Egyptian Informatics Journal (2011) **12**, 151–163

Cairo University

Egyptian Informatics Journal

www.elsevier.com/locate/eij
www.sciencedirect.com

ORIGINAL ARTICLE

Intelligent query processing for semantic mediation of information systems

Saber Benharzallah *, **Okba Kazar**, **Guy Caplat**

Department of Computer Science, University of Mohamed Khider Biskra, 07000 Algeria, INSA de Lyon, Villeurbanne Cedex, France

Received 23 February 2011; revised 21 May 2011; accepted 20 June 2011
Available online 20 July 2011

KEYWORDS

Query answering;
Semantic mediation;
Multi-agent systems;
OWL DL

Abstract We propose an intelligent and an efficient query processing approach for semantic mediation of information systems. We propose also a generic multi agent architecture that supports our approach. Our approach focuses on the exploitation of intelligent agents for query reformulation and the use of a new technology for the semantic representation. The algorithm is self-adapted to the changes of the environment, offers a wide aptitude and solves the various data conflicts in a dynamic way; it also reformulates the query using the schema mediation method for the discovered systems and the context mediation for the other systems.

1. Introduction

Interoperability has been a basic requirement for modern information systems environment. The cooperation of systems is confronted with many problems of heterogeneities and must take in the account of the open and dynamic aspect of modern environments.

Various types of heterogeneity can be encountered cited as follows: technical, syntactic, structural and semantic heterogeneity. The resolution of semantic heterogeneity is becoming more important than before. Its types appear as: naming conflicts (taxonomic and linguistic problems), values conflicts (units and scales problems,...).

The high number of the information sources implies the increase and the diversification of the conflicts number, as well as an increase in the time of localization of relevant information. It increases also the time of transmission of the queries towards all these information sources and the time response of the information sources. Therefore, the solutions of semantic interoperability should have an intelligent processor for query processing that allows the adaptation of the environment's changes and solves the various data conflicts in a dynamic way. Each solution provides some advantages to the detriments of others. Each one of them treats just one part of the data conflicts.

* Corresponding author.
E-mail addresses: sbharz@yahoo.fr (S. Benharzallah), kazarokba@yahoo.fr (O. Kazar), guy.caplat@insa-lyon.fr (G. Caplat).

Peer review under responsibility of Faculty of Computers and Information, Cairo University.
doi:10.1016/j.eij.2011.06.003

Production and hosting by Elsevier

In this paper; we propose an intelligent and an efficient query answering the approach for semantic mediation of information systems. We propose also a generic multi agent architecture that supports our approach. Our approach focuses on the exploitation of intelligent agents for query reformulation and the use of a new technology for the semantic representation. Our algorithm is self-adapted to the changes of the environment, offers a wide aptitude and solves the various data conflicts in a dynamic way. It reformulates the query using the schema mediation method for the discovered systems (described in Section 3) and the context mediation for the other systems.

In Section 2, we present a synthesis of the various existing approaches. Section 3 describe our approach. In Section 4 we present the basic concepts. Then Sections 5 and 6 describe the various types of agents and the query processing. Section 7 presents the technical aspects and prototype implementation.

2. Related works

As the query processing problem in distributed systems has been discussed in traditional databases and Semantic Web, two possible orientations have been proposed: the integration guided by the sources (schema mediation), and the integration guided by the queries (context mediation) [1,4–6,8,10,11,21].

The schema mediation is a direct extension of the federate approach. Data conflicts are statically solved. In the schema mediation; the mediator should be associated with a knowledge set (mapping rules) for locating the data sources. The query processing follows an execution plan established by rules which determine the relevant data in order to treat a query (static resolution of queries). It requires a pre-knowledge on the systems participating in the cooperation. The mediator's role is to divide (according to the global schema) the user query in several sub-queries supported by the sources and gathers the results. The global schema is generally specified by object, logic, XML or OWL interfaces [3,5,17,22,24]. In all these works, the objective is to build a global schema which integrates all the local schemas. When one operates in an evolutionary world where sources can evolve all the time, the elaboration of a global schema is a difficult task. It would be necessary to be able to reconstruct the integrated schema each time a new source is considered or each time an actual source makes a number of changes [4]. Generally, the time response of the queries of this approach is better than the context mediation which requires much time (it uses the semantic reconciliation). In this approach; the transparency (is to give the illusion to the users whom they interact with a local system, central and homogeneous) is assured. The degree of automation of the resolution of the data conflicts is weak, and the scalability (the system effectiveness should be not degraded and the query processing remains independent of the addition or the suppression of systems in a given architecture) and evolutionarity (to control the update, the remove and the addition of information systems) are less respected compared to the context mediation.

