	Because bus 'B28' is in street 'X' and there is dense traffic in that area
(7)	O-	What action could be useful?
(8)	S-	There are two alternative control actions: a) to do a timetable rotation b) to introduce a bus from another line
(9)	O-	Can you explain why action a)?
(10)	S-	Because there is a generalised delay in line 'L3'
(11)	O-	How did you infer it?
(12)	S-	Because 'B28', 'B29' and 'B30' are delayed, and the delay of some line is qualified as "generalised" if ...
(13)	O-	Which is the impact action a) can produce to the passengers?
(14)	S-	75 person-minute
(15)	O-	Forget the initial control action
(16)	O-	Communicate timetable rotation in line 'L3'
(17)	S-	... Control action forwarded

Exchange of Bus Fleet Information
Bus Fleet Control
Bus Fleet Advisement

the new actions being considered (I). The dialogue finishes by taking up again the theme of bus fleet control (II).

(Step 2) Pragmatic Analysis

This step aims at obtaining a preliminary version of the BFMS CAL, containing those illocutionary expressions which best describe the communicative purposes of the BFMS and the BFO in the sample dialogue of table 4. For this purpose, we use the aforementioned dictionary of English speech act verbs [29] as a major reference. In addition, the first steps towards the design of the BFMS IPL will be given, by identifying the patterns of message exchange for each sub-dialogue by means of UML sequence diagrams [17]. The analysis will be structured in line with the initial types of interaction of figure 10.

Exchange of Bus Fleet Information. The interaction between the BFMS and the BFO concerning the exchange of information about the bus fleet being monitored, can be subdivided into three independent conversations. The first one covers utterances (1) and (2). The BFO *asks a question* to the BFMS, concerning the situation of some bus[8]. This search for information is answered with the second utterance, which can be simply reported by the speech act verb *inform*. Utterances (13) and (14) can be also interpreted in terms of a search for

[8] The illocutionary force of message (1) is not simply spelled out as *ask*, as this verb has also the sense of *asking someone to do something*, which is a different illocutionary act [29, p. 49].

information, in this case about the future state of the bus fleet if some control action is taken.

The nature of the message exchange covered by (11) and (12) also suggests its classification as an *exchange of bus fleet information*. However, in this case the purpose of the BFO is not to get information, but to understand why the BFMS believes that the information is actually true. Thus, the capabilities attached to the BF Informer role include not only the access to relevant information, but also the enactment of knowledge to provide supporting evidence. So, in sending message (11) the BFO *requests an explanation* to understand why the BFMS believes that the delay of some line qualifies as generalised, while message (12) stands for the requested *explanation* in terms of the premises used to infer that proposition[9].

Table 5 summarises the illocutionary expressions identified for this type of social interaction. Moreover, Figure 11 depicts an UML interaction diagram, which describes the sequence of messages sent between the BFO class and the BF Informer role, where the control flow line is partitioned in terms of the three conversation units described above. Additionally, Table 5 and Figure 11 contain the results of an equivalent analysis for the two remaining types of interaction, which will be discussed in the sequel.

Bus Fleet Control. Two conversations between the BFMS and the BFO in the context of the control of the bus fleet can be identified in Table 4. The first one is initiated by message (3), is then temporally interrupted by an exchange concerning a related but different topic, and concludes with message (15). The second conversation is initiated with message (16) and finishes with (17).

In this context, the BFMS acts as a mere intermediary, transferring to the bus drivers the different instances of control actions that the BFO intends to be performed. As the transfer is expected to be accomplished *immediately* and *without discussions*, utterances (3) and (16) should be interpreted as *commands* [29, pp. 38-39]. As far as message (15) is concerned, it stands for the *cancellation* of the previous command, which is sent by the BFO after a conversation with the BFMS, in its role of BF Advisor concerning its suitability. Last, in sending the message (16) the BFMS *informs* the BFO that the transfer of the control action was successfully performed.

Bus Fleet Management Advice. Utterances ranging from (4) to (10) refer to the advice which the BFMS offer to the BFO. In uttering message (4), the BFMS reacts to the control action that the BFO intends to enact, *warning* the BFO *not to* apply that action. According to the definition given in [29, pp. 177-178], the illocutionary purpose of a warning consists of an attempt to avoid something *bad* to happen to the addressee. In our case, the "bad thing" is spelled out in the warning[10], and consists of the waste of time and resources that an ineffective control decision would imply.

Message (4) is followed by an explanatory subdialogue, consisting of messages (5) and (6), about the warning being issued. Hence, similarly to the BF Informer

[9] The knowledge used to infer that proposition is omitted in this particular utterance.
[10] Indeed, the warning is simply performed by making public that "bad thing".

role, the BF Advisor is also able to justify the content of its advice. In sending message (5) the BFO *request an explanation* to understand why the action is not effective. Message (6) stands for the corresponding *explanation*, which simply refers to the information used to infer that conclusion.

The dialogue goes on with the BFO *consulting* the BFMS, and the BFMS *recommending* two alternative control actions to solve the problem of the line. This latter message is also followed by an explanatory subdialogue initiated by the BFO in order to understand why one of the alternatives would solve the problem. It should be noted that this explanatory subdialogue, as well as the previous one, are more marginal than messages (4), (7) and (8), which can be considered to form the core of the advisory dialogue.

