# A peer-to-peer information system for the semantic web

S. Bergamaschi, F. Guerra, and M. Vincini
{sonia.bergamaschi,francesco.guerra,maurizio.vincini}@unimo.it

Dipartimento di Ingegneria dell'Informazione
Università di Modena e Reggio Emilia
Via Vignolese 905, 41100 Modena, Italy

**Abstract.** Data integration, in the context of the web, faces new problems, due in particular to the heterogeneity of sources, to the fragmentation of the information and to the absence of a unique way to structure and view information. In such areas, the traditional paradigms, on which database foundations are based (i.e. client server architecture, few sources containing large information), have to be overcome by new architectures. The peer-to-peer (P2P) architecture seems to be the best way to fulfill these new kinds of data sources, offering an alternative to traditional client/server architecture.
In this paper we present the SEWASIE system that aims at providing access to heterogeneous web information sources. An enhancement of the system architecture in the direction of P2P architecture, where connections among SEWASIE peers rely on exchange of XML metadata, is described.

## 1 Introduction

Data integration has been extensively studied in the past, in the domain of company infrastructures. Data integration, in the context of the web, faces new problems, due in particular to the heterogeneity of sources, to the fragmentation of the information and to the absence of a unique way to structure and view information. In such areas, the traditional paradigms, on which database foundations are based (i.e. client server architecture, few sources containing large information), have to be overcome by new architectures. The peer-to-peer (P2P) architecture seems to be the best way to fulfill of these new kinds of data sources, offering an alternative to traditional client/server architecture. The most relevant advantages of the approach are related to the improved scalability (increased storage, increased bandwidth) and flexibility of the systems.

New problems, related to the definition of specific ways to describe the contents of a source, i.e. its metadata w.r.t this architecture, and to allow sources to exchange data with each other, have to be faced. These issues are being recently addressed, and partially resolved by (proposed) standard like XML (that allows systems to exchange data across different platform), RDF (that provides a uniform manner to describe sources), OWL (that is a proposal of a standard ontology definition language) and WSLD (that is the language to define web services). In this paper, we focus on research problems associated with knowledge management and search in data-sharing P2P systems. Starting from a previous work [6], we present the SEWASIE system that aims at providing access to heterogeneous web information sources. An enhancement of the system architecture in the direction of P2P architecture, where connections among SEWASIE peers rely on exchange of XML metadata, is described.

## 2 Peer-to-Peer approach

Though data-sharing P2P systems are capable of sharing enormous amounts of data (e.g., 0.36 petabytes on the Morpheus network as of October 2001), such a collection is useless without mechanisms allowing users to quickly understand and search for desired pieces of data. Designing such a mechanism is difficult in P2P systems for several reasons: scale of the system, unreliability of individual peers, different semantics associated to similar peers, etc. In particular it has to define the behavior of peers in three areas:

**Topology:** Defines how peers are connected to each other. In some systems (e.g. Gnutella, www.gnutella.com), peers may connect to whomever they wish. In other systems, peers are organized into a rigid structure, in which the number and nature of connections is dictated by the protocol. Defining a rigid topology may increase efficiency, but will restrict autonomy.

**Data placement:** Defines how data or metadata is distributed across the network of peers. For example, in Gnutella, each node stores only its own collection of data. In Chord[8], data or metadata is carefully placed across nodes in a deterministic fashion. In super-peer networks [7], metadata for a small group of peers is centralized onto a single super-peer.

**Message routing:** Defines how messages are propagated through the network. When a peer submits a query, the query message is sent to a number of the peer's "neighbors" (that is, nodes to whom the peer is connected), who may in turn forward the message sequentially or in parallel to some of their neighbors, and so on. When, and to whom, messages are sent is dictated by the routing

protocol. Often, the routing protocol can take advantage of known patterns in topology and data placement, in order to reduce the number of messages sent.

The distributed, heterogeneous and unstructured nature of the Web poses a new challenge to query-answering over multiple data sources. In particular, it is no longer realistic to assume that the involved data sources act as if they were a single (virtual) source, modeled as a global schema, as is done in classical data integration approaches. In this paper, we propose an alternative approach where we replace the role of a single virtual data source schema with a peer-to-peer approach relying on limited shared (or overlapping) vocabularies between peers. Since overlaps between vocabularies of peers will be limited, query processing will have to be approximate. We provide a formal model for such approximate query processing based on limited shared vocabularies between peers, and we show how the quality of the approximation can be adjusted in a gradual manner. The result is a flexible architecture for query-processing in large, distributed and heterogeneous environments, based on a formal foundation. This architecture is suitable for knowledge-sharing in the peer-to-peer-style networks that are expected to be typical of the Semantic Web.