Many works are dedicated to the proposition of automatic approaches for schemas/ontologies integration [30,31]. The schemas mapping notion have been particularly investigated in many studies, therefore it leads to the elaboration of several systems such as DIKE [7], COMA [13], CUPID [14]. It is possible to find analyses and comparisons of such systems in [18]. Several ontologies based approaches for integration of information were suggested [46]. In [4,20] survey of this subject is presented. Among the many drawbacks of these works is that they do not describe the integration process in a complete way; they always use assumptions like pre-existence mappings [23,33] from a part, and from another part, they provide methods to calculate mappings between general or specific ontologies [30] and they do not indicate how to really exploit it for automatic integration or for the query reformulation [22,33].

In [3,21] the authors have proposed an extended schema mediation named DILEMMA based on the static resolution of queries. The mediation is ensured by a couple mediator/wrapper and a knowledge base associated with each system that takes part in the cooperation. The mediator comprises a queries processor and a facilitator. This approach provide a better transparency and makes it possible to solve the semantic values conflicts, but in a priori manner. The automation degree of the resolution of the data conflicts is enhanced compared to the schema mediation. This later always involves the recourse of an expert of the domain. It has a low capacity to treat evolutionarity and the scalability.

The role of the mediator in the context mediation approach is to identify, locate, transform and integrate the relevant data according to semantics associated with a query [3,21]. The resolution of data conflicts is dynamic and does not require the definition of a mediation schema. The user's queries are generally formulated in terms of ontologies. The data are integrated dynamically according to the semantic information contained in the description of the contexts. This approach provides a best evolutionarity of the local sources and the automation degree of the resolution of the data conflicts is better compared to schema mediation. Two categories of context mediation are defined: the single domain approach SIMS [9], COIN [10] working on a single domain where all the contexts are defined by using a universal of consensual speech. The scalability and evolutionarity are respected but remains limited by the unicity of the domain. Multi-domains approaches Infosleuth [11], Observer [12] they use various means to represent and connect heterogeneous semantic domain: ontologies, hierarchy of ontologies and method of statistical analysis.

In the context mediation approach the data conflicts are dynamically solved during the execution of the queries (dynamic query resolution), allowing the best evolution of the local sources and the automation degree is enhanced compared to the schema mediation, this to the detriment of time response of the queries (it uses the semantic reconciliation). Concerning the semantic conflicts, the majority of the projects solve only the taxonomic conflicts (Coin [10]). The resolution of the values conflicts is either guided by the user (Infosleuth [11]), or unsolved in the majority of cases (Observer [12,28]).

The agent paradigm gives a new insight for the systems nature development such as: complex, heterogeneous, distributed and/or autonomous [15,34,35,38,47]. Several works of semantic interoperability use the agent paradigm [11,16,25,29,32].

Infosleuth project [11] is used to implement a set of cooperative agents which discover, integrate and present information according to the user or application needs for which they produce a simple and coherent interface. The Infosleuth's architecture project consists of a set of collaborative agents, communicating with each other. Users express their queries on a specific ontology using KIF (Knowledge Interchange Format) and SQL. The queries are dispatched to the specialized agents (agent broker, ontological, planner,...) to retrieve data on distributed sources. The resolution of many semantic conflicts remains guided by the user [3]. They use specialized agents seen as threads which are widely different from the usual definition of the cognitive agent given in the distributed artificial intelligence.

In [25], the authors propose a multi-agent system to achieve semantic interoperability and to resolve semantic conflicts related to evolutive ontologies domain. In this approach, the query processing and the validation of the mappings are completely related to the users [29]. They propose an agent based intelligent meta-search and recommendation system for products through consideration of multiple attributes by using ontology mapping and Web services. This framework is intended for an electronic commerce domain.

3. Approach description

Our objective is to realize a semantic mediation having the following characteristics:

- Give permission to the system which provides its context of application to find the information systems and information shared by those systems. This information is integrated dynamically for the system for which it can use them transparently.
- To ensure the advantages of the context and schema mediation, and to avoid their disadvantages. Our approach focuses on the dynamic change of the mediation system, of the context mediation to the schema mediation. This change is done by the use of the intelligent agents, in order to ensure the open aspect of the context mediation (high automation degree of the resolution of the data conflicts,...) and the robustness of the schema mediation (the formulation of the queries in schema terms,...).
- To solve the majority of the semantic conflicts by using new technologies for semantic representation, and by respecting a high automation degree of the query processing. The query processing algorithm reformulates the query by using the schema mediation method for the systems discovered and the context mediation for the other systems. So, our query processing approach is self-adapted to the changes of the environment.

Thus our approach is considerable as intelligent and as efficient. Our query processing approach is self-adapted to the changes of the environment; it is also focuses on the dynamic change of the mediation system, of the context mediation to the schema mediation. Indeed, the computational complexity of the scheme mediation approach is less complex than the computational complexity of the context mediation approach (described in Section 6).

