SEWASIE
Semantic Webs and AgentS in Integrated Economies
IST-2001-34825

WP3
Task T3.1
Deliverable D3.1

General framework for query reformulation

(FINAL, 10/02/2003)

Abstract – The purpose of this document is to define a general semantic framework for query management in Sewasie. The process of query reformulation, query merging and information reconciliation in SEWASIE is based on data integration within each SINode, and on the network of SEWASIE brokering agents. The Query agents are the carriers of the user query from the user interface to the SINodes and are responsible for the task of answering queries, based on the interaction with the brokering agent network. In this document, we discuss the above mentioned task, with the goal of providing a general formal framework, that will be used as a basis for the design of query management techniques. In particular, we distinguish between the formal framework that will be used in the design of the query manager module within each SINode, and the one that will be used in the design of techniques for the interaction between query agents and brokering agents.

Zoran Majkic
UNIROMA

Document information

Document ID code	D3.1		
Keywords	Query management, Query reformulation, Query agent, Data integration		
Classification	**FINAL**	Date of reference	**10/02/2003**
Distribution level	**Partners of the Sewasie consortium, and EU commission**		

Editor	Zoran Majkic	UNIROMA
Authors	Maurizio Lenzerini	UNIROMA
	Zoran Majkic	UNIROMA
Reviewer	Guido Vetere	IBM

Version history		
Date	*Version*	*Description*
08/01/2003	DRAFT 1	Draft version
15/01/2003	DRAFT 2	Second draft
10/02/2003	FINAL	Final version

Copyright notices

Contents

1 Executive summary

This document is organized in three sections. The first one (section 2) recalls the various tasks of the query management in Sewasie, namely, to single out SINodes managed by BA, to single out the other brokering agents that are relevant for a query, to reconstruct the answer on the basis of the answers of a number of SINodes. Section 3 deals with the problem of answering a query posed to a single SINode: this task is carried out by the query manager of the SINode of interest and is based on the semantics of the general framework for data integration. Intuitively, the source schema describes the structure of the sources, where the real data are, while the global schema provides a reconciled, integrated, and virtual view of the underlyng sources. The assertions in the mapping establish the connection between the elements of the global virtual view and those of the source schema. Queries are posed in terms of the global virtual view, and are expressed as a conjunctive queries over the global virtual view alphabet. Section 4 is devoted to query management in the context of different brokering agents. One of the basic characteristics of the brokering agents in Sewasie is that they act at the same level, with no unifying structure above them. We show here that, what is needed is a mechanism that is able to define mapping between a number of brokering agents without resorting to any unifying conceptual structure. In other words, the formal framework we are illustrating follows the Peer-to-Peer (P2P) paradigm. We first present an overview of the most well-known P2P systems, and then we describe the basic elements of a new semantics that correctly captures the modular structureof the Sewasie broekring agent network, and opens up the possiblity of effective query answering techniques.

2 Query management in Sewasie

The basic user query scenario we have in mind concerns a user at a workstation (or handling a handheld computer, or a cellular phone with network connection capabilities), looking for information on a topic, possibly within the context of a broader-scoped task. The user may then issue a request expressed in some fixed language to the network. The user interface translates the user request into a query expressed in a formal language (user query language), and sends a probe out (the query agent) scouting for answers.

The query agent is the carrier of the user query from the user interface to the SINodes, where concrete data are located. One basic task of the query agent is to interact with the brokering agents, in order to solve the query. Starting from a specified brokering agent, the query agent initiates a process constituted by a series of queries posed to the various brokering agents with the goal of getting information on the matter of interest (see Section 6 of [12]). A typical interaction between a query agent and a brokering agent may imply that the brokering agent will provide directions to relevant SINodes and information on SINode contents, or reference the query agent to other brokering agents. The query agent will then move to such nodes and query them, or may move on to the other brokering agents to ask them for directions again. During this process, the query agent is informed by the brokering agents about which SINodes contain relevant data, so that the query agent may access such SINodes, collect partial answers and integrate them.

When the query agent is informed that a certain SINodes may have relevant answers to the query, it has to issue the right query to such SINode. Moreover, after receiving the various answers from the SINodes, the query agent has to integrate them (data reconciliation) in a way that is meaningful to the user.

A query issued to an SINode is managed by the query manager associated to such SINode. A query manager is the coordinated set of functions which take an incoming query, define a decomposition of the query according with the mapping of the global virtual view of the SINode onto the specific data sources available and relevant for the query, sends the queries to the wrappers in charge of the data sources, collects their answers, performs any residual filtering as necessary, and finally delivers whatever is left to the requesting query agent.

From all the above observations, one can infer that query management in the Sewasie framework involves different tasks, summarized as follows:

- Given a query Q expressed in terms of an ontology understood by a brokering agent B,

  1. Single out the SINodes $S_1, \ldots, S_n$ managed by B that are relevant for computing the answer to Q, and reformulate Q in terms of n queries $Q_1, \ldots, Q_n$, to be posed to the SINodes $S_1, \ldots, S_n$, respectively.
  2. Single out the brokering agents $B_1, \ldots, B_m$ (besides B) that may have links to SINodes containing relevant information for computing the answer to Q, and reformulate Q in terms of m queries $T_1, \ldots, T_m$, to be posed to the brokering agents $B_1, \ldots, B_m$, respectively.
  3. Reconstruct the answer to Q on the basis of the answers to $Q_1, \ldots, Q_m, T_1, \ldots, T_m$.

  Note that step (2) above is recursive, in the sense that answering a query T_i posed to the brokering agent B_i is done through the very same process we are describing. This means that the overall strategy for query management must deal with the problem of how to stop recursion. We refer the reader to Section 4 for a discussion about this issue. Note also that both step (1) and step (2) are carried out by the query agent on the basis of the mata-data (mapping) exported by the brokering agent B.