## 3 Core topics

### 3.1 Global architecture

The idea underlying our proposal is that *at a local level things may be done more richly than at a wider level.* Each peer contains specific information about the involved domains but only an high layer of this knowledge should be exported to other peers by using a standard language and a standard model to represent the structure of the source. We should therefore envision a multi-level architecture, with local nodes and communities with strong ties helping them develop a strong integration of their knowledge and information, with a semantic context which is well defined and offers a globally integrated ontology to represent everything, while at a wider level the relationships among distinct nodes are established by means of weaker mappings. The definition of the architecture is guided by the following reference scenarios:

- **building and maintaining a new information node**: we think each web site is a peer of this P2P network. Updates on the web sites will involve a change in the exported information.
- **establishing and maintaining mapping relationships among distinct information nodes**, that directly derives of the previous point: a change of

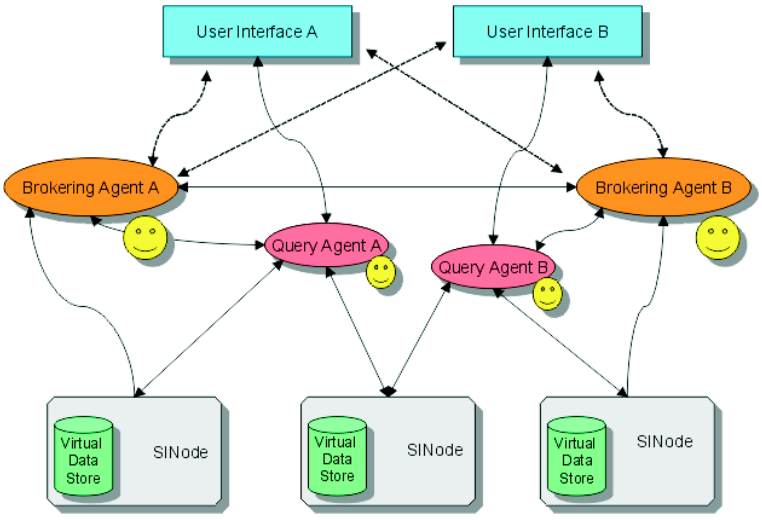


**Fig. 1.** General Overview of the peer-to-peer SEWASIE architecture

the peer information will generate a change of the mappings among different peers.

– **querying the system by a user**, in order to obtain the required information

A satisfying the aforementioned principles, goals, and desiderata system architecture is shown in Figure 1.

The **information nodes (SINodes)** group together modules which work to define and maintain a single administrative, or logical node of information presented to the whole network. A single information node may comprise several different systems.

The **user interface** contains modules which work together to offer an integrated user interaction with the semantic search system. Typically each user in the network will install and configure its own instance of the user interface. This interface has to be personalised and configured with the specific user profile and the reference to the ontologies which are commonly used by this user.

The **brokering agents** are the peers responsible for maintaining a view of the knowledge handled by the network, as well as the information on the specific content of some information nodes which are under direct control. These agents have direct control over a number of information nodes, and provide the means to publish a manifesto within the network of the locally held information.

The **query agents** are the carriers of the user query from the user interface to the information nodes, and have the task of solving a query, interacting with the brokering agent network. Starting from a user- or task- specified brokering agent, they may access other BAs, connect with several information nodes, collect partial answers, and integrate them.

### 3.2 Information Nodes

Information Nodes (**SINodes**) are mediator-based systems, each including a Virtual Data Store, an Ontology Builder, and a Query Manager. In [5, 1] we proposed the mediator-based system *MOMIS* (Mediator envirOnment for Multiple Information Sources) as a pool of tools to provide an integrated access to heterogeneous information. More to face the issues related to scalability in the large-scale, in [4, 2] we propose the exploitation of *mobile agents* in the information integration area, and, in particular, their integration in the *MOMIS* infrastructure.

**Virtual Data Store (VDS):** it represents a virtual view of the overall information managed within any SINode and consists of the managed information sources, wrappers, and a metadata repository. The managed Information Sources are heterogeneous collections of structured, semi-structured, or unstructured data, e.g. relational databases, XML or HTML documents.