3.1. The philosophy of our approach

Our architecture is divided into two levels (Fig. 1): physical entities level and the agent level. The first level includes especially existing information systems (Iss), including legacy systems. These systems are developed using conventional technologies such as databases management systems, they can be relational, object, XML,... Originally, these systems are designed to meet local needs and not necessarily work together.

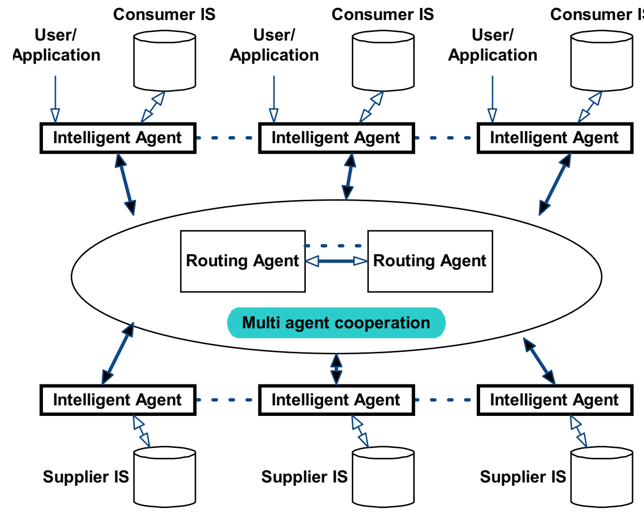


Figure 1 Generic architecture based agent for context and schema mediation.

The second agent level is designed on the top of existing physical systems. There are two types of agents: intelligent agents (IAs) and routing agents (RAs). An IA is an intermediary between an information system and the semantic mediation environment. An information system can play the role of information supplier and/or consumer. So an IA can play the role of IA supplier (IAS) and/or consumer (IAC). In other words, if the AI asks the query, so in this case is called AIC, if an another agent asked the IA to run a query in this case it is called AIF.

The routing agents play an important role in the GAACSM architecture. They assume context mediation, and organize intelligent agents in near domains semantically, it is ensured using a semantic proximity which allows to separate the semantic interoperable information systems in segments (or groups, or set of domains). The agents belonging to the same group are considered as near agents semantically (described in Section 5.2), we use this segmentation to avoid the useless communications between agents, and to improve our query processing approach.

The cooperation suggested in our solution is based on:

– A preliminary construction of information before its integration in the architecture system.
– The static and dynamics query resolution.

The integration phase of a new information system (IS) in our proposed mediation system begins with the creation of an IA and continues with the fastening of this last to a routing agent (RA) which is *nearest semantically*, Algorithm 5.

Before creating an IA, we've to create its knowledge base (KB). An IA is an intermediary between an information system and the semantic mediation environment. Mainly, the KB of an IA contains: the context of its local information system, the name of the domain, the ontology which describes the name of the domain and the ontology of the semantic conflicts values (OSCV). This information makes it possible to prepare the IA to the semantic mediation.

The new IA integrated into the system of mediation applies the Contract Net protocol and sends an invitation describing its domain. The RAs receiving the call and provide their ability (*semantic proximity rate*, Algorithm 4). As soon as, the IA receives answers from all RAs, then it evaluates these rates, and makes its choice on a RA which is the nearest semantically (Algorithm 5). The chosen RA adds the previous IA to its net contacts.

Our approach does not use a global schema or some predefined mappings. Users interrogate the consuming system (the queries formulated in term of the consuming schema).

At the beginning, the intelligent agent consuming (IAC) applies the dynamic query resolution protocol (context mediation, Section 6.2) because it does not have information on the suppliers systems. This protocol is applied via the RA which is the nearest semantically with the IAC. During the dynamic evaluation of the query, the intelligent agent suppliers (IASs) update their histories and add information (mapping between terms of query ontology and their ontologies) to facilitate their dynamic integration with the IAC.

Each IAS replies with results, the RA updates its KB and reorders the list of IASs that are the most important to previous IAC (in other words; the IASs which contain results are at the head of the list). If no IAS replies, the RA sends the query to other RAs. If there are replies, the RA adds the IASs of other RAs to its KB (autoreorganization).

During the operation of the mediation system, the IAC applies the protocol to discover suppliers which are the nearest semantically to its domain, and to integrate them dynamically in order to use them in the schema mediation. For this aim, it cooperates with the RA. Indeed; the RA updates its KB during its communication with the other agents. Particularly, its KB contains for each IA an ordered list of its IASs which are not discovered yet. These IASs should be near semantically to it. The first IAS in the list is the one which has largest number of responses of IA. After that the first IAS becomes the next supplier solicited to the following dynamic integration done by IA.