- Given a query posed in terms of the virtual global view associated to an SINode, retrieve the answer to the query. This task is carried out by the query manager of the SINnode of interest, and is characterized as follows. The first goal of this task is to derive a query plan that is able to correctly access the data sources under the control of that SINode. The

second goal of this task is to execute the query, thus computing the corresponding answer. For all the issues regarding the structure of the SINodes, especially those related to the representation of the ontology managed by SINodes, we refer the reader to [27].

In the rest of this document, we discuss the above tasks, with the goal of providing a general formal framework, that will be used as a basis for the design of query management techniques. In particular, the formal development presented in Section 3 will be used in the design of the query manager module within each SINode, with the goal of devising suitable techniques for answering a query posed to such node. In Section 4 we present the formal foundations that will be used in the design of techniques to be used by the query agent in order to decide which queries to issue to the various brokering agents.

It should be taken into account that the purpose of this document is to define a general semantic framework for query management in Sewasie, and not to illustrate the actual techniques for query management. The development of such techniques will build on the framework presented here, and will be the subject of subsequent tasks within Workpakage WP3.

3 Query management in the context of a single SINodes

In this section we deal with the problem of answering a query posed to an SINode. As we said before, this task is carried out by the query manager of the SINnode of interest, and is characterized as follows: Given a query posed in terms of the virtual global view associated to the SINode, derive a query plan that is able to correctly access the data sources under the control of that SINode, and execute the query according to this plan.

Since each SINode provides an abstract view of information stored in several sources, each SINode can be seen as a local data integration system with a global schema. Data integration is the problem of combining data residing at different sources, and providing the user with a unified view of these data [19, 20, 30]. The kind of data integration systems we refer to in this section is characterized by an architecture based on a global schema (global virtual view) and a set of sources. The sources contain the real data, while the global schema provides a reconciled, integrated, and virtual view of the underlying sources.

Modeling the relation between the sources and the global schema is a crucial aspect in data integration. Three basic approaches have been proposed to this purpose.

- The first approach, called global-as-view (GAV), requires that the global schema is expressed in terms of the data sources.
- The second approach, called local-as-view (LAV), requires the global schema to be specified independently from the sources, and the relationships between the global schema and the sources are established by defining every source as a view over the global schema.
- The third approach, called GLAV, is a combination of the two previous methods.

Our next goal in this section is to discuss the characteristics of these three modeling mechanisms.

Irrespectively of the method used for the specification of the mapping between the global schema and the sources, one basic service provided by the data integration system is to answer queries posed in terms of the global schema. Given the architecture of the system, query processing in data integration requires a reformulation step: the query over the global schema has to be reformulated in terms of a set of queries over the sources.

Since sources are in general autonomous, in many real-world applications the problem arises of mutually inconsistent data sources. In practice, this problem is generally dealt with by means of suitable transformation and cleaning procedures applied to data retrieved from the sources.

The rest of this section is organized as follows. In the next subsection, we illustrate a general framework for data integration within one SINode. Then, we formally define the three approaches to the specification of the mapping in a data integration system, namely LAV¡ GAV, and GLAV. Finally, we discuss the main issues arising in the design of a query manager within one SINode.

3.1 General framework for data integration within an SINnode

As we said before, we conceive a single SINode as a data integration systems based on a so-called global schema. In other words, each SINode combines the data residing at different sources, and provide the external user with a unified view of these data. Such a unified view is represented by the global schema, and provides a reconciled view of all data, which can be queried by the user. Obviously, one of the main task in the design of an SINode is to establish the mapping between the sources and the global schema. Such a mapping should be suitably taken into account in formalizing the notion of SINode.

It follows that, from the perspective of the query manager, the main components of an SINode are the global schema, the sources, and the mapping. Thus, we formalize an *SINode* $\mathcal{I}$ in terms of a triple $\langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, where

- $\mathcal{G}$ is the *global schema*, expressed in a language $\mathcal{L}_\mathcal{G}$ over an alphabet $\mathcal{A}_\mathcal{G}$. The alphabet comprises a symbol for each element of $\mathcal{G}$ (i.e., relation if $\mathcal{G}$ is relational, class if $\mathcal{G}$ is object-oriented, etc.).
- $\mathcal{S}$ is the *source schema*, expressed in a language $\mathcal{L}_\mathcal{S}$ over an alphabet $\mathcal{A}_\mathcal{S}$. The alphabet $\mathcal{A}_\mathcal{S}$ includes a symbol for each element of the sources.
- $\mathcal{M}$ is the *mapping* between $\mathcal{G}$ and $\mathcal{S}$, constituted by a set of *assertions* of the forms

$$q_\mathcal{S} \rightsquigarrow q_\mathcal{G},$$
$$q_\mathcal{G} \rightsquigarrow q_\mathcal{S}$$

  where $q_\mathcal{S}$ and $q_\mathcal{G}$ are two queries of the same arity, respectively over the source schema $\mathcal{S}$, and over the global schema $\mathcal{G}$. Queries $q_\mathcal{S}$ are expressed in a query language $\mathcal{L}_{\mathcal{M},\mathcal{S}}$ over the alphabet $\mathcal{A}_\mathcal{S}$, and queries $q_\mathcal{G}$ are expressed in a query language $\mathcal{L}_{\mathcal{M},\mathcal{G}}$ over the alphabet $\mathcal{A}_\mathcal{G}$. Intuitively, an assertion $q_\mathcal{S} \rightsquigarrow q_\mathcal{G}$ specifies that the concept represented by the query $q_\mathcal{S}$ over the sources corresponds to the concept in the global schema represented by the query $q_\mathcal{G}$ (similarly for an assertion of type $q_\mathcal{G} \rightsquigarrow q_\mathcal{S}$). We will discuss several ways to make this intuition precise in the following sections.

Intuitively, the source schema describes the structure of the sources, where the real data are, while the global schema provides a reconciled, integrated, and virtual view of the underlying sources. The assertions in the mapping establish the connection between the elements of the global schema and those of the source schema.

Queries to $\mathcal{I}$ are posed in terms of the global schema $\mathcal{G}$, and are expressed in a query language $\mathcal{L}_\mathcal{Q}$ over the alphabet $\mathcal{A}_\mathcal{G}$. A query is intended to provide the specification of which data to extract from the virtual database represented by the SINode.