Table 5. Preliminary CAL of the Bus Fleet Management System ACL

Preliminary **BFMS CAL**		
BF Info. Exchange	**BF Control**	**BF Advisement**
CAL		
BF Informee	**BF Controller**	**BF Advisee**
ask (1)(13) *request an explanation(11)*	*command (3)(16)* *cancel (15)*	*request an explanation (5)(9)* *consult (7)*
BF Informer	**BF Intermediary**	**BF Advisor**
inform (2)(14) *explain (12)*	*inform (17)*	*warn not to (4)* *explain (6)(10)* *recommend (8)*

(Step 3) Reuse Analysis

This step aims at identifying the subset of the preliminary BFMS CAL that can be adequately formalised by some components of the FIPA CAL. Moreover, the reuse of some FIPA performatives will be transferred to FIPA Interaction Protocols as well. In the sequel, we will analyse the possibility of reusing the FIPA ACL for each type of social interaction.

Exchange of Bus Fleet Information. The performatives that characterise the behaviour of the BF Informer and BFO are summarised in table 5. The FIPA performatives *query-ref* and *inform-ref* are adequate formalisations of the verbs *ask* and *inform*, while *explain* and *request an explanation* have no clear counterparts in the FIPA CAL.

The lack of significant differences between the meaning of the identified performatives of the FIPA CAL and the corresponding components of the BFMS CAL, can be shown by comparing their respective definitions, as stated by the NSM components in [29] and the FIPA SL formulae in [8]. For instance, the meaning of the speech act verb *inform*, according to the definition given in [29, pp. 301-302], can be paraphrased as follows (the different components are numbered to facilitate further reference to the definition):

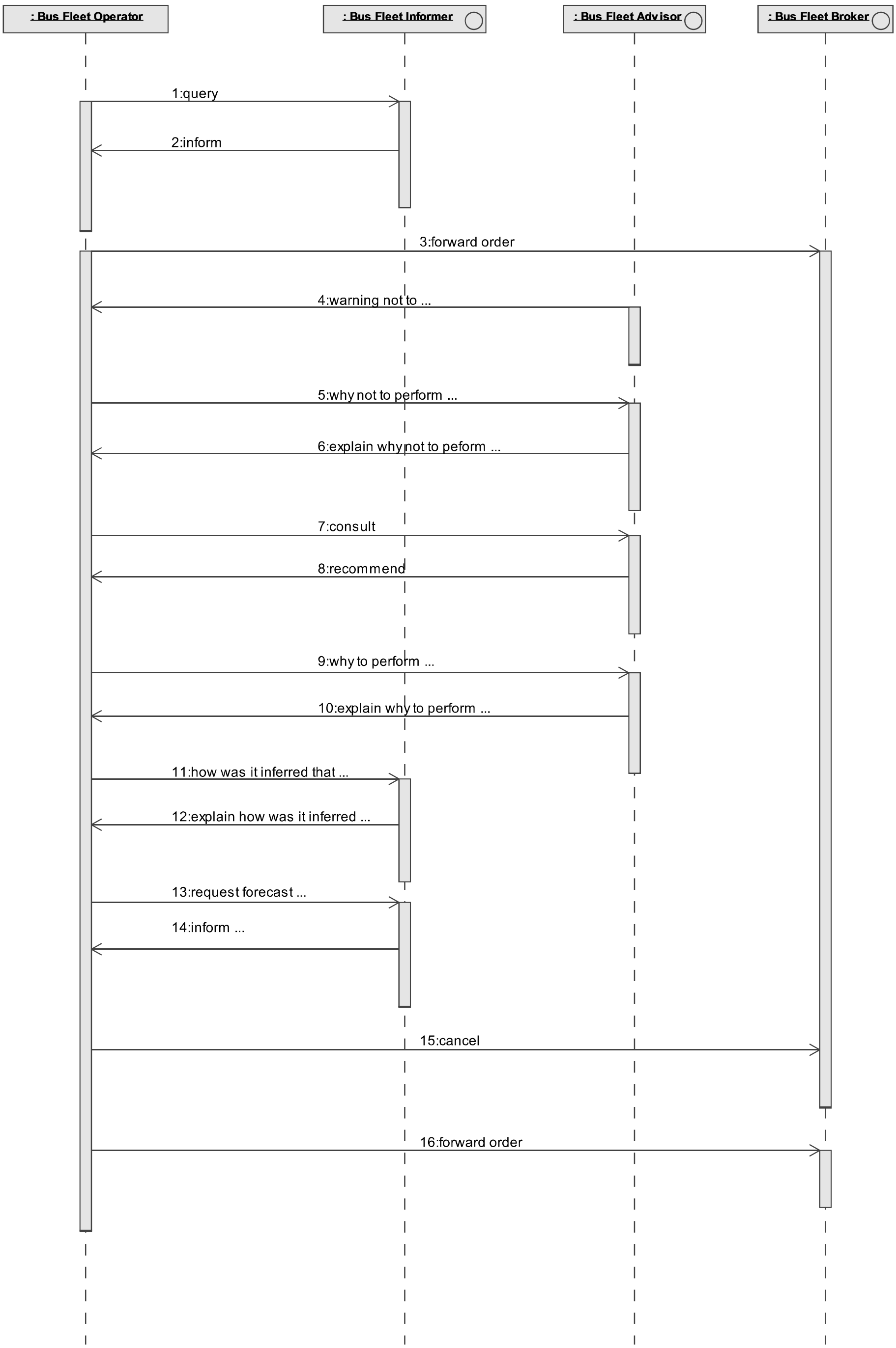

Fig. 11. First Step Towards the BFMS IPL: Segmentation of the sample dialogues

(1) I assume that you want to know things about X
(2) I know something about X that I think you should know
(3) I assume I should cause you to know it
(4) I say: (. . .)
(5) I say this because I want to cause you to know it
(6) I assume that you will understand that this is not something that could be untrue
(7) I assume that I will cause you to know it by saying this

Whereas the meaning of the FIPA performative *inform-ref*, according to the definition given in [8, p. 17], is[11]:

$$
\begin{aligned}
&<i, inform-ref(j, Ref\ x\ \delta(x))> \\
&\quad FP: Bref_i Ref\ x\ \delta(x) \wedge \\
&\qquad \neg B_i(Bref_j Ref\ x\ \delta(x) \vee Uref_j Ref\ x\ \delta(x)) \\
&\quad RE: Bref_j Ref\ x\ \delta(x)
\end{aligned}
$$

The English speech act verb *inform* basically aims at causing the addressee to know something about X (as stated in the illocutionary component, numbered 5). This fits the rational effect postulated for the FIPA *inform-ref* performative: that the receiver of the message believes the individual (or individuals) x that satisfies some property $\delta(x)$. Moreover, both the ability precondition of *inform-ref* and components (2) and (6) of *inform* make reference to the fact that the speaker believes that proposition.

