An agent framework for supporting the *MIKS* integration process

Domenico Beneventano, Sonia Bergamaschi[*,†],
Gionata Gelati, Francesco Guerra, Maurizio Vincini[†]

[*] CSITE-CNR, viale Risorgimento 2, 40136 Bologna, Italy
Email: beneventano, bergamaschi@unimo.it

[†]Dipartimento Ingegneria dell'Informazione, via Vignolese 905, 41100 Modena Italy
Email: gelati, guerra, vincini@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, the roles they play in enhancing the feature of the *MOMIS* infrastructure. *MOMIS* (Mediator agent for Integration of Knowledge 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).

In this paper we describe the new agent-based framework concerning the integration process as implemented in *MIKS* (Mediator agent for Integration of Knowledge Sources) system.

I. 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 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 [2] 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 [24]. 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* [4], [7] (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. in the following we describe the new agent-based framework called /miks/. This proposal has been implemented within the *MIKS* (Mediator agent for Integration of Knowledge Sources) system. This paper describes the roles of the agents as far as the integration process is concerned.

Mobile agents can significantly improve the design and the development of Internet applications thanks to their characteristics. The agency feature [18] 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 [19] 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 [10].

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 [12] 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 [6]. Middleware systems dealing in some way with a set of data sources commonly fall back on wrapper components or simply wrappers [26]. 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 II presents the key steps of the *MIKS* integration process. Section III introduces the *MIKS* system, illustrating the role of the agents in a framework supporting information access and integration. Section IV describes how the agents interact. Section V discusses the related work in the area of intelligent information agents.

II. 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.

III. The role of the agents in the *MIKS* integration process

In large-scale corporate information scenarios, the search for data is usually extended to a large and distributed amount of data, heterogeneity related to data structures and semantics is substantial, the number of users can be 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 [8] and multi-agent systems *(MASs)*[31]. The agent paradigm [28] 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 [20]. 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 [5]. Related work on the use of agents in information management systems (such as the Infosleuth project) is reported in section V.

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

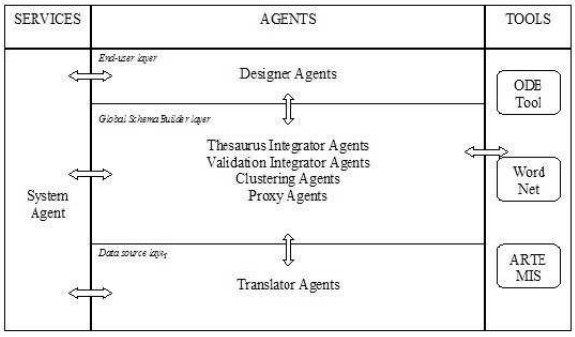

Fig. 1. Classification of agents for supporting the integration process

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 [20], 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 systems are meant to be interoperable.

In the following, we concentrate our attention on the part of the agent framework designed for assisting the integration process. For a complete description including the multi-agent system that supports *MIKS* querying phase we refer to [21].

A. The Multi-Agent System

The MAS we have designed for integration purposes includes agents that support all of the four phases we have presented in section II. Agents carry out activities for manipulating data to create information at higher levels of abstraction. Figure 1 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 1, 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 [22], a FIPA-compliant development tool [15].

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)
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. Some interesting proposals facing the problem of querying information sources which provide different levels of query capabilities have been introduced in [13], [17], [27]. The aim is pruning efficiently out information sources that are unsuitable for a given query and generating executable query plans.

A key feature of *TAs* is that they 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 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 agentifiyng 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 [3], [4]. 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).

IV. Agent interaction patterns

As we have seen, diverse types of agents populate the MAS intended to support the integration process 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 2 and 3 show the two extreme configurations the system can operate:

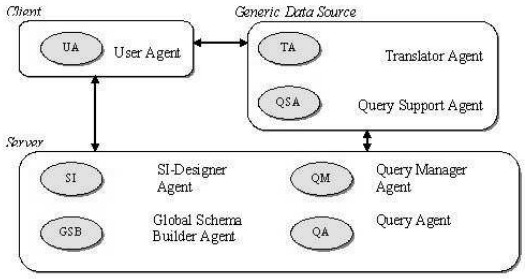

Fig. 2. The *MIKS* agents organised into a client/server configuration

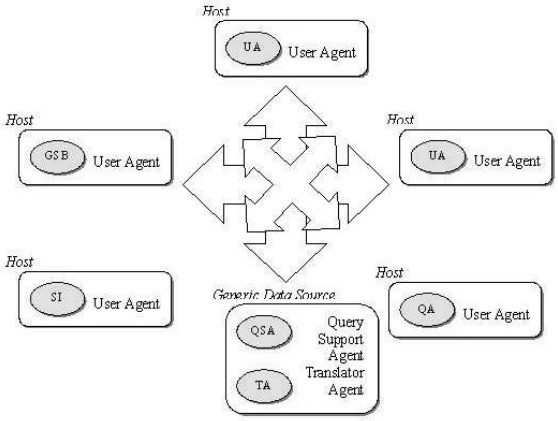

Fig. 3. The *MIKS* agents organised into a fully distributed configuration

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.