Wrappers are the "docking stations" of the heterogeneous data sources. They are software modules in charge of the mediation between the internal representation of each data source and the functionalities of the SINode. Different wrappers have to be defined to cover structurally diverse sources. The wrapper interface will be uniform and independent of the underlying source type. Two major functions need to be performed by the wrappers: to support the translation of the structure of the information managed by local sources into the SINode description language and the translation of the queries from the SINode query language into the specific query language of the underlying source.

The architecture of the VDS module is inherently distributed (i.e. in most cases its functionality will be distributed among several host machines of different types). As a consequence, these components will all need to have inter-process communication functionalities to support the interaction.

**Ontology Builder (OB):** it is the collective name of a set of functionalities which will support the creation and maintenance of a *global virtual view* (GVV) of the managed sources and the mapping description between the GVV itself and the integrated sources. The ontology building process is a coopera-

tive one, involving the designers; it begins with the creation of a common thesaurus of the information provided by wrappers, that is terminological intensional and extensional relationships describing intra-schema knowledge about classes and attributes of each source schemas. Based on such information and on designer supplied relationships capturing specific domain knowledge, the OB performs semiautomatic intra and inter-schema analysis by exploiting lexicon derived relationships, which are based on processes like synonyms identification or generalisation-specialisation relations, and inferring new relationships.

All these relationships are considered in the subsequent phase of construction of the ontology. Such an activity is based on hierarchical clustering techniques and supports the emergence of a number of global classes (GVV) representative of all the classes coming from the sources and of a mapping description between the GVV and the local sources.

**Query Manager (QM):** It is the coordinated set of functions which take an incoming query, define a decomposition of the query w.r.t. the mapping of the GVV of the SINode onto the data sources relevant for the query, send the queries to the wrappers in charge of the data sources, collect their answers, perform any residual filtering as necessary, and finally deliver to the requesting query agent.

### 3.3 Brokering Agents

In a distributed information system it is necessary to maintain and share the information about the knowledge made available by the system. In order to facilitate the interoperability and reusability of knowledge resources, we need to provide a flexible infrastructure to taking into account the change of data and metadata and not to provide a specific application useful only to a target domain. The proposed approach builds on the W3C XML standard and uses an object oriented data and query model [5] throughout the SEWASIE Network. In the proposed semantic search system architecture this task will be performed by the brokering agents (BAs), which are peers of the network that share knowledge and metadata. The knowledge within a brokering agent is represented as an ontology, i.e. a network of interrelated concepts. Relationships between concepts are defined as mappings expressed by a formal language. The language, called $ODL_{I^3}$ (see [5] for a detailed description) is based on ODMG object model and OLCD Description Logic [3] and is represented by XML language. The $ODL_{I^3}$ language [5] may be used to describe heterogeneous schemata of data sources in a common way. In the context of the global ontology of an information node $ODL_{I^3}$ introduces new constructors useful in the integration process and in the

global integrated view representation. In particular, there are intensional relationships expressing inter-schema knowledge for the source schemas defined between classes and attributes names (terms): SYN (Synonyms), BT (Broader Terms), NT (Narrower Terms) and RT (Related Terms). Intensional relationships SYN, BT and NT between two classes may be "strengthened" by establishing that they are also extensional relationships. Moreover, mappings between the global integrated view and schemata of data sources can be defined.

A brokering agent knows exactly all the ontologies which are present in the underlying information nodes, and has general information about related (to its own) ontologies in other nodes.

The depth of the information of the BA becomes more and more shallow with the distance (with respect to some metrics) between the ontologies for which it is "expert" (those of its underlying information nodes) and other ontologies covered within the system. Its information on other (non local) ontologies is incomplete.

Different brokering agent roles may be envisioned, depending on the business model of the organisation which deploys the brokering agent. A company (or a group of companies) may establish a brokering agent to manage a common access to its information sources. On the other end, a specialised information brokering enterprise may establish a brokering agent that combines ontologies provided by several other brokering agents and clustering around a specific domain. Thus, we can identify the following classes of brokering agents:

- local service BAs are servicing a single information node or a group of nodes and are usually close to the nodes
- pure informant BAs are collectors of references to several brokering agents and may specialise in a certain domain, and may be positioned anywhere in the network

At an abstract level, the operation of the brokering agent can be divided into two phases:

1. Design Phase (mapping time)
   - The brokering agent receives a local ontology describing data available from a information node and has to map it into its own map of ontologies.
   - The brokering agent receives information about other ontologies from other brokering agents. This information also has to be mapped into the existing ontology of the brokering agent.
2. Runtime Phase (query time)
   - A query agent wants to know where it can find the information for a query. The terms of the query are then matched with the concepts known

by the brokering agent. The brokering agent looks up its internal meta information and returns pointers to the information nodes or other brokering agents which have the requested information (or parts thereof).
- A component of the system (e.g., the query interface) extracts the complete ontology which is managed by the brokering agent.