Still, there is a number of assumptions implied by the verb *inform* which are not met by the definition of *inform-ref*. They refer to its *effectiveness*, expressed by means of component (7), and the implication that the message is likely to be important for the addressee [29], which is captured by the modal "should" used in components (2) and (3). The first characteristic is not particularly well suited to the case in which the sender of the message is a software agent, whereas the second one is not specially important in the context of our application. Nevertheless, both characteristics are not in contradiction with the definition of *inform-ref*, so that this performative constitutes an adequate formalisation of *inform*.

With regard to the role model of the BF Informer role, played by the BFMS, the reuse of the FIPA *inform-ref* performative has important consequences. This is due to the fact that the characteristic role of the FIPA *inform-ref* performative is the *Informer* role. Therefore, the BF Informer role could be considered either as an *specialisation* of this generic role, or as a role which *plays* the *Informer* role. As the capability of informing the BFO is essential to this role, we conceive it as a specialisation of the FIPA Informer role.

Concerning the interaction protocols for the exchange of bus fleet information, two kinds of protocols should be distinguished. The first one relates to the exchange of information, and the second one to the explanation of the communicated information. As far as the first type of protocols is concerned, the FIPA-Query-Protocol of the FIPA IPL covers the conversation included in the dialogue shown in table 4, so that it can be reused.

[11] The meaning of the $Bref_i$ operator is established by means of the following equivalence: $Bref_i Ref\ x\ \delta x \equiv (\exists y) B_i(Ref\ x\ \delta x = y)$.

Bus Fleet Control. The BF Intermediary role of the BFO shows interesting similarities to the FIPA Broker role identified in section 2. The behaviour of the latter is characterised by the reception of *proxy* and *propagate* messages, denoting requests to forward particular communicative acts to third party agents. This is essentially what the BF Controller requires from the BF Intermediary, being the third party agents some of the bus drivers.

Taking into account the meaning of the *proxy* and *propagate* performatives, as the BFO does not require recursive propagation of the control actions being ordered to the bus drivers, the *proxy* performative seems to be the most adequate of both. On the other hand, there is a number of assumptions of the English speech act verb *command* which are not shared by the FIPA *proxy* performative. For instance, the *proxy* message only attempts to make public the "wanting" of the sender to cause the receiver to perform some communicative action, whereas the *command* verb has additional implications of authority and power. Still, these relationships are public in the BMFS domain, so the reduced expressiveness of the *proxy* performative with respect to *command* act is not overly important. In addition to the *command* speech act, the BF Controller also issues *cancellations* of previous commands. The FIPA *cancel* performative captures adequately the use of the verb in the context of dialogue 4. The FIPA *inform* performative can be also used to formalise the illocutionary action of *informing* that a *proxy* was successfully performed. Thus, the BF Controller and the BF Intermediary can be considered specialisations of the FIPA Brokering Requester and FIPA Broker roles of figure 4, respectively.

Concerning the protocols for the interaction between the BF Controller and the BF Intermediary, they should give account of the two message exchanges identified in step 2. The first conversation consists of a *proxy* message, followed by a *cancellation*. The second one starts with a *proxy* and is finished with an *inform-if*. Neither of these conversations is fully covered by the FIPA-Brokering-Protocol of the FIPA IPL. First, both of them lack the *agreement* of the FIPA Broker to perform the transfer. Second, the FIPA-Brokering-Protocol does not consider the possibility of cancelling the proxy message.

Bus Fleet Management Advice. Table 5 summarises the speech act expressions which characterise the behaviour of the BF Advisor and BF Advisee roles, identified in step 2. Concerning the expression *request an explanation*, it basically consists of a FIPA *request* that the receiver provides a justification. The other illocutionary expressions do not have direct counterparts in the FIPA CAL.

(Step 4) Extension Design

The formalisation of the IPs and CAs which were not adequately covered by the FIPA ACL, is the subject of this step of the design process. First, the organisational model will be refined according to the advisory and explanatory communicative actions not included in the FIPA CAL. Subsequently, we will formalise these CAs and their corresponding protocols.

(Step 4.1) Refinement of the Organisational Model

This step aims at the identification of the characteristic generic roles and types of interaction corresponding to the illocutionary expressions not covered by the FIPA CAL. Two groups of expressions can be identified:

- *warnings* and *recommendations* performed by the BF Advisor are types of CAs which are likely to be performed by many other types of agents (e.g. the reviewer role of section 2. This suggests the identification of a generic role *Advisor*. Correspondingly, the BF Advisee is the characteristic role of the *consult* illocutionary act. Both types of agents participate in *Advisory Interactions*.
- *explanations* are performed by the BF Informer role, as well as by the BF Advisor. The type of agent which will typically perform explanations is termed an *Explainer*. This role participates with the *Explainee* in the context of *Explanatory Interactions*. The Explainee requests the explainer to clarify why some proposition is true.