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 (Fig. 4), 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. 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 ini-

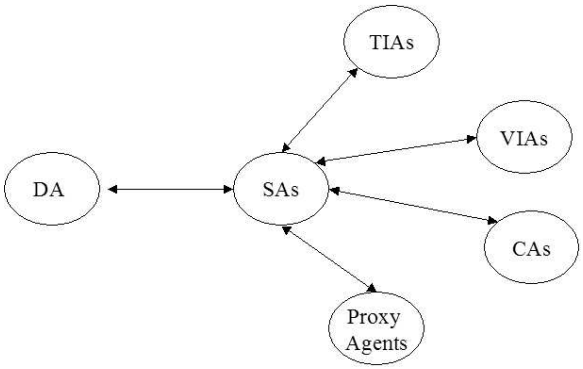

Fig. 4. System initialisation

tialisation 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 5 show the complete set of interactions. Notice the *User Agents* can request and obtain from *Proxy Agents* the *Global Virtual View*.

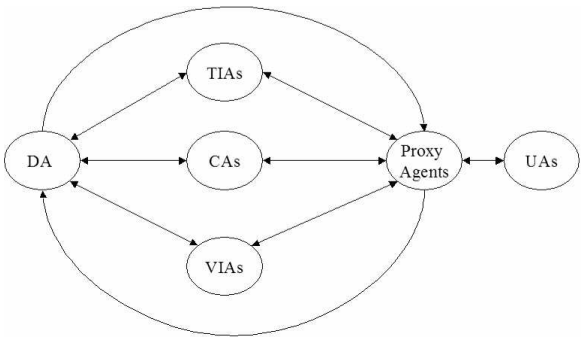

Fig. 5. A possible interaction pattern for the integration process

A. Coordination issues

As we have seen, there are a number of interactions that take place among agents. We can have interactions among agents (a proxy agent communicating with a translator agent) or direct interactions between an agent and a data source (a translator agent accessing the resources of a data source). These kinds of interactions are needed not only to exchange data to be delivered to the mediator system or to the user, but also to enable coordination among agents. A few alternatives for coordination models are proposed in [9].

As for the integration process, the key coordination issue concerns how to manage the collection of information coming from different sources. Given a knowledge domain, it is not unfrequent the case in which data sources store similar or overlapping data. In order to efficiently handle the gathering of information, coordination has to take place among agents. The Proxy Agents and the Translator Agents play an essential role in the process of coordinating the other agents actions. These agents provide auxiliary information like the visited sites, source descriptions, overlapping information, source their data come from and so on. Further, Translator Agents act as access points to the underlying resources of the data sources, helping enforce security policies for accessing resources at local hosts. This way, Proxy Agents and Translator Agents provide similar functions to those of reactive tuple spaces proposed in [9].

V. Related Work

In the area of heterogeneous information integration, many projects based on a mediator architecture have been developed. The mediator-based TSIMMIS project [14] 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. Differently 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 [11] 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 [1] 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 [23] 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 [16] provides integrated access to multiple distributed heterogeneous 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 [25] 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 architecture of the two systems is quite different. Agents with common functionalities (translator agents/query support agents and resource agents, user agents, query agents) are still observable even though they reflect the two distinct approaches. One last remark concerns the presence in Infosleuth of service agents (in particular broker agents). The *MIKS* provide the same services in a centralised manner with the *SA*.

Another experience is the RETSINA multi-agent infrastructure for in-context information retrieval [29]. In particular the LARKS description language [30] 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] Y. Arens, C. A. Knoblock, and C. Hsu. Query processing in the sims information mediator. *Advanced Planning Technology*, 1996.

[2] C. Batini, M. Lenzerini, and S. B. Navathe. A comparative analysis of methodologies for database schema integration. *ACM Computing Surveys*, 18(4):323–364, 1986.

[3] I. Benetti, D. Beneventano, S. Bergamaschi, A. Corni, F. Guerra, and G. Malvezzi. Si-designer: a tool for intelligent integration of information. *Int. Conference on System Sciences (HICSS2001)*, 2001.

[4] D. Beneventano, S. Bergamaschi, F. Guerra, and M. Vincini. The momis approach to information integration. In *Conference on Enterprise Information Systems (ICEIS01), Setbal, Portugal*, 2001.

[5] S. Bergamaschi, G. Cabri, F. Guerra, L. Leonardi, M. Vincini, and F. Zambonelli. Supporting information integration with autonomous agents. In *CIA*, pages 88–99, 2001.