After the dynamic integration, the IAC updates its knowledge base by mapping rules (Algorithm 3), and considers the IAS as discouvred system.

During the operation of the system, the IAC discovers some suppliers and adapts itself with the environment. So, to treat a query two protocols should be applied: the static query resolution protocol (Section 6.1) is adopted for the discovered systems and the dynamic query resolution (Section 6.2) for other systems (Algorithm 6).

4. Basic concepts of the our approach

In what follows, we present a cooperation scenario which will be used throughout this paper.

4.1. Cooperation scenario

In this section, we describe an interoperability example between heterogeneous systems. A given company wishes to provide an information service regarding the concerts of various artists (extension of the example cited in [44]) from the world. We chose this example for reasons of simplification.

The schema of the consuming system is as follows:

Class (CS)
FunctionalProperty (nbC domain (CS) range (xsd:integer))
DatatypeProperty (artistN domain (CS) range (xsd:string))
DatatypeProperty (dateC domain (CS) range (xsd:date))
DatatypeProperty (Pfree domain (CS) range (xsd:integer))
DatatypeProperty (Psold domain (CS) range (xsd:integer))
DatatypeProperty (Pprice domain (CS) range (xsd:float))

nbC (integer): number identifying a concert, artistN (string): Name of artist, dateC (date): Date of concert, Pfree (integer): number of a free places, Psold (integer): number of sold places, Pprice (float): price of a place (Euro).

The schema of the supplier system 1 is given below:

Class (SS1)
Class (Place)
FunctionalProperty (id domain (SS1) range (xsd:integer))
DatatypeProperty (nam domain (SS1) range (xsd:string))
DatatypeProperty (seance domain (SS1) range (xsd:date))
ObjectProperty (EidPlc domain (SS1) range (Place))
DatatypeProperty (ticket domain (SS1) range (xsd:float))
FunctionalProperty (idplc domain (Place) range (xsd:integer))
DatatypeProperty (nbP domain (Place) range (xsd:integer))
DatatypeProperty (totP domain (SS1) range (xsd:integer))
FunctionalProperty (id domain (SS1) range (xsd:integer))

id (integer): number identifying a concert, nam (string): name of artist, seance (date): date of a seance, Eidplc (integer): an identifier reference to the relation Place-idplc, ticket (float): price of a place (Dinars), idplc (integer): identifier identifies nbP and totP, nbP (integer): number of a free places, totP (integer): number of total places.

The schema of the supplier system 2 is the following:

Class (SS2)
FunctionalProperty (nomCons domain (SS2) range (xsd:integer))
DatatypeProperty (NamArtist domain (SS2) range (xsd:string))
DatatypeProperty (ConsDate domain (SS2) range (xsd:date))
DatatypeProperty (soldP domain (SS2) range (xsd:integer))
DatatypeProperty (totalP domain (SS2) range (xsd:integer))
DatatypeProperty (Tprice domain (SS2) range (xsd:float))

numCons: number identifying a concert, NamArtist: Name of artist, ConsDate: Date of seance, soldP: number of a sold places, totalP: Number of total places, Tprice: Price of a place (Dollars).

4.2. Semantic representation of the application domain of agents

In order to facilitate and to automate the integration of the new IA in our semantic mediation system, we propose to describe the name of the application domain of each IA by using ontology.

Definition 1. Description of the application domain (or ontology of the application domain). It is the set of all terms organized in the form of an OWL DL ontology. It describes the *name* of the application domain.

Example 1. The following example assumes the existence of a domain name called *University*. The construction of an OWL DL ontology corresponding to this domain's name is the following (part of ontology):

< owl:Class rdf:ID = "University" >
< rdfs:subClassOf rdf:resource = "#academicWorld"/ >
< rdfs:label > University < /rdfs:label >
< rdfs:label > educational institution < /rdfs:label >
< rdfs:comment >
a large and diverse institution of higher learning created to educate for life and for a profession and to grant degrees
< /rdfs:comment >
< /owl:Class >
< owl:Class rdf:ID = "academicWorld" >
< rdfs:label > academic World < /rdfs:label >
< /owl:Class >
...
< rdfs:label > ... < /rdfs:label >
< rdfs:comment > ... < /rdfs:comment >
< /owl:Class >

Definition 2. Semantic proximity. It can be seen as a measure between two OWL DL ontologies belonging to different application domains. Given two application domainsd_1, $d_2$and their ontologies $O_{d1}$and O_{d2} respectively. The semantic proximity is a function SemProx:$O_{d1} \times O_{d2} \rightarrow [0,1]$. This function calculates the semantic proximity between the two names $d_1$and d_2, by using both ontologies O_{d1} and O_{d2}.