Figure 12 shows the refinement of the role model for the BFMS class, which takes into account the *Explainer* and *Advisor* generic roles identified above. It also shows the refinement of the role model due to the reuse of some components of the FIPA CAL, as discussed in Step 3. A dashed line separates those roles and classes which are domain-dependent, from potentially reusable ones: the FIPA Informer and Broker roles, the Explainer and the Advisor. Abstract FIPA roles identified are represented in light gray, while the new explanatory and advisory abstract roles are indicated by a dark gray background.

Just as the BF Informer is considered a specialisation of the FIPA Informer role, the BF Advisor is modelled as a specialisation of the generic Advisor role. Concerning the relationship between the BF Advisor and the Explainer, there are two possibilities: either the BF Advisor *plays* that role, or it is the Advisor role which actually does. We adopt the latter alternative, as any kind of advisor needs explanatory capabilities to reach its goals[12]. Note that this relationship between the explainer and the advisor is not of a generalisation/specialisation nature, as the advisor will only sometimes be an explainer (not during its whole life-cycle).

(Step 4.2) Bus Fleet Management System CAL

Steps 3 and 4.1 showed that the initial set of domain-dependent IPs can be further subdivided into five more generic ones, attending to their characteristic types of CAs. Among these abstract IPs, the explanatory and advisory interactions are not adequately supported by the FIPA CAL. This step of the design process attempts to identify the content of the Advisory and Explanatory CALs.

[12] On the contrary, although our BF Informer needs explanatory capabilities, this is not the case for *any* kind of Informer. Hence, the link between the Explainer and the BF Informer is established directly.

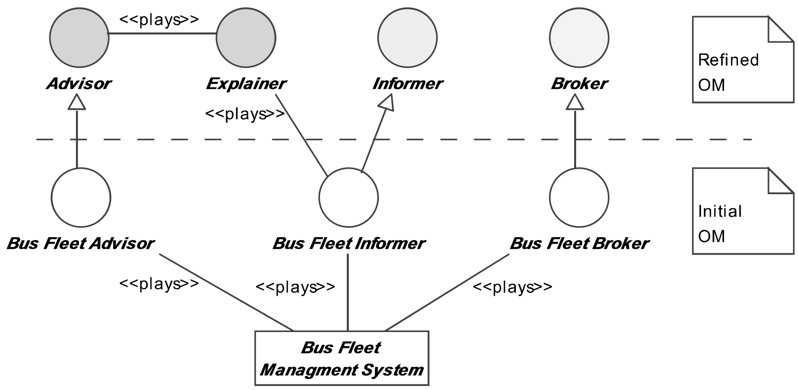

Fig. 12. Refined Role Model of the Bus Fleet Management System Class

The Advisory CAL. The identification of a set of performatives characteristic of advisory interactions departs from the already mentioned natural language performatives *warn not to* and *recommend*, for the *Advisor* role, and the *consult* performative for the *Advisee* role. The goal of this section is to formalise these performatives, as well as to potentially identify additional elements for a more comprehensive *Advisory CAL* that is suitable for the use of software agents. The analysis will draw on the results of [24] complemented by a conceptual analysis of the advisory domain, in order to justify the identification of the Advisory CAL components.

The Advisory Domain. Figure 13 shows a UML class diagram with the major concepts of the advisory domain. On the one hand, the *advisee* is situated in an *environment* where some *events* happen, that give rise to certain *facts*. Some of these events may well be certain *actions* of the advisee, which are intentionally performed in order to achieve some *goals*. These, in turn, are characterised by a subset of the preferences of the agent[13], which are linked to some "chosen" facts.

Setting out from this model, we define an *Advisor* as the type of agent which takes care of the *satisfaction* and *maintenance* of the preferences of some other agent (the *Advisee*)[14]. This general definition will become more specific as the communicative actions for this kind of agent are identified. In the following, a number of illocutionary actions are identified, by considering different contextual situations associated to the goals and the events of the environment:

[13] This generalisation relationship between *preferences* and *goals* represents the so-called "realism constraint" property [3].

[14] Therefore, we assume that the goals and preferences of the advisee are perfectly known by the advisor. This suggests the existence of a *profiling* agent, a mandatory role to be played by any advisor. The characterisation of this role is subject to future research.

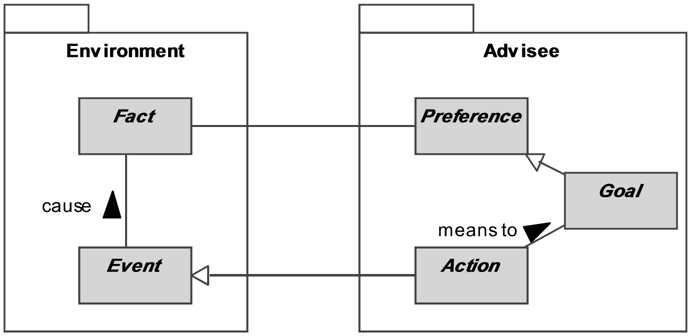

Fig. 13. Conceptual Model of Advisory Interactions

- The advisee is committed to some goal, and does not know which action is the most suitable to achieve it. In this context, the advisee may perform a *consultation* to the advisor.
- The advisor acknowledges this mental state of the advisee (either based on a consultation or proactively). Then, it may *suggest* or *recommend* some action to the advisee.
- The advisor foresees the possible future occurrence of some events which affect negatively some preference of the user, and issues a *warning* (note that the future event may be some action of the advisee).
- The advisor acknowledges the occurrence of some event that enables the achievement of a goal of the advisee. The advisee is *notified* accordingly.