The above definition of SINode is general enough to capture virtually all approaches in the literature. Obviously, the nature of a specific approach depends on the characteristics of the mapping, and on the expressive power of the various schema and query languages. For example, the language $\mathcal{L}_\mathcal{G}$ may be very simple (basically allowing the definition of a set of relations), or may allow for various forms of integrity constraints to be expressed over the symbols of $\mathcal{A}_\mathcal{G}$. Analogously, the type (e.g., relational, semistructured, etc.) and the expressive power of $\mathcal{L}_\mathcal{S}$ varies from one approach to another.

The main goal of this deliverable is to define the general framework for query management in Sewasie. One of the basic task within this purpose is to specify the semantics of an SINode, form the perspective of query management. Such a specification is crucial for characterizing the nature of query answering within a single SINode.

In what follows, a database (DB) for a schema $\mathcal{T}$ is simply a set of collection of sets, one for each symbol in the alphabet of $\mathcal{T}$ (e.g., one relation for every relation schema of $\mathcal{T}$, if $\mathcal{T}$ is relational, or one set of objects for each class of $\mathcal{T}$, if $\mathcal{T}$ is object-oriented, etc.). We also make the assumption that the structures constituting the databases involved in our framework (both the global database and the source databases) are defined over a fixed domain Γ.

In order to assign semantics to an SINode $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, we start by considering a *source database* for $\mathcal{I}$, i.e., a database $\mathcal{D}$ that conforms to the source schema $\mathcal{S}$ and satisfies all constraints in $\mathcal{S}$. Based on $\mathcal{D}$, we now specify which is the information content of the global schema $\mathcal{G}$. We call *global database* for $\mathcal{I}$ any database for $\mathcal{G}$. A global database $\mathcal{B}$ for $\mathcal{I}$ is said to be *legal with respect to* $\mathcal{D}$, if:

- $\mathcal{B}$ is legal with respect to $\mathcal{G}$, i.e., $\mathcal{B}$ satisfies all the constraints of $\mathcal{G}$;
- $\mathcal{B}$ satisfies the mapping $\mathcal{M}$ with respect to $\mathcal{D}$.

The notion of $\mathcal{B}$ satisfying the mapping $\mathcal{M}$ with respect to $\mathcal{D}$ depends on how to interpret the assertions in the mapping. We will see in the next section that several approaches are conceivable. Here, we simply note that, no matter which is the interpretation of the mapping, in general, several global databases exist that are legal for $\mathcal{I}$ with respect to $\mathcal{D}$. This observation motivates the relationship between data integration and databases with incomplete information [31], which will be discussed in several ways later on in the paper.

Finally, we specify the semantics of queries posed to an SINode. As we said before, such queries are expressed in terms of the symbols in the global schema of $\mathcal{I}$. In general, if q is a query of arity n and $\mathcal{DB}$ is a database, we denote with $q^{\mathcal{DB}}$ the set of tuples (of arity n) in $\mathcal{DB}$ that satisfy q.

Given a source database $\mathcal{D}$ for $\mathcal{I}$, the answer $q^{\mathcal{I},\mathcal{D}}$ to a query q in $\mathcal{I}$ with respect to $\mathcal{D}$, is the set of tuples t of objects in Γ such that $t \in q^{\mathcal{B}}$ for *every* global database $\mathcal{B}$ that is legal for $\mathcal{I}$ with respect to $\mathcal{D}$. The set $q^{\mathcal{I},\mathcal{D}}$ is called the set of *certain answers* to q in $\mathcal{I}$ with respect to $\mathcal{D}$.

Note that, from the point of view of logic, finding certain answers is a logical implication problem: check whether it logically follows from the information on the sources that t satisfies the query.

3.2 Modeling the data integration component in an SINode

One of the most important aspects in the design of an SINode is the specification of the correspondence between the data at the sources and those in the global schema. Such a correspondence is modeled through the notion of mapping as introduced in the previous section. It is exactly this correspondence that will determine how the queries posed to the system are answered.

Three basic approaches for specifying the mapping in a data integration system have been proposed in the literature, called *local-as-view* (LAV), *global-as-view* (GAV), and GLAV respectively [30, 19]. We discuss these approaches separately in the remainder of this subsection.

3.2.1 Local as view

In an SINode $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$ based on the LAV approach, the mapping $\mathcal{M}$ associates to each element s of the source schema $\mathcal{S}$ a query $q_{\mathcal{G}}$ over $\mathcal{G}$. In other words, the query language $\mathcal{L}_{\mathcal{M},\mathcal{S}}$ allows only expressions constituted by one symbol of the alphabet $\mathcal{A}_{\mathcal{S}}$. Therefore, a LAV mapping is a set of assertions, one for each element s of $\mathcal{S}$, of the form

$$s \rightsquigarrow q_{\mathcal{G}}$$

From the modeling point of view, the LAV approach is based on the idea that the content of each source s should be characterized in terms of a view $q_{\mathcal{G}}$ over the global schema. A notable case of this type is when the data integration system is based on an enterprise model, or an ontology [17]. Note that the LAV approach favors the extensibility of the system: adding a new source simply means enriching the mapping with a new assertion, without other changes.