The design and runtime phases are not strictly separated. Ontologies need to be mapped and updated during the whole life cycle of a brokering agent.

The crucial role of the brokering agent is the creation and maintenance of the map of semantic relationships among concepts from different information nodes in the system. In particular, the correlation among the concepts coming from different sources (ontologies of information nodes and other brokering agents) relies on terminological relationships. They are created by the brokering agent which, in a (semi-) automatic way, analyses the meaning of the concepts in different ontologies and tries to discover terminological relationships among them by exploiting lexical ontologies such as Wordnet. Once the repository of these mappings has been created, the brokering agent is in charge of its maintenance: changes in the network have to be integrated to make the repository consistent with the new scenario.

At runtime, when a request from a query agent is received, the quality of the answer of the BA (and of the query agent) depends to a large degree on the quality of the semantic relationships that have been created. Incorrect relationships will lead to incorrect answers of the query agent, e.g. data that should not be included in an answer to a query will be delivered as result. Incomplete (or missing) relationships will lead to incomplete results, e.g. data that should be included in the answer is not delivered as a result (although it is somewhere available in the network). Thus, special attendance has to be given to the creation of the semantic relationships. Automatic creation of the semantic relationships is possible, but the creation of relationships based only on the lexical similarity between terms will lead to incorrect and incomplete results. On the other hand, manual control of the creation is not possible in every case as a brokering agent might handle a large number of information nodes. Thus, updates to the semantic relationships might happen quite frequently and manual control would significantly slow down the process of establishing semantic relationships.

### BA's Architecture

The basic architectural schema of the BA is described in figure 2. The map keeper is the maintainer of the mapping information concerning the local information nodes and the other BAs and supported ontologies. It builds is a partial

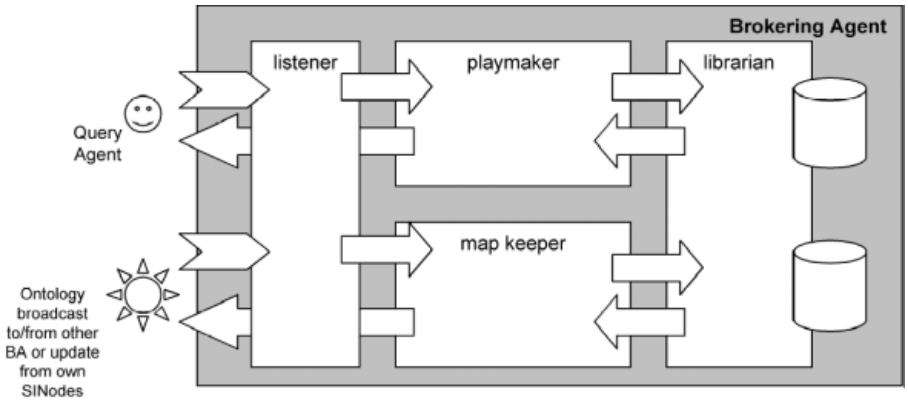


**Fig. 2.** Basic Architectural Schema of the BA

description of the information available within the system, with varying degrees of precision and richness; higher detail will be available for the local information nodes, less detail will be available for other ontologies.

The playmaker receives a request from the listener on behalf of a query agent, and identifies the information nodes or other BAs which may offer an answer (or parts thereof). It connects with the waiting query agent and provides the corresponding directions, consisting of addresses of SINodes and/or other BAs, plus ontology information so that the Query Agent can perform the necessary query rewriting before submitting the queries to the SINodes.

The listener process is in charge of receiving requests from other entities in the system; in fact, it embeds much of the transport layer and communication support, including the enforcement of security and policy features. The listener is in charge of authentication (identification of the user, identification of the server), establishing the parameters for a secure connection, logging, and verifying the user profile vs the requested service. After receiving a request it dispatches it to an instance of the playmaker or the map keeper for further processing. In this phase the listener may also act as a load balancer when the playmaking and map keeping functionalities are deployed on a distributed architecture.

The librarian is a service component, being the provider of repository services to the other components of the BA. It is made of a listening module and one or more repository modules. Wrappers may be used to encapsulate different long term storage devices (like DBMSs, file systems, XML files, or whatever else).