The algorithm ASCO2 is used to calculate this proximity [36,37].

Both Definitions 1 and 2 will be used so that an IA determines the RA nearest semantically (Section 5.2).

4.3. Semantic mediation model

Both sub sections reflect the necessary steps to model completely the shared information. Fig. 2 summarizes the necessary steps to model the shared information.

4.3.1. Representation of the semantic conflicts of values

The resolution of the semantic conflicts of values requires the recourse to another type of ontology. This ontology (OSCV) classifies the semantic conflicts of values; it is composed of hierarchy of RDF classes which allow distinguishing the data values.

Definition 3. OSCV Ontology. An ontology for the classification of semantic conflicts of values is expressed by a tuple (OSCV, CV, CI, RC), it is defined as a set of many concepts, instances and their interrelationships, where OSCV is the root vertex in graph RDF, CV is a distinct set of virtual concepts, CI is a distinct set of instanceable concepts. RC refers to the sibling relationships on CV and CI. (An example of OSCV ontology is presented in Fig. 3.)

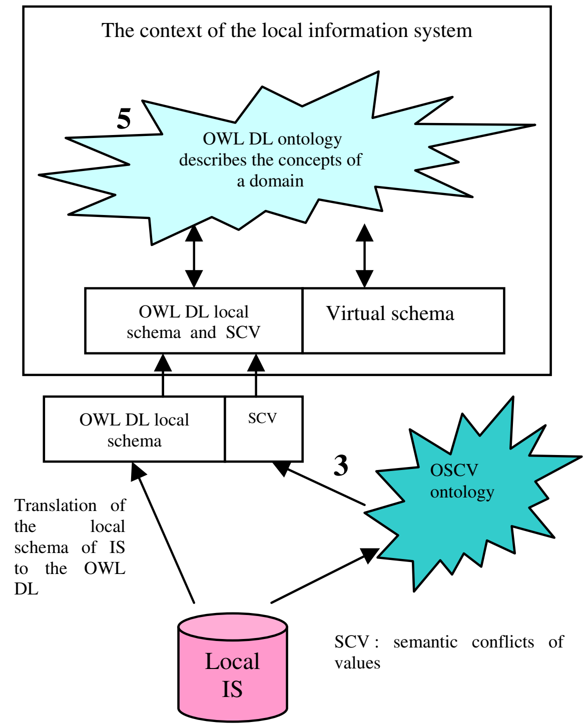


Figure 2 Necessary steps required for the modeling of shared information.

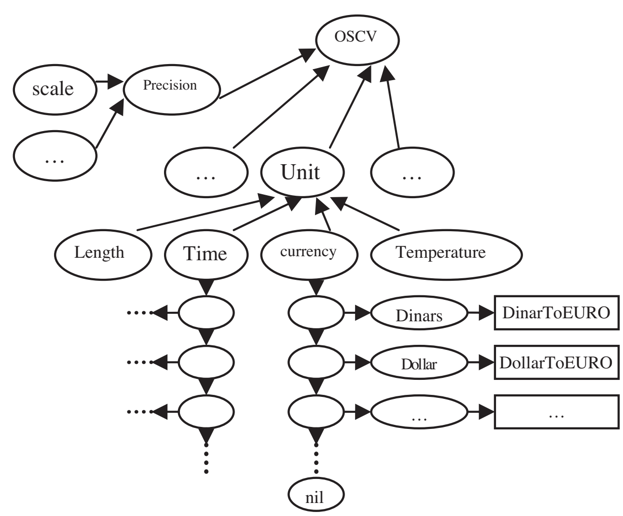


Figure 3 An example of OSCV ontology.

OSCV Ontology is gradually built and enriched by the addition of a new IA to the mediation system. It is considered as a KB's element of an IA. The addition of a new IA to the mediation system can create new semantic conflicts of values (SCV), of this fact; an update of the OSCV ontology is necessary, in other words; the new IA sends the new semantic conflicts of values to all IAs which take part in the mediation system. The other IAs takes into account new semantic conflicts of values and updates their OSCV ontologies.

4.3.2. Schema, ontology and context

Our approach uses the OWL DL [19] as common data model. The OWL DL enriches the RDF Schemas model by defining a rich vocabulary to the description of complex ontologies. So, it is more expressive than RDF and RDFS which have some insufficiency of expressivity because of their dependence only on the definition of the relations between objects by assertions. OWL DL brings also a better integration, evolution, division and easier inference of ontologies [19].

To build an ontology from a schema; we propose the following steps: (a) we use the schema to extract the concepts and the relations between them, in other words; in finding the semantic organization of the various concepts (used in the schema) and the relation between them (initial construction). (b) We add the synonyms and the antonyms of each name of class in *'label'*, (c) We add comments on the name of classes by using *'comment'*, (d) We add for each name of a class its sub concepts, its super concepts and its class's sisters.