To better characterize each source with respect to the global schema, several authors have proposed more sophisticated assertions in the LAV mapping, in particular with the goal of establishing the assumption holding for the various source extensions [1, 16, 23, 9]. Formally, thisv means that in the LAV mapping, a new specification, denoted $as(s)$, is associated to each source

element s. The specification $as(s)$ determines how accurate is the knowledge on the data satisfying the sources, i.e., how accurate is the source with respect to the associated view $q_{\mathcal{G}}$. Three possibilities have been considered in the literature, namely sound, complete, and exact. Given the dynamic characteristics of the Sewasie architecture, we restrict our attention to the first option, i.e., we assume that sources are sound. Note that this is the most common assumption in the whole literature on data integration. v A sound source is a source whose extension provides a subset of the tuples satisfying the view definition $q_{\mathcal{G}}$ associated to it by the mapping. In other words, given a source database $\mathcal{D}$, from the fact that a tuple is in $s^{\mathcal{D}}$ one can conclude that it satisfies the associated view over the global schema, while from the fact that a tuple is not in $s^{\mathcal{D}}$ one cannot conclude that it does not satisfy the corresponding view. This is coherent with the fact that the system is highly dynamic, and thus, new information may come into the system whenever new sources are considered. Formally a database $\mathcal{B}$ *satvisfies the assertion*

$$s \rightsquigarrow q_{\mathcal{G}}$$

ith respect to $\mathcal{D}$ if

$$s^{\mathcal{D}} \subseteq q_{\mathcal{G}}^{\mathcal{B}}$$

Note that, from a logical point of view, a sound source s with arity n is modeled through the first order assertionv

$$\forall \mathbf{x}\, s(\mathbf{x}) \rightarrow q_{\mathcal{G}}(\mathbf{x})$$

where $\mathbf{x}$ denotes variables $x_1, \ldots, x_n$.

Information Manifold [22], and the system presented in [25] are examples of LAV systems. Information Manifold expresses the global schema in terms of a Description Logic [2], and adopts the language of conjunctive queries as query languages $\mathcal{L}_{\mathcal{Q}}$, and $\mathcal{L}_{\mathcal{M},\mathcal{G}}$. The system described in [25] uses an XML global schema, and adopts XML-based query languages for both user queries and queries in the mapping. More powerful schema languages for expressing the global schema are reported in [13, 18, 8, 7]. In particular, [13, 18] discusses the case where various forms of relational integrity constraints are expressible in the global schema, including functional and inclusion dependencies, whereas [8, 7] consider a setting where the global schema is expressed in terms of Description Logics [4], which allow for the specification of various types of constraints.

3.2.2 Global as view

In the GAV approach, the mapping $\mathcal{M}$ associates to each element g in $\mathcal{G}$ a query $q_{\mathcal{S}}$ over $\mathcal{S}$. In other words, the query language $\mathcal{L}_{\mathcal{M},\mathcal{G}}$ allows only expressions constituted by one symbol of the alphabet $\mathcal{A}_{\mathcal{G}}$. Therefore, a GAV mapping is a set of assertions, one for each element g of $\mathcal{G}$, of the form

$$g \rightsquigarrow q_{\mathcal{S}}$$

From the modeling point of view, the GAV approach is based on the idea that the content of each element g of the global schema should be characterized in terms of a view $q_{\mathcal{S}}$ over the sources. In some sense, the mapping explicitly tells the system how to retrieve the data when one wants to evaluate the various elements of the global schema. This idea is effective whenever the data integration system is based on a set of sources that is stable. Note that, in principle, the GAV approach favors the system in carrying out query processing, because it tells the system how to use the sources to retrieve data. However, extending the system with a new source is now a problem: the new source may indeed have an impact on the definition of various elements of the global schema, whose associated views need to be redefined. v Analogously to the case of LAV mapping, we only consider the case of sound sources. A database $\mathcal{B}$ *satisfies the assertion*

$$g \rightsquigarrow q_{\mathcal{S}}$$

with respect to a source database $\mathcal{D}$ if

$$q_{\mathcal{S}}^{\mathcal{D}} \subseteq g^{\mathcal{B}}$$

The logical characterization of GAV is therefore through the first order assertions

$$\forall \mathbf{x}\, q_{\mathcal{S}}(\mathbf{x}) \rightarrow g(\mathbf{x})$$

It is interesting to observe that the implicit assumption in many GAV proposals is the one of exact views. Indeed, in a setting where all the views are exact, there are no constraints in the global schema, and a first order query language is used as $\mathcal{L}_{\mathcal{M},\mathcal{S}}$, a GAV data integration system enjoys what we can call the "single database property", i.e., it is characterized by a single database, namely the global database that is obtained by associating to each element the set of tuples computed by the corresponding view over the sources. This motivates the widely shared intuition that query processing in GAV is easier than in LAV. However, it should be pointed out that the single database property only holds in such a restricted setting.

In particular, the possibility of specifying constraints in $\mathcal{G}$ greatly enhances the modeling power of GAV systems, especially in those situations where the global schema is intended to be expressed in terms of a conceptual data model, or in terms of an ontology [5]. In these cases, the language $\mathcal{L}_{\mathcal{G}}$ is in fact sufficiently powerful to allow for specifying, either implicitly or explicitly, various forms of integrity constraints on the global database.

Most of current data integration systems follow the GAV approach. Notable examples are TSIMMIS [14], Garlic [10], COIN [15], MOMIS [3], Squirrel [32], and IBIS [6]. Analogously to the case of LAV systems, these systems usually adopt simple languages for expressing both the global and the source schemas. IBIS is the only system we are aware of that takes into account integrity constraints in the global schema.

3.2.3 Combining LAV and GAV

A third kind of mapping, combining the advantages of both LAV and GAV, is called GLAV. In GLAV, the mapping between $\mathcal{G}$ and $\mathcal{S}$ is constituted by a set of assertions of the form

$$q_{\mathcal{S}} \rightsquigarrow q_{\mathcal{G}}\,,$$

where $q_{\mathcal{S}}$ and $q_{\mathcal{G}}$ are two queries of the same arity, respectively over the source schema $\mathcal{S}$, and over the global schema $\mathcal{G}$.

Again, we assume that sources provide sound information, and therefore, we formalize the above assertions as follows. A database $\mathcal{B}$ *satisfies the assertion* $q_{\mathcal{S}} \rightsquigarrow q_{\mathcal{G}}$ with respect to a source database $\mathcal{D}$ if

$$q_{\mathcal{S}}^{\mathcal{D}} \subseteq g_{\mathcal{G}}^{\mathcal{B}}$$

The logical characterization of GAV is therefore through first order assertions of the form

$$\forall \mathbf{x}\, q_{\mathcal{S}}(\mathbf{x}) \rightarrow q_{\mathcal{G}}(\mathbf{x})$$

It is easy to see that both LAV and GAV are just special cases of the GLAV mapping. Since GLAV offers the advantages of the other approaches, it is our aim in Sewasie to study data integration strategies that are based on the GLAV approach.