### 3.4 Query Agents

The query agent is the actual carrier of a query from a user to the system. The term "query" has to be interpreted as a general statement in a known intermediate query language which is interpreted by SINode components (SINode query managers) within the system. This query includes information on the context of the user at the time of the establishment of the query. This means that information about the specific activity of the user, his/her preferences, feedback on appreciation of the results of similar queries in the past under similar circumstances, and so on, are embedded in the query.

The query agent is the network query manager and "motion item" of the system, and it should be the only carrier of information among the users and the system. Therefore it should be able to do several jobs:

- carrying a query and the relevant pieces of the user ontology/profile which may help the brokering agents to qualify the semantics of the query (this includes both the case of a user–defined query and the case of system–defined query, like a query issued by the monitoring agent)
- defining the query plan, doing the query rewriting for a specific SINode, and merging the results from several SINodes
- processing the information given by the BA and identifying the SINodes to be accessed for answering the query, and on which further BA peers to contact to possible get more answer to the query
- carrying back the results (both data and metadata)

Query agents are instantiated by the users for each request to the system, or also by the system itself for a load balancing criteria. In a such a way, the user performs a global search in a virtual environment and ignores where the information has been actually maintained and managed.

## 4 Concluding Remarks

Within highly decentralized, dynamic and mobile information system, the traditional client/server architecture seems not to be sufficient. Searching in the web by using a client/server paradigm, i.e. search engines, cover only a subset of available information and often the information is not up to date due to large amount of time for the crawling operation.
In contrast, querying information sources in a P2P network performs the retrieval of up to date information stored only in the relevant data stores. In this context Napster and Gnutella are examples of application in which every peer

shares information with all other peers of the community.
In this paper, we present the SEWASIE system that aims at providing access to heterogeneous information sources. Our approach, like above systems, supposes each peer shares information and each data source (i.e. customers of SEWASIE network) decides directly what data and information they want to make available over the network, without the need for publishing and maintaining them in a specific server. So that, each peer (i.e. the BA) is responsible for the subset of information it makes available on the network and no centralized global information system is required. In fact, by choosing a peer-to-peer platform for our application, we have overcome the need for building a centralized (virtual or materialized) repository for a large amount of semi-structured information, with the related problems of synchronizing multiple accesses to the data and the scalability problems of each centralized architecture.

**Acknowledgements**

This work is supported in part by the 5th Framework IST programme of the European Community through project SEWASIE within the Semantic Web Action Line. The SEWASIE consortium comprises in addition to the author' organization (Sonia Bergamaschi is the coordinator of the project), the Universities of Aachen RWTH (M. Jarke), Roma La Sapienza (M. Lenzerini, T. Catarci), Bolzano (E. Franconi), as well as IBM Italia (G. Vetere), Thinking Networks AG (C. Engels) and CNA (A. Tavernari) as user organisation.

## References

1. D. Beneventano, S. Bergamaschi, S. Castano, A. Corni, R. Guidetti, G. Malvezzi, M. Melchiori, and M. Vincini. Information integration: The momis project demonstration. In *VLDB 2000, Proceedings of 26$^{th}$ International Conference on Very Large Data Bases, September, 2000, Cairo, Egypt*, pages 611–614. Morgan Kaufmann, 2000.
2. D. Beneventano, S. Bergamaschi, G. Gelati, F. Guerra, and M. Vincini. Miks: an agent framework supporting information access and integration. In S. Bergamaschi, M. Klusch, P. Edwards, and P. Petta, editors, *Intelligent Information Agents - The AgentLink Perspective Lecture Notes in Computer Science N. 2586*, pages 22–49, Heidelberg, Germany, 2003. Springer-Verlag.
3. D. Beneventano, S. Bergamaschi, and C. Sartori. Description logics for semantic query optimization in object-oriented database systems. In *ACM Transaction on Database Systems, to be pubblished (March 2003)*, 2003.
4. 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.

5. 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.
6. S. Bergamaschi and F. Guerra. Peer to peer paradigm for a semantic search engine. In *Workshop on Agents and Peer-to-Peer Computing*, July 2002. LNCS 2530, Springer.
7. Neil Daswani, Hector Garcia-Molina, and Beverly Yang. Open problems in data-sharing peer-to-peer systems. In *Proceedings of the 9th International Conference on Database Theory*, 2003.
8. I. Stoica, R. Morris, D. Liben-Nowell, D. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan. Chord: a scalable peer-to-peer lookup protocol for internet applications. *IEEE/ACM Transactions on Networking*, 11(1):17–32, 2003.