[6] S. Bergamaschi, G. Cabri, F. Guerra, L. Leonardi, M. Vincini, and F. Zambonelli. Exploiting agents to support information integration. *International Journal on Cooperative Information Systems*, 11(3), 2002.

[7] S. Bergamaschi, S. Castano, D. Beneventano, and M. Vincini. Semantic integration of heterogenous information sources. *Journal of Data and Knowledge Engineering*, 36(3):215–249, 2001.

[8] J. (eds.) Bradshaw. *Handbook of Agent Technology*. AAAI/MIT Press, 2000.

[9] G. Cabri, L. Leonardi, and F. Zambonelli. Agents for information retrieval: Issues of mobility and coordination. *Journal of Systems Architecture*, 46:1419–1433, 200.

[10] G. Cabri, L. Leonardi, and F. Zambonelli. Agents for information retrieval: Issues of mobility and coordination. *Journal of Systems Architecture*, 46(15):1419–1433, 2000.

[11] 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, and E.L. Wimmers. Object exchange across heterogeneous information sources. Technical report, Stanford University, 1994.

[12] R. G. G. Cattell, editor. *The Object Database Standard: ODMG93*. Morgan Kaufmann Publishers, San Mateo, CA, 1994.

[13] C.-C. K. Chang and H. Garcia-Molina. Mind Your Vocabulary: Query Mapping Across Heterogeneous Information Sources. In *Proc. of ACM SIGMOD*, pages 335–346, 1999.

[14] S. Chawathe, H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakostantinou, J.Ullman, and J. Widom. The TSIMMIS project: Integration of heterogeneous information sources. In *IPSJ Conference, Tokyo, Japan*, 1994. ftp://db.stanford.edu /pub/chawathe/1994/ tsimmis-overview.ps.

[15] FIPA. Specifications, 97.

[16] M. R. Genesereth, A. M. Keller, and O. Duschka. Infomaster: An information integration system. In *Proc. of ACM SIGMOD*, 1997.

[17] L. M. Haas, D. Kossmann, E. L. Wimmers, and J. Yang. Optimizing queries across diverse data sources. In *Proc. of VLDB*, pages 276–285, 1997.

[18] N. R. Jennings and M. J. Wooldridge. Applications of intelligent agents. In Nicholas R. Jennings and Michael J. Wooldridge, editors, *Agent Technology: Foundations, Applications, and Markets*, pages 3–28. Springer-Verlag: Heidelberg, Germany, 1998.

[19] N. M. Karnik and A. R. Tripathi. Design issues in mobile-agent programming systems. *IEEE Concurrency*, 6(3):52–61, 1998.

[20] M. Klusch. Information agent technology for the internet: A survey. *Data and Knowledge Engineering*, 36(3):337–372, 2001.

[21] M. Klusch, S. Bergamaschi, P. Edwards, and P. Petta, editors. *Intelligent Information Agents Research and Development in Europe: An AgentLink Perspective*. LNCS State of the Art Surveys. Springer Verlag, Heidelberg, Germany, To be published.

[22] Telecom Lab. Jade - java agent development environment.

[23] A. Y. Levy, A. Rajaraman, and J. J. Ordille. Querying heterogeneous information sources using source descriptions. In *Proc. of VLDB*, pages 251–262, 1996.

[24] S. E. Madnick. From vldb to vmldb (very many large data bases): Dealing with large-scale semantic heterogenity. In *VLDB*, pages 11–16, 1995.

[25] M. Nodine, J. Fowler, T. Ksiezyk, B. Perry, M. Taylor, and A. Unruh. Active informaiton gathering in infosleuth. *International Journal of Cooperative Information Systems*, 9(1-2):3–27, 2000.

[26] Y. Papakonstantinou, H. Garcia-Molina, and J. Widom. Object exchange across heterogeneous information sources. In P. S. Yu and A. L. P. Chen, editors, *11th Conference on Data Engineering*, pages 251–260, Taipei, Taiwan, 1995. IEEE Computer Society.

[27] M. T. Roth and P. M. Schwarz. Don't scrap it, wrap it! A wrapper architecture for legacy data sources. In *Proc. of VLDB*, pages 266–275, 1997.

[28] Y. Shoham. Agent-oriented programming. *Artificial Intelligence*, 60:51–92, 1993.

[29] K. Sycara. In-context information management truough adaptative collaboration of intelligent agents. *Intelligent Information Agents*, pages 78–99, 1999.

[30] K. P. Sycara, M. Klusch, S. Widoff, and J. Lu. Dynamic service matchmaking among agents in open information environments. *SIGMOD Record*, 28(1):47–53, 1999.

[31] M. Wooldridge. *An Introduction to Multiagent Systems*. Wiley, 2002.