3.3 Issues in the design of the query manager for a single SINode

As we said before, most GAV data integration systems do not allow integrity constraints in the global schema, and assume that views are exact. It is easy to see that, under these assumptions, query processing can be based on a simple unfolding strategy. When we have a query q over the alphabet $\mathcal{A}_{\mathcal{G}}$ of the global schema, every element of $\mathcal{A}_{\mathcal{G}}$ is substituted with the corresponding

$s_1^{\mathcal{D}}$:

12	*calvin*	*rome*	21
15	*alice*	*hong kong*	24

$s_2^{\mathcal{D}}$:

AF	*hotdog corp.*
BN	*banana ltd.*

$s_3^{\mathcal{D}}$:

12	*AF*
16	*BN*

Figure 1: Extension of sources for the example

query over the sources, and the resulting query is then evaluated at the sources. As we said before, such a strategy suffices mainly because the data integration system enjoys the single database property. Notably, the same strategy applies also in the case of sound views.

However, when the language $\mathcal{L}_{\mathcal{G}}$ used for expressing the global schema allows for integrity constraints, and the views are sound, then query processing in GAV systems (and thefore in GLAV systems) becomes more complex. Indeed, in this case, integrity constraints can in principle be used in order to overcome incompleteness of data at the sources. The following example shows that, even in the GAV approach, by taking into account integrity constraints, one can obtain answers that would be missed by simply unfolding the user query. The example is based on the relational model, but it can be easily rephrased in a different model.

Let $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$ be an SINode, where $\mathcal{G}$ is constituted by the relations

$$\begin{array}{l} \mathsf{employee}(Ecode, Ename, Ecity) \\ \mathsf{company}(Ccode, Cname) \\ \mathsf{employed}(Ecode, Ccode) \end{array}$$

and the constraints

$$\begin{array}{lcl} \mathsf{employed}[Ecode] & \subseteq & \mathsf{employee}[Ecode] \\ \mathsf{employed}[Ccode] & \subseteq & \mathsf{company}[Ccode] \end{array}$$

The source schema $\mathcal{S}$ consists of three sources. Source s_1, of arity 4, contains information about employees with their code, name, city, and date of birth. Source s_2, of arity 2, contains codes and names of companies. Finally, Source s_3, of arity 2, contains information about employment in companies. The mapping $\mathcal{M}$ is defined by

$$\begin{array}{lcl} \mathsf{employee} & \rightsquigarrow & \{\, x, y, z \mid \mathsf{s}_1(x, y, z, w) \,\} \\ \mathsf{company} & \rightsquigarrow & \{\, x, y \mid \mathsf{s}_2(x, y) \,\} \\ \mathsf{employed} & \rightsquigarrow & \{\, x, w \mid \mathsf{s}_3(x, w) \,\} \end{array}$$

Now consider the following user query q, asking for codes of employees:

$$\{\, x \mid \mathsf{employee}(x, y, z) \,\}$$

Suppose that the data stored in the source database $\mathcal{D}$ are those depicted in Figure 1: by simply unfolding q we obtain the answer $\{12, 15\}$. However, due to the integrity constraint $\mathsf{employed}[Ecode] \subseteq \mathsf{employee}[Ecode]$, we know that 16 is the code of a person, even if it does not appear in $s_1^{\mathcal{D}}$. The correct answer to q is therefore $\{12, 15, 16\}$. Observe that we do not know any value for the attributes of the employee whose $Ecode$ is 16. Given a source database $\mathcal{D}$, let us call "retrieved global database" the global database that is obtained by populating each relation r in the global schema according to the mapping, i.e., by populating r with the tuples obtained by evaluating the query that the mapping associates to q. In general, integrity constraints may be violated in the retrieved global database (e.g., the retrieved global database for the above example).

The assumption of sound views asserts that the tuples retrieved for a relation r are a subset of the tuples that the system assigns to r; therefore, we may think of completing the retrieved global

database by suitably adding tuples in order to satisfy the constraints, while still conforming to the mapping. When a constraint is violated, there are several ways of adding tuples to the retrieved global database to satisfy such a constraint. In other words, in the presence of constraints of the type illustrated in the example, in the global schema, the semantics of the SINode must be formulated in terms of a *set* of databases, instead of a single one. Since we are interested in the certain answers $q^{\mathcal{I},\mathcal{D}}$ to a query q, i.e., the tuples that satisfy q in all global databases that are legal for $\mathcal{I}$ with respect to $\mathcal{D}$, the existence of several such databasesv complicates the task of query answering.

Taking into account the above observations, the design of query processing algorithms in the context of an SINode must be carefully carried out. Indeed, we can anticipate that the query manager of an SINode will be constituted by three conceptually separate phases.

1. the query is *expanded* to take into account the integrity constraints in the global schema;
2. the atoms in the expanded query are *unfolded* according to their definition in terms of the mapping, obtaining a query expressed over the sources;
3. the expanded and unfolded query is *executed* over the sources, to produce the answer to the original query.

4 Query management in the context of different brokering agents

In the previous section, we discussed query management within a single SINode. As we said in the first section, a query agent is directed to the relevant SINodes by the brokering agents. The interaction between the query agent and the brokering agents is the subject of this section.

There are two main issues to address in this context:

1. First, given a query Q and a brokering agent B, define the mechanisms that allow B to inform the query agent about which are the SINodes under the control of B that the query agent should access in order to retrieve relevant answers to Q.

2. Second, given a query Q and a brokering agents B, define the mechanisms that allow B to inform the query agent about which brokering agents should be accessed in order to retrieve other possible answers to the query.

Issue (1) is related to the knowledge that a brokering agent has over its SINodes. In principles, there are two approaches to characterize this knowledge.