The construction of this ontology is closely related to the context of the application domain of the information system.

Example 2. The following example indicates the schema ontologies of the consuming, supplier 1 and 2 systems built by using the preceding steps (it is a concise representation, Figs. 4–6).

The b, c, d information are represented by:

The ontology corresponding to the schema supplier 1 is given in Fig. 5.

The ontology corresponding to the schema supplier 2 is given in Fig. 6.

Definition 4. Clarification of the semantic conflicts of values. It focuses on the use of OSCV ontology to clarify or represent the semantic conflicts of values in an OWL DL schema.

This operation allows making extension of the data model of local information system by clarifying the semantic conflicts of values in order to facilitate their detections and their resolutions

Example 3. The following example shows how to use this ontology to clarify the semantic conflicts of values related to two attributes: dateC and Pprice of OWL DL schema of the consuming system.

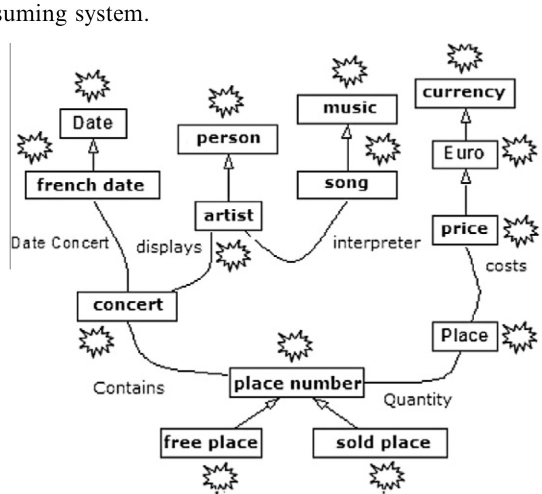


Figure 4 Ontology of consuming system.

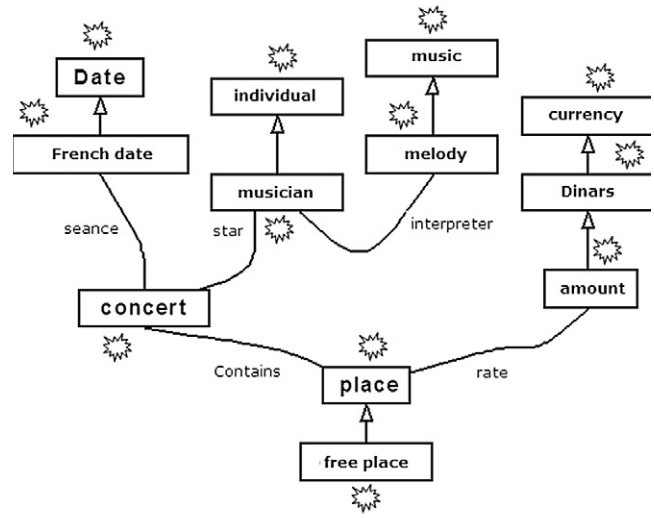


Figure 5 Ontology of supplier 1 system.

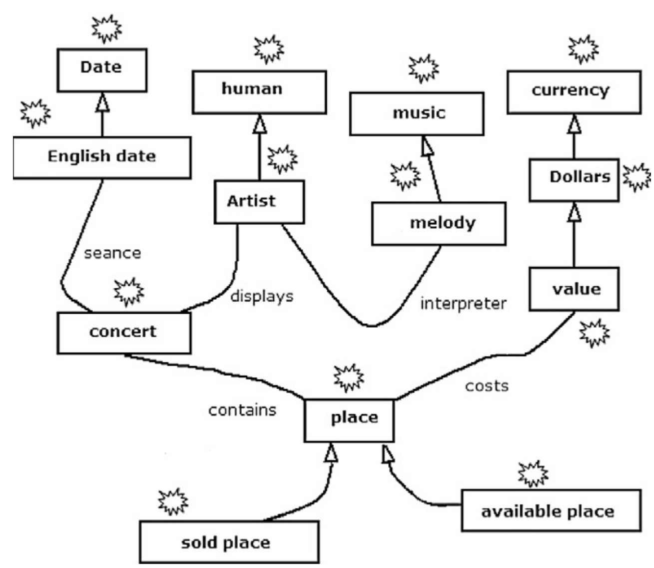


Figure 6 Ontology of supplier 2 system.

```
< owl:DatatypeProperty rdf:ID = "dateC" >
< rdfs:domain rdf:resource = "#CS"/ >
< rdfs:range rdf:resource = "& xsd;date"/ >
< OSCV: Date > French < OSCV: Date / >
< /owl:DatatypeProperty >
< owl:DatatypeProperty rdf:ID = "Pprice" >
< rdfs:domain rdf:resource = "#CS"/ >
< rdfs:range rdf:resource = "& xsd;float"/ >
< OSCV: Currency > Dollar < OSCV: Currency / >
< /owl:DatatypeProperty >
```

Definition 5. Schema-ontology mapping. Given a schema S and its ontology O. a schema-ontology mapping is expressed by the function:

MSO : S → O

x → e

We can see the elements of a schema S as individuals of the classes defined in ontology O.