The formal definitions of these illocutionary acts will be given in terms of both, the NSM and FIPA SL formalisms. The FIPA SL language is obviously more restricted than the Natural Semantic Metalanguage (both syntactically and semantically), but as our artificial illocutionary actions are simpler than their natural language counterparts, a proper formalisation is still possible. Also, note that we will make use of some modal operators that are not part of the FIPA SL specification. These include the choice operator C_i (which stands for the preferences of some agent i), and the temporal modalities $Henceforth$ and *Possible*. Although the choice modality is described in the FIPA ACL semantic framework as being part of the SL language [8], the FIPA SL specification [9] does not capture it (most probably because it is not needed to define the semantics of any FIPA CAs). The temporal modalities are useful abbreviations introduced by Sadek [20] with the following semantics:

$$\begin{aligned} Possible(p) &\equiv \exists e Feasible(e, p) \\ Henceforth(p) &\equiv \neg Possible(\neg p) \end{aligned}$$

In the sequel, some examples for the formalisation of advisory CAs will be provided, which complement those found in [24].

Warn. The speech act verb *warn* is used in the context of our sample dialogue in the syntactic pattern *warn not to*, but there are many other frames in which this verb can be used as well: *warn that*, *warn against*, *warn to*, etc. . A semantic characterisation of the natural language verb *warn* attempts to capture all these possible uses, hence it is not quite explicit with respect to the dictum [29, p. 177]:

Definition 2 *Warn (English CAL)*

I think you might do something that would cause something bad to happen to you
I say: (. . .)
I say this because I want to cause you to be able to cause that bad thing not to happen to you

Our proposal for a performative to be used by an artificial agent, similar to the English illocutionary verb *warn not to*, focuses on the "bad thing", *X*, expected to happen, and its possible cause: some event *Y*. This leads to the following definition:

Definition 3 *Warn (Advisory CAL)*

I think Y could be done
I think X is caused by Y
I think of X as something that is bad for you
I say: Y will cause to happen something bad (X) to you
I say this because I want to cause you to know that X could happen

Definition 3 indicates that the propositional content of a warning consists of some state of affairs X and some event sequence Y. The speaker expresses her belief that it is possible that Y be done in the future, and that this event will cause X to happen. Moreover, the speaker expresses her belief that X is a bad thing to happen for the hearer. This is obviously related to preferences, and so to the choice operator. A possible formalisation is: $B_i C_j \neg X$. The formal model of the *warn* performative follows:

$$
\begin{aligned}
&<i, warn(j, Y, X)> \\
&\quad FP: B_i Possible(Done(Y)) \wedge \\
&\qquad B_i Henceforth(Done(Y) \Rightarrow X) \wedge \\
&\qquad B_i C_j \neg X \\
&\quad RE: B_j Possible(X)
\end{aligned}
$$

Notice. Parallel to the definition of the performative *warn*, we may consider a communicative action by which the advisor aims at informing the addressee of the possibility of achieving some goal X, due to the some positive event Y, which has just happened in the environment. This type of CA differs from a warning in two major points: first, event Y is not expected to happen in the future, but has just happened instead; second, the foreseeable consequence X of event Y is actually something positive for the addressee, i.e. it goes in line with its preferences.

In summary, event Y is considered to enable the achievement of some goal X of the addressee. The English catalogue of speech act verbs does not include any specific illocution to denote this type of CA. The label *notice* attempts to capture the fact that event Y is something that can be seen as "hot news", which the advisee should know [29, p. 304]. Nevertheless, the NSM and FIPA SL definitions outline the differences with respect to the meaning of *notify* as stated in [29, pp. 304-305]:

Definition 4 *Notice (Advisory CAL)*

I think Y has been done
I think X could be caused by Y
I think of X as something that is good for you
I say: Y could cause to happen something good (X) to you
I say this because I want to cause you to know that X could happen

The formalisation of this NSM definition follows the pattern of the *warn* performative. As before, Y stands for some event and X for some proposition:

$$
\begin{aligned}
&{<}i, notice(j, Y, X){>}\\
&\quad FP: B_i Done(Y) \wedge\\
&\qquad B_i Henceforth(Done(Y) \Rightarrow Possible(X)) \wedge\\
&\qquad B_i C_j X\\
&\quad RE: B_j Possible(X)
\end{aligned}
$$

Consult. Definitions of the English speech act verb *consult* emphasise the reciprocal character of this verb, in the sense that an extended verbal exchange, rather than a single short answer is expected [29]. The addressee need not be an expert on the matter, but just someone related to the speaker's decision. In the following definition, "causing X" can be conceived as a general case of "achieving a goal X":

Definition 5 *Consult (Advisory CAL)*

I don't know what to do (Y), that will cause X
I say: I want you to say what I can do, that will cause X
I say this because I want to cause you to know that I want to know what to do

The formalisation of this performative follows the definition of the FIPA *cfp* performative, which is defined as a *query-ref* about the condition(s) under which the receiver would perform some action if the sender of the message wanted her to. This amounts to ask for the conditions under which the receiver would perform a *propose* message [8]. Equivalently, the *consult* performative can be defined as a *query-ref* about the kind of actions Y which the receiver agent may *recommend* to the sender in order to achieve some goal X [24]):

$$
\begin{aligned}
&{<}i, consult(j, X){>} \equiv\\
&\quad {<}i, query-ref(j, Ref\ Y\ (Possible(Done({<}j, Y{>})) \wedge\\
&\qquad\qquad Henceforth(Done({<}j, Y{>}) \Rightarrow Possible(X)))){>}
\end{aligned}
$$

The Explanatory CAL. In this section, a catalogue of performatives to support explanatory interactions will be designed. Similarly to the advisory catalogue, we first analyse the major concepts underlying the explanatory domain. Second, the formal models of the identified performatives will be provided.