- In the first approach, the brokering agent represents the knowledge over its SINodes in terms of a virtual ontology reconciling the information stored in the underlying SINodes. In this approach, the variuos SINodes are considered as data sources under the control of the brokering agent, and a LAV mapping is used to link the SINodes and the virtual ontology held by the brokering agent.

- In the second approach, the brokering agent simply stores a set of mapping assertions between the SINodes under its control, without any explicit usage of a virtual common representation.

The decision of which approach to follow in Sewasie will be taken in the future, taking into account that the first approach is covered by the techniques developed for query management within a single SINode, and the second approach can be captured by means of the techniques described in the following for query management in the context fo different brokering agents. Therefore, issue (1) does not need further development here.

On the contrary, issue (2) will be discussed in the following. The main point here is that every brokering agent act at the same level, with no unifying structure above them. In other words, if one wants to model the data integration problem underlying the interaction between brokering

agents, one should come up with a formal framework which is of different nature with respect to the one adopted in the characterization of SINodes. Whereas in a SINode, data integration is based on the existence of the global virtual view, such a notion does not show up when trying to formalize the interconnection between brokering agents. What is needed here is a mechanism that is able, given two brokering agents, to define mappings between them, without resorting to any unifying conceptual structure.

From all the above observations, it follows that the formal framework we are seeking should follow the Peer to Peer (P2P) paradigm. Roughly speaking, each brokering agent can be considered a peer, and the interconnetion between brokering agents can be seen as mappings between peers. In the following subsection, we present a brief overview of P2P data integration systems proposed so far in the literature. In the subsequent subsection, we illustrate the foundations for a P2P characterization of the brokering agent network in Sewasie.

4.1 An overview of P2P Systems

Peer-to-peer systems offer an alternative to traditional client-server systems. In such systems, every node (peer) of the system acts as both client and server and provides part of the overall information available from an Internet-scale distributed environment. A suitabel infrastructure is adopted for mastering the complexity of the architecture. Napster [26], which made the P2P idea popular, avoids some of this complexity by employing a centralized database with references to the information items (files) on the peers. Gnutella, another well-known P2P system, has no central database, and is based on a communication-intensive search mechanism.

More recently, a Gnutella-compatible P2P system, called Gridella [21], has been proposed, which follows the so-called Peer-Grid (P-Grid) approach. Gnutella draws on research in distributed and cooperative information systems to provide decentralized, scalable data access structures. P-Grid is a virtual binary tree that distributes replication over community of peers and supports efficient search. In particular, search time and number of generated messages grow as $O(\log_2 n)$ with the number of data items n in the network. Peers in P-grid perform construction and search/update operations without any central control or global knowledge. P-Grid's search structure exhibits the following properties:

- it is completely decentralized,
- all peers serve as entry points for search,
- interactions are strictly local,
- it uses randomized algorithms for access and search,
- search is robust against node failures.

At first glance, many of the challenges in designing P2P systems seem to fall clearly under banner of the distributed systems community. However, upon closer examination, the fundamental problem in most P2P systems is the placement and retrieval of *data*. Indeed, current P2P systems focus strictly on handling semantic-free, large-granularity requests for objects by identifier (typically a name), which both limits their utility and restricts the techniques that might be employed to distribute the data. These current sharing systems are largely limited to applications in which objects are large, opaque, and atomic, and whose content is well-described by their name. Moreover, they are limited to caching, prefetching, or pushing of content at the object level, and know nothing of overlap between objects.

These limitations arise because the P2P world lacks focus on the areas of semantics, data transformation, and data relationships. Yet, these are some of the core strengths of data management, where queries, views, and integrity constraints can be used to express relationships between existing objects. Based on these considerations, data-oriented approaches to P2P have been proposed recently. for example, in the Piazza system [29], data origins serve original content, peer nodes cooperate to store materialized views and answer queries, nodes are connected

by bandwidth-constrained links and advertise their materialized views and belong to spheres of cooperation with which they share resources.

The overall cost of answering a query includes the transfer cost from the storage provider or data origin to the query evaluator, the cost of resources utilized at the query evaluator and other nodes, and the cost to transfer the results to the query initiator. The *data placement problem* is to distribute data and work so the full query workload is answered with lowest cost under the existing resource and bandwidth constraints. While a cursory glance at the data placement problem suggests many similarities with multi-query optimization in a distributed database, there are substantial differences: a P2P system has no centralized schema and no central administration. Ideally, the Piazza system will take the current query workload, find commonalities among the queries, exploit materialized views whenever cost-effective, distribute work under resource and bandwidth constraints, and determine whether certain results should be materialized for future use. In order to perform this optimization Piazza address two sub-problems:

- Propagating information about materialized view: when a query is posed, the first step is to consider whether it can be answered using the data at ”nearby” storage providers, and to evaluate the costs of doing so. This requires the query initiator to be aware of existing materialized views and properties such as location and data freshness.
- Consolidating query evaluation and data placement: A node may pose a query that cannot be evaluated with the data available from known peers. In this case, the data must be retrieved directly from the data origins.

Another data-oriented approach is described in [24], which introduces the Local Relational Model (LRM) as a data model specifically designed for P2P applications. The LRM assumes that the set of all data in P2P network consists of local (relational) databases, each with a set of acquaintances, which define the P2P network topology. For each acquaintance link, domain relations define translation rules between data items, and coordination formulas define semantic dependencies between the two databases. Two of the main goals of the data model are to allow for inconsistent databases and to support semantic interoperability in the absence of a global scheme.

Semantically, the LRM is characterized in terms of *relational spaces*: a relational space is a pair consisting of a set of databases (the peers) and domain relation which makes explicit the relations among domains of the databases. The LRM semantics is a variation of the semantic of distributed first-order logic, which itself is an extension of the Local Models Semantics, proposed in [11] . The coordination formulas that relate the contents of peer databases and define what it means for a coordination formula to be satisfied (with respect to a relational space) are used as *deductive rules*, and define a global answer to a query with respect to a relational space. The intuition is to compute the union of all the answers of the peer databases, taking into account the information carried by domain relations. This approach can serve as an example showing the need for a foundation of sound and complete implementation of a query answering mechanism in a P2P environment.