Example 4. Mappings MSO between the consuming schema and its ontology are the following:

< Concert rdf:ID = ”nbC” >
< /Concert >
< Concert rdf:ID = ”CS” >
< /Concert >
< Artist rdf:ID = ”artistN” >
< /Artist >
< Date rdf:ID = ”dateC” >
< /Date >
< freeplace rdf:ID = ”Pfree” >
< / freeplace >
< soldplace rdf:ID = ”Psold” >
< / soldplace >
< Price rdf:ID = ”Pprice” >
< /Price >

Algorithm 1 Sim(e1,e2)

Require: ontology O_1 *and* O_2 , $e1 \in O_1$, $e2 \in O_2$

1: Calculation SimN of e1,e2,

2: Calculation SimC of e1,e2,

3: Calculation SimV of e1,e2,

4: Calculation SimR of e1,e2,

5 : $SimTer(e1, e2) = \alpha_1 \times SimN + \alpha_2 \times SimC$

6 : $SimStruc(e1, e2) \leftarrow \beta_1 \times SimV + \beta_2 \times SimR$

7 : $Sim(e1, e2) \leftarrow \alpha \times SimTer + \beta \times SimStruc$

End

Figure 7 Semantic similarity algorithm.

Definition 6. Context. It describes the assumptions, the explicit information of definition and use of a data. In our approach, the context is defined by (S, SCV, O, MSO) such as: S is a schema, SCV is a semantic conflicts of values, O defines an ontology and MSO is a schema-ontology mapping.

Definition 7. Query language. We adapted the language defined in [2] as a query language in our architecture. Given L the set of individuals and values belonging to OWL DL data types. Given V the set of variables disjoint from those of L. A query Q_i in ontology O_i is of the form $Q^i_C \wedge Q^i_P$, where

- Q^i_C is a conjunction of $C^i(x)$ where $C^i \in C$ and $x \in L \cup V$.
- Q^i_P is a conjunction of $P^i(x, y)$ where $P^i \in P$ and $x, y \in L \cup V$.

Example 5. This query is formulated in terms of the consuming schema.

$Q = CS(x) \wedge \text{artistN}(x, \text{“artist1”}) \wedge \text{dateC}(x, y)$

This means the knowledge of the date or the dates of the concerts of the artist “artist1”.

4.4. Basic definitions

In the discovery of mappings between two agents (IAC and IAS), it is necessary to compare the ontology of the IAC with that of the IAS. The entities of two ontologies are compared using a semantic similarity. In this section, we define our semantic similarity. The definitions necessary for GAACSM architecture will be also presented.

Definition 8. Semantic similarity. The calculation of the semantic similarity between the two concepts is calculated from the elementary calculations of similarity which take into account the various elements of the environment of a concept in its domain. The various adopted measures are: the terminology of the concept and environment in which the concept is located. These measurements are selected from a deep study of the various similarities measures [1,36,34] and from the definition of an ontology of schema in GAACSM architecture. Our algorithm which calculates the semantic distance between two elements e1, e2 is as follows (Fig. 7):oú:$\alpha \in [0,1]$, $\beta \in [0,1]$, $\alpha_1 \in [0,1]$, $\alpha_2 \in [0,1]$, $\beta_1 \in [0,1]$ and $\beta_2 \in [0,1]$. SimTer: terminological similarity. SimStruc: structural similarity. SimN: Similarity of names using their synonyms and antonyms. SimC: Comments similarity of the two concepts. SimV: Structural similarity vicinity (Our approach is based on the assumption that if the neighbors of two classes are similar, these two classes are also considered as similar). SimR: Roles similarity. (The roles are the links between two OWL DL classes.)

Definition 9. Comparison of two ontologies. The comparison of two ontologies, belonging to different IAs, The comparison is defined by the Comp function as follows: Comp: $O \rightarrow O'$ such as Comp(e1) = e′1 if Sim(e1,e′1) > tr where O and O’ are two ontologies to be compared, tr indicates a minimal level of similarity belonging to the interval [0,1], e1 ∈ O and e′1 ∈ O′.

Definition 10. Sub schema Adaptation of an IA.