The Explanatory Domain. According to the proposed definition of the speech act verb *explain* in [29, pp. 295-297], to explain something roughly means to cause someone *understand* something. For instance, the thing to be understood could be the reason *why* something is the case, *what* is something or *how* to do something. In the context of our application, explanations of the beliefs of some agent with respect to certain propositions are required (e.g. the belief of the BFMS that some action is instrumental with respect to the solution of the generalised delay problem in some line). In general, the *explainer* will typically be a software agent, and the *explainee* the human user of the application.

In order to make understandable the belief in some proposition X, the software agent will draw on the way in which that information has been obtained. There are two possibilities: either the agent came to believe X on the basis of some *inference* process, or it was *informed* that X is true. In the first case, the explanation will rest on the premises and knowledge used in the reasoning process. In the second, the explanation will be backed by the corresponding source of information. Figure 14 summarises the previous analysis.

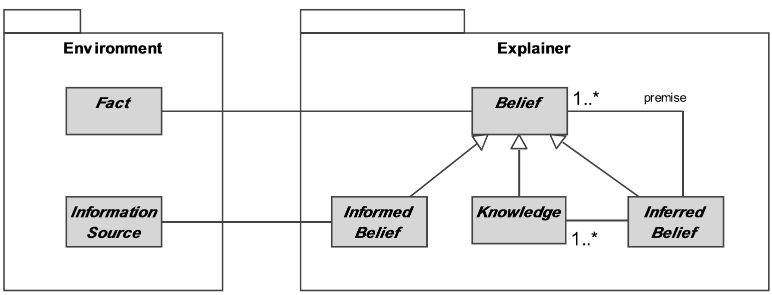

Fig. 14. Conceptual Model of Explanatory Interactions

Accordingly, we can identify two types of explanatory CAs for the explainer role: *explanation of inferred propositions*, and *explanation of observed propositions*. The explainee is characterised by *requesting an explanation*. In the sequel these CAs will be formalised.

Explain Inferred Proposition. The *explanation of some inferred proposition* act will draw on the thing to be explained, i.e. the conclusion X, as well as the premises Y and knowledge Z used in the inference process. The illocutionary purpose of this kind of communicative act is simply to cause the addressee to know the way in which the conclusion was inferred. Thus, the formalisation can be performed by simply overloading the propositional content of the FIPA *inform* CA.

Our formalisation of this definition is based on an extension of the FIPA SL language with the modal operator $Inferred_i(X, Y, Z)$, which represents the attitude of some agent i towards some proposition X, which has been inferred on the basis of premises Y and knowledge Z[15].

$$<i, explain - inferred - fact(j, X, Y, Z)> \equiv$$
$$<i, inform(j, Inferred_i(X, Y, Z))>$$

Explain Observed Proposition. The formalisation of the illocutionary act *to explain some observed proposition*, can be performed on the basis of the *inform* performative, as well. In this case, the formalisation requires the introduction of the modal operator $Observed_i$, which models the attitude of some agent i towards a proposition that has been obtained from some information source (it might be some other agent j, in case that agent i was informed by j). Hence, the model for this performative derives from the definition of *inform* in the following way:

$$<i, explain - observed - fact(j, X, Y)> \equiv$$
$$<i, inform(j, Observed_i(X, Y))>$$

Request for explanation. The explainee may perform a request for explanation for some proposition, whenever it does not understand the reasons for it. Similarly to the *query-if* and *query-ref* performatives, the formalisation draws on the *request* performative and the action disjunction operator.

$$<i, request - for - explanation(j, X)> \equiv$$
$$<i, request(j, <j, explain - inferred - fact(i, X, Y, Z)> \mid$$
$$<j, explain - observed - fact(i, X, Y)>)>$$

(Step 4.3) Bus Fleet Management System IPL

In step 2 of the design process of the BFMS ACL, we have argued that some FIPA Interaction Protocols can be reused in the context of the exchange of bus fleet information and bus fleet control interactions. However, the interaction between the BFMS and BFO concerning bus fleet advice and explanatory dialogues require the design of new protocols. This final substep of the ACL design process attempts to develop the interaction protocols for these generic types of interaction, on the basis of the performatives included in the Advisory and Explanatory CALs identified in the last substep. These protocols are the basis for a more comprehensive Advisory and Explanatory IPLs. The following content is based upon results in [25].

[15] Hence, the following is an axiom: $\models Inferred_i(X, Y, Z) \Rightarrow B_i(X \wedge Y \wedge Z)$.

The Advisory IPL. Advisory interactions might be initiated both by the advisor and the advisee. The former may *warn* the user *not to* perform some action she intends to enact. She may also *notify* the user of some event that enables the achievement of one of her goals, or proactively issue a *recommendation* or *suggestion*. On the other hand, the advisee may start an advisory interaction by *consulting* the advisor about the best way to achieve some goal p. In this latter case, the interaction enters a state parameterised by the goal to be achieved by the user. The interaction may enter this state from a warning or a notification as well. In case of a warning, the user may consult the advisor with respect to the best way of achieving goal p', which was the reason to consider action a. In case of a notification, the user may just consult the advisor regarding the best way to achieve the notified fact. The consultation may be answered with a suggestion or a recommendation of the advisor, to perform some action. This can cause further consultations about how to perform that action.

Figure 15 shows a UML state diagram for the advisory interaction protocol. The four transitions from the initial pseudo-state depict the four alternatives ways of starting the interaction, as commented previously.