4.2 Formal P2P framework in Sewasie

We now present the main ideas for formally characterizing query management in the context of different Sewasie brokering agents. In what follows, we conceive a brokering agent B_i as a software component characterized by a network ontology G_i (expressed in a language $\mathcal{L}_{\mathcal{O}}$ over an alphabet $\mathcal{A}_{G_i}$). The term “network ontology” is used to characterize the ontology that a brokering agent exports in order to allow for mappings with other brokering agents.

A brokering agent network $\mathcal{N}$ in Sewasie is then characterized by:

1. n brokering agents $B_1, \ldots, B_n$, each one with the associated network ontology G_i.
2. A set of mapping assertions, each one relating two brokering agentsq B_i and B_j, and each one having the form

$$q_{B_i} \rightsquigarrow q_{B_j}$$

where q_{B_i} is a query over G_i, and q_{B_j} is a query over G_j, where both queries are expressed in a suitable query language $\mathcal{L}_{\mathcal{M}}$.

Intuitively, the meaning of the mapping assertion $q_{B_i} \rightsquigarrow q_{B_j}$ is that, the concept in the ontology G_j represented by q_{B_j} has q_{B_i} as a counterpart in G_i. This implies that, whenever a query agent is interested, either explicitely, or implicitely, in the answers of the query q_{B_j} posed to brokering agent B_j, it should also issue query q_{B_i} posed to brokering agent B_i. In other words, q_{B_i} is asserted to be the G_i-concept which reflects at best the G_j-concept represented by q_{B_j}.

Note that in the simplest case, q_{B_j} (resp., q_{B_i}) is an atomic concept in the corresponding network ontology. However, in the general case, mapping concepts between different ontologies requires to use queries over such ontologies. This is why we introduced the query language $\mathcal{L}_{\mathcal{M}}$ in our formalization.

Queries to $\mathcal{N}$ are posed in terms of a network ontology $\mathcal{G}_i$ (i.e., the network ontology of the brokering agent b_i), and are expressed in a query language $\mathcal{L}_{\mathcal{Q}}$ over the alphabet $\mathcal{A}_{\mathcal{G}_i}$. A query is intended to provide the specification of which data to extract from the brokering agent network $\mathcal{N}$.

In order to formally define the semantics of query answering in our framework, the crucial step is to define the formal semantics of mapping assertions in a brokering agent network $\mathcal{N}$.

Recent papers on the P2P paradigm (e.g., [29]) try to reach this goal by resorting to firt-order logic for interpreting the mapping assertions. The basic idea of this approach is to interpret a mapping assertion

$$q_{B_i} \rightsquigarrow q_{B_j}$$

as the first order formula

$$\forall \mathbf{x}\, q_{B_i}(\mathbf{x}) \rightarrow q_{B_j}(\mathbf{x})$$

This idea, although appealing from certain points of view, has several drawbacks:

- First, by considering the mapping assertions as first order formulas, we are in fact assigning to the various nodes of the network a sort of global semantics, thus completely ignoring the structure of the P2P system (in our case, the structure of the brokering agent network). In other words, the first order interpretation of a P2P system considers the various peers like homogeneous pieces of a single coherent artifact, instead of conceiving each peer as an autonomous module interacting with the others.
- Second, with the above characterization, it is very easy to show that, as soon as the query language $\mathcal{L}_{\mathcal{M}}$ has reasonable expressive power, query answering becomes undecidable. Indeed, one can easily verify that undecidability shows up if $\mathcal{L}_{\mathcal{M}}$ allows for expressing conjunctive queries. In order to overcome such problem, some authors proposes to limit the expressive power of the mapping assertions, for example avoiding cyclic references between peers.

Based on the above observations, we propose a new semantics, with the following aims:

- We want to take into account that peers in our context are to be considered autonomous sites, that exchange information. In other words, peers are modules, and the modular structure of the system should be explicitely reflected in the definition of its semantics.
- We do not want to limit a-priori the topology of the mapping assertions between the peers in the system. In particular, we do not want to impose acyclicity of assertions.
- We seek for a semantic characterization that leads to setting where query answering is decidable, and possibly, polynomially tractable.

The basic idea to meet the above goals, is to interpret a mapping assertion

$$q_{B_i} \rightsquigarrow q_{B_j}$$

as a logical statement of the form:

> whenever it is **known** by the system that α is a **certain** answer of the query q_{B_i}, then α also satisfies the expression represented by q_{B_i} in G_j.

The crucial property is that only answers that are certain (i.e., known) for B_i are propagated to B_j. From the logical point of view, this means that we are decoupling B_i and B_j in such a way that the interconnection between brokering agents does not take place at the level of the logical models of the corresponding ontologies. Rather, the relationship between B_i and B_j is established at the level of answers to queries (i.e., at the level of what is true in **all** the various models of the ontologies).

The formal development resulting from the above principles is outside the scope of the present deliverable, and will be illustrated in the next deliverable of Workpackage 3. We only anticipate the following obsevations:

- In order to formally define the notion of a brokering agent **knowing** an answer to a query, we will resort to the work by Reiter [28].
- We will adopt the language of conjunctive query both as the language $\mathcal{L}_{\mathcal{M}}$, and as the language $\mathcal{L}_{Q}$.
- We do not impose any constraints on the topology of the mapping assertions.
- Query answering in the resulting framework is not only decidable, but also polynomially tractable.

References

[1] Serge Abiteboul and Oliver Duschka. Complexity of answering queries using materialized views. In *Proc. of the 17th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS'98)*, pages 254–265, 1998.

[2] Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, 2002. To appear.

[3] D. Beneventano, S. Bergamaschi, S. Castano, A. Corni, R. Guidetti, G. Malvezzi, M. Melchiori, and M. Vincini. Information integration: the MOMIS project demonstration. In *Proc. of the 26th Int. Conf. on Very Large Data Bases (VLDB 2000)*, 2000.