– Given two intelligent agents A, B.
– Given the schema S_a, the ontology O_a of A, and the ontology O_b of the agent B.
– Given the function Comp: $O_a \rightarrow O_b$ the comparison between two ontologies O_a and O_b of A and B respectively.
– Given CO_{ab} the set of the elements e ∈ O_a, such that Comp(e) = e′ and Sim(e, e′) > tr with e′ ∈ O_b.
– Given a sub-schema Ss_a the set of elements x ∈ S_a such as MSO(x) = e with e ∈ CO_{ab}.

The adaptation of the sub schema Ss_a of S_a(of the agent A) on the ontology O_b of the agent B is the function:

$\text{Adapt} : Ss_a \rightarrow O_b$

$X \rightarrow e'$

with: $X \in Ss_a$, $e' \in O_b$ where there exist $e \in CO_{ab}$ such that Comp(e) = e′ and Sim(e, e′) > tr, MSO(x) = e.

Definition 11. Semantic enrichment of a query. Given the context C represented by (S, CSV, O, MSO) and $Q = Q^i_C \wedge Q^i_P$, a query formulated in term of the schema S.

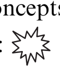
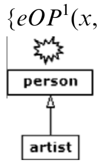
The semantic enrichment of this query, by using the ontology O, is defined by the following rules:

(1) Find using the function MSO, the equivalent classes of $C^i(x)$ and $P^i(x, y)$ of the query Q^i_C and Q^i_P respectively in the ontology O. They are noted by $OC^i(x)$ and $OP^i(x, y)$ respectively.
(2) Find by using the subsumption relation, the ancestors classes of each class of $OC^i(x)$ and $OP^i(x, y)$. They are noted by $pOC^i(x)$and$pOP^i(x, y)$ respectively.
(3) Find by using the subsumption relation, the sub classes of each class of $OC^i(x)$ and $OP^i(x, y)$. They are noted by $cOC^i(x)$ and $cOP^i(x, y)$ respectively.
(4) Find by using the equivalent relation, the equivalent classes of each class of $OC^i(x)$ and $OP^i(x, y)$. They are noted by $eOC^i(x)$ and $eOP^i(x, y)$ respectively.
(5) We clarify, by using the schema S, the semantic conflicts of values which exist in the query Q. This information is noted by $csvQ$.

A query Q enriched semantically is composed of ($Q^i_C \wedge Q^i_P$, ,$OC^i(x)$, $\{eOC^i(x)\}$, $\{pOC^i(x)\}$, $\{cOC^i(x)\}$, $OP^i(x, y)$, $\{eOP^i(x, y)\}$, $\{pOP^i(x, y)\}$, $\{cOP^i(x, y)\}$, csQ). This enrichment is called *query ontology*.

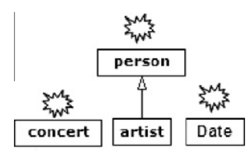
Example 6. Given the following query formulated in terms of the consuming schema $Q = CS(x) \wedge \text{artistN}(x, \text{“artist1”}) \wedge \text{dateC}(x, y)$. The semantic enrichment of the query is as follows:

We have $Q = Q^i_C \wedge Q^i_P$,

– The Correspondent of the concept $CS(x)$, based on the function MSO, is the concept $OC^i(x) = concert$.
– Concepts $\{eOC^i(x)\}$,$\{pOC^i(x)\}$,$\{cOC^i(x)\}$ are represented by: 
– The Correspondent of the concept $OP^{i1}(x, y) =$ artistN(x, “artist1”), based on the function MSO, is the concept *Artist*.
– Concepts $\{eOP^1(x, y)\}$,$\{pOP^1(x, y)\}$,$\{cOP^1(x, y)\}$ are represented by 

– The Correspondent of the concept $OP^2(x, y)$ = dateC(x,y), based on the function MSO, is the concept *Date*.
– Concepts $\{eOP^2(x, y)\}$,$\{pOP^2(x, y)\}$,$\{cOP^2(x, y)\}$ are represented by 

The semantic enrichment of the query (query ontology) $SC(x) \wedge$ artistN(x, “artiste1”) $\wedge$ dateC(x,y) is the ontology:



And $csvQ$ = {dateC : < OSCV: Date > French date < OSCV: Date / > }

Algorithm 2

Require: query ontology O_Q *of IAC and ontology* O_{IAS} of IAS

1: Calculation of the function *Comp:* $O_Q \rightarrow O_{IAS}$

2: Calculation of the set CO_{QIAS}

3: Calculation of the sub-schema Ss_Q

4 : Calculation of the function *Adapt* : $Ss_Q \rightarrow O_{IAS}$

5: /*Evaluate of the semantic query *Adapt*