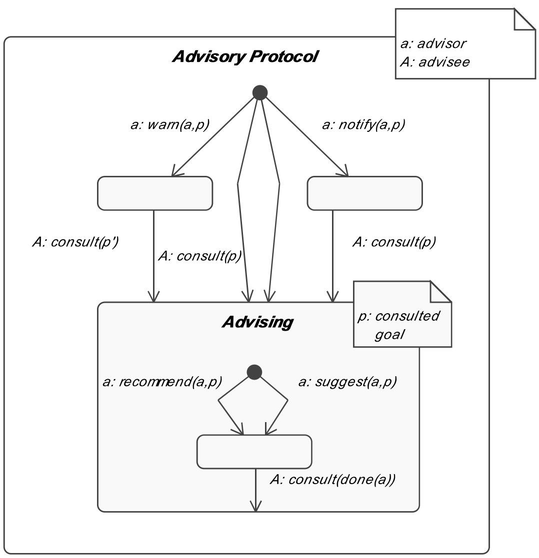

Fig. 15. Conceptual Model of Advisory Interactions

The Explanatory IPL. Explanatory interactions are parameterised with respect to some proposition to be explained, and start with a *request-for-explanation* message. The explainer may have come to know that proposition either by: (1) directly observing that fact from some observation source (*explain-observed-fact* message), or (2) by inferring this fact from a set of premises $p_1, p_2, ...$ and a knowledge base consisting of the set of propositions $k_1, k_2, ...$

(*explain-inferred-fact* message). In this latter case, the explainee may request further explanation of some premise p_i. Figure 16 shows the UML state diagram for the advisory interaction protocol.

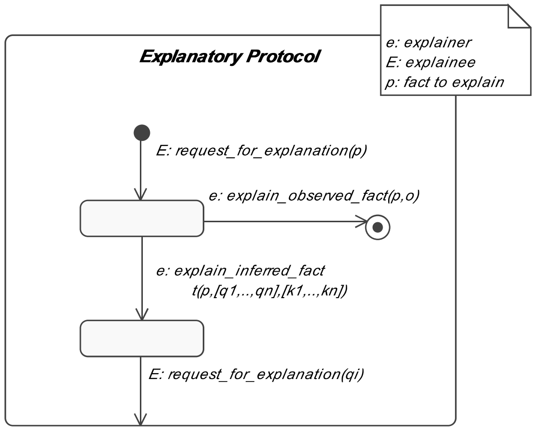

Fig. 16. Conceptual Model of Advisory Interactions

4.3 Resulting ACL for the Bus Fleet Management System

Table 6 summarises the result of our ACL design example. The content of the BFMS ACL consists of two major parts: the BFMS CAL and BFMS IPL. It is structured according to the supported generic types of interaction as identified in figure 12. Moreover, it also shows the relationship between these generic types of interaction and the domain-dependent types of interaction identified in the initial organisational model (see figure 10). Observe that the ACL reuses part of the FIPA ACL performatives and protocols, and that it contains two new generic types of interaction, contributing potentially reusable libraries of communicative actions and protocols for other multiagent domains.

5 Discussion

In this paper we have described a method for the principled design of domain extensions to the general-purpose FIPA ACL, based on an organisational structuring. Our approach follows the work by Pitt and Mamdani [16] with respect to the general design strategy, using a core language and providing extended components as required by new types of social interaction. However, it differs significantly in that we explicitly establish and follow the link between the ACL and the types of interaction specified in the organisational model, thus providing clues as to how these types of interaction might inform the design of the ACL. The RICA metamodel also draws upon other methodologies found in the

Table 6. Agent Communication Language for the Bus Fleet Management System

BFMS ACL				
BF Info. Exchange		**BF Control**	**BF Advisement**	
FIPA Info. Exchange	**Explanatory Interaction**	**FIPA Brokering**	**Advisory Interaction**	**Explanatory Interaction**
CAL				
Informee	**Explainee**	**Brokering Requester**	**Advisee**	**Explainee**
query	*request-for-explanation*	*proxy* *cancel*	*consult*	*request-for-explanation*
Informer	**Explainer**	**Broker**	**Advisor**	**Explainer**
inform-ref	*explain-inferred-fact* *explain-observed-fact*		*warn* *notify* *recommend*	*explain-inferred-fact* *explain-observed-fact*
IPL				
FIPA-Query-P.	*Explanatory-P.*	*FIPA-Brokering-P.*	*Advisory-P.*	*Explanatory-P.*

literature, mainly [14]. As with [2] and [5], it focuses on the relation between organisational and ACL components. However, our approach is especially concerned with the reusability and interoperability of CAs and IPs.

The present design method constitutes a conservative extension of the FIPA CAL and IPL, as it does not significantly endanger interoperability. On the one hand, the organisational structuring of the libraries helps the designer to better understand their contents; on the other, formal definitions of new performatives (e.g. based on FIPA SL and NSM) help to enforce a unique interpretation. In addition, our proposal fosters reusability of the extensions, and shows how the reuse of ACL performatives and protocols implies the reuse of the corresponding characteristic components of the organisational model: generic types of social interactions and roles.

We are aware that our design method has been illustrated only in the context of a closed environment (the centrally designed bus fleet management application). However, we believe that it also scales up well to open environments. In this case, an agreement between designers or a standardisation body such as FIPA is required in order to identify specific extensions to performative and protocol catalogues for particular organisations, as suggested by our structuring approach. In future work we will examine as to how far the organisational analysis of ACLs may have an impact on the current FIPA architecture.

References

1. J.L Austin. *How to do Things with Words.* Clarendom Press, Oxford, 1962.
2. Jeffrey M. Bradshaw, Stewart Dutfield, Pete Benoit, and John D. Woolley. KAoS: Toward an industrial-strength open agent architecture. In Jeffrey M. Bradshaw, editor, *Software Agents*, chapter 17, pages 375–418. AAAI Press / The MIT Press, 1997.
3. Philip R. Cohen and Hector J. Levesque. Intention is choice with commitment. *Artificial Intelligence, AI*, 42(2–3):213–261, March 1990.
4. Frank Dignum, Barbara Dunin-Keplicz, and Rineke Verbrugge. Creating collective intention through dialogue. *J. of the IGPL*, 9(2):305–319, 2001.