[4] Alexander Borgida. Description logics in data management. *IEEE Trans. on Knowledge and Data Engineering*, 7(5):671–682, 1995.

[5] Andrea Calì, Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini. Accessing data integration systems through conceptual schemas. In *Proc. of the 20th Int. Conf. on Conceptual Modeling (ER 2001)*, pages 270–284, 2001.

[6] Andrea Calì, Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini. Data integration under integrity constraints. In *Proc. of the 14th Conf. on Advanced Information Systems Engineering (CAiSE 2002)*, volume 2348 of *Lecture Notes in Computer Science*, pages 262–279. Springer, 2002.

[7] Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini. Answering queries using views over description logics knowledge bases. In *Proc. of the 17th Nat. Conf. on Artificial Intelligence (AAAI 2000)*, pages 386–391, 2000.

[8] Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, Daniele Nardi, and Riccardo Rosati. Description logic framework for information integration. In *Proc. of the 6th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'98)*, pages 2–13, 1998.

[9] Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Moshe Y. Vardi. Answering regular path queries using views. In *Proc. of the 16th IEEE Int. Conf. on Data Engineering (ICDE 2000)*, pages 389–398, 2000.

[10] M. J. Carey, L. M. Haas, P. M. Schwarz, M. Arya, W. F. Cody, R. Fagin, M. Flickner, A. Luniewski, W. Niblack, D. Petkovic, J. Thomas, J. H. Williams, and E. L. Wimmers. Towards heterogeneous multimedia information systems: The Garlic approach. In *Proc. of the 5th Int. Workshop on Research Issues in Data Engineering – Distributed Object Management (RIDE-DOM'95)*, pages 124–131. IEEE Computer Society Press, 1995.

[11] C.Ghidini and F.Giunchiglia. Local models semantics or contextual reasoning = locality + compatibility. *Artificial Intelligence*, 127, 2001.

[12] The Sewasie Consortium. Specification of the architecture of the brokering agent. *Sewasie, Deliverable D.4.1.a, Final Version*, Feb. 2003.

[13] Oliver Duschka. *Query Planning and Optimization in Information Integration*. PhD thesis, Stanford University, 1997.

[14] Hector Garcia-Molina, Yannis Papakonstantinou, Dallan Quass, Anand Rajaraman, Yehoshua Sagiv, Jeffrey D. Ullman, Vasilis Vassalos, and Jennifer Widom. The TSIMMIS approach to mediation: Data models and languages. *J. of Intelligent Information Systems*, 8(2):117–132, 1997.

[15] Cheng Hian Goh, Stéphane Bressan, Stuart E. Madnick, and Michael D. Siegel. Context interchange: New features and formalisms for the intelligent integration of information. *ACM Trans. on Information Systems*, 17(3):270–293, 1999.

[16] Gösta Grahne and Alberto O. Mendelzon. Tableau techniques for querying information sources through global schemas. In *Proc. of the 7th Int. Conf. on Database Theory (ICDT'99)*, volume 1540 of *Lecture Notes in Computer Science*, pages 332–347. Springer, 1999.

[17] Michael Gruninger and Jintae Lee. Ontology applications and design. *Communications of the ACM*, 45(2):39–41, 2002.

[18] Jarek Gryz. Query folding with inclusion dependencies. In *Proc. of the 14th IEEE Int. Conf. on Data Engineering (ICDE'98)*, pages 126–133, 1998.

[19] Alon Y. Halevy. Answering queries using views: A survey. *Very Large Database J.*, 10(4):270–294, 2001.

[20] Richard Hull. Managing semantic heterogeneity in databases: A theoretical perspective. In *Proc. of the 16th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS'97)*, 1997.

[21] K.Aberer, M.Punceva, M.Hauswirth, and R.Schmidt. Improving data access in p2p systems. *IEEE Internet Computing*, 2002.

[22] Thomas Kirk, Alon Y. Levy, Yehoshua Sagiv, and Divesh Srivastava. The Information Manifold. In *Proceedings of the AAAI 1995 Spring Symp. on Information Gathering from Heterogeneous, Distributed Enviroments*, pages 85–91, 1995.

[23] Alon Y. Levy. Obtaining complete answers from incomplete databases. In *Proc. of the 22nd Int. Conf. on Very Large Data Bases (VLDB'96)*, pages 402–412, 1996.

[24] L.Serafini, F.Giunchiglia, J.Mylopoulos, and P.A.Bernstein. The local relational model: Model and proof theory. *Technical Report 0112-23, ITC-IRST*, 2001.

[25] Ioana Manolescu, Daniela Florescu, and Donald Kossmann. Answering XML queries on heterogeneous data sources. In *Proc. of the 27th Int. Conf. on Very Large Data Bases (VLDB 2001)*, pages 241–250, 2001.

[26] Napster. World-wide web. *www.napster.com*, 2001.

[27] University of Modena e Reggio Emilia. Specification of the general framework for the multilingual semantic enrichment processes and of the semantically emriched data stores. *Sewasie, Deliverable D.2.1, Final Version*, Feb. 2003.

[28] Raymond Reiter. What should a database know? *J. of Logic Programming*, 14:127–153, 1990.

[29] S.Gribble, A.Halevy, Z.Ives, M.Rodrig, and D.Suciu. What can databases di for peer-to-peer? *WebDB Workshop on Databases and the Web*, 2001.

[30] Jeffrey D. Ullman. Information integration using logical views. In *Proc. of the 6th Int. Conf. on Database Theory (ICDT'97)*, volume 1186 of *Lecture Notes in Computer Science*, pages 19–40. Springer, 1997.

[31] Ron van der Meyden. Logical approaches to incomplete information. In Jan Chomicki and Günter Saake, editors, *Logics for Databases and Information Systems*, pages 307–356. Kluwer Academic Publisher, 1998.

[32] Gang Zhou, Richard Hull, Roger King, and Jean-Claude Franchitti. Using object matching and materialization to integrate heterogeneous databases. In *Proc. of the 3rd Int. Conf. on Cooperative Information Systems (CoopIS'95)*, pages 4–18, 1995.