5. M. Esteva, J. A. Rodriguez, C. Sierra, P. Garcia, and J. L. Arcos. On the formal specifications of electronic institutions. In F. Dignum and C. Sierra, editors, *Agent-mediated Electronic Commerce (The European AgentLink Perspective)*, volume 1191 of *LNAI*, pages 126–147, Berlin, 2001. Springer.
6. J. Ferber and O. Gutknetch. A meta-model for the analysis of organizations in multi-agent systems. In Y. Demazeau, editor, *ICMAS'98*, pages 128–135. IEEE Press, 1998.
7. Jacques Ferber, Olivier Gutknecht, Catholijn M. Jonker, Jean-Pierre Müller, and Jan Treur. Organization models and behavioural requirements specification for multi-agent systems. In *Proc. of the ECAI 2000 Workshop on Modelling Artificial Societies and Hybrid Organizations*, 2000.
8. Foundation for Intelligent Physical Agents. *FIPA Communicative Act Library Specification.* http://www.fipa.org/specs/fipa00037, 2000.
9. Foundation for Intelligent Physical Agents. *FIPA SL Content Language Specification.* http://www.fipa.org/specs/fipa00008/XC00008D.html, 2000.
10. Georg Gottlob, Michael Schrefl, and Brigitte Röck. Extending object-oriented systems with roles. *ACM Transactions on Information Systems*, 14:268–296, 1996.
11. J. Hernandez, S. Ossowski, and A. Garcia-Serrano. Multiagent architectures for intelligent traffic management systems. *Transportation Reaserch (Part C)*, 10, 2002.
12. Kazuhiro Kuwabara, Toru Ishida, and Nobuyasu Osato. Agentalk: Coordination protocol description for multiagent systems. In Victor Lesser, editor, *Proceedings of the First International Conference on Multi–Agent Systems*, page 455, San Francisco, CA, 1995. MIT Press.
13. S. Ossowski and J. M. Serrano. Towards agent-based architectures for advanced decision support. In *Proceedings of the Second Iberian Workshop on Physical Agents (WAF)*, 2001.
14. H. Van Dyke Parunak and J. Odell. Representing social structures in UML. In J. P. Müller, E. Andre, S. Sen, and C. Frasson, editors, *Proceedings of the Fifth International Conference on Autonomous Agents*, pages 100–101, Montreal, Canada, May 2001. ACM Press.
15. B. Pernici. Objects with roles. In *Proc. of the Conference on Office Information Systems*, pages 205–215, Cambridge, MA, 1990.
16. J. Pitt and A. Mamdani. Designing agent communication languages for multi-agent systems. *Lecture Notes in Computer Science*, 1647:102–114, 1999.
17. James Rumbaugh, Ivar Jacobson, and Grady Booch. *The Unified Modeling Language Reference Manual.* Addison-Wesley, 1999.
18. M. D. Sadek. Dialogue acts are rational plans. In *Proceedings of the ESCA/ETRW Workshop on the structure of multimodal dialogue*, pages 1–29, Maratea, Italy, 1991.
19. M. D. Sadek. A study in the logic of intention. In William Nebel, Bernhard; Rich, Charles; Swartout, editor, *Proceedings of the 3rd International Conference on Principles of Knowledge Representation and Reasoning*, pages 462–476, Cambridge, MA, October 1992. Morgan Kaufmann.
20. M. D. Sadek. A study in the logic of intention. In William Nebel, Bernhard; Rich, Charles; Swartout, editor, *Proceedings of the 3rd International Conference on Principles of Knowledge Representation and Reasoning*, pages 462–476, Cambridge, MA, October 1992. Morgan Kaufmann.
21. J.R Searle. *Speech Acts.* Cambridge University Press, 1969.
22. J.R. Searle and D. Vanderveken. *Foundations of illocutionary logic.* Cambridge University Press, 1985.

23. J. M. Serrano and S. Ossowski. Domain extensions to the FIPA-ACL: an application to decision support systems. In *SEID-2000*, pages 1–14. University of Vigo, 2000.
24. J. M. Serrano and S. Ossowski. The design of communicative act libraries: a linguistic perspective. *Applied Artificial Intelligence*, In Press, 2002.
25. J. M. Serrano and S. Ossowski. The design of interaction protocols: an organizational approach. In *Proceedings of the Agent Communication Language and Conversation Policies Workshop*, 2002.
26. C. Sierra, N. R. Jennings, P. Noriega, and S. Parsons. A framework for argumentation-based negotiation. *LNCS*, 1365:177–193, 1998.
27. Teun A. van Dijk. *Discourse as Social Interaction*. SAGE Publications, 1995.
28. A. Wierzbicka. A semantic metalanguage for a crosscultural comparison of speech acts and speech genres. *Language and Society*, 14:491–513, 1985.
29. A. Wierzbicka. *English speech act verbs. A semantic dictionary*. Academic Press, Australia, 1987.
30. Ludwig Wittgenstein. *Philosophical Investigations*. Macmillan, London, 1953.
31. Michael Wooldridge, Nicholas R. Jennings, and David Kinny. The gaia methodology for agent-oriented analysis and design. *Autonomous Agents and Multi-Agent Systems*, 3(3):285–312, September 2000.
32. Franco Zambonelli, Nicholas R. Jennings, and Michael Wooldridge. Organizational abstractions for the analysis and design of multi-agent systems. In Paolo Ciancarini and Michael J. Wooldridge, editors, *AOSE*, volume 1957 of *LNCS*, pages 235–252. Springer, 2000.

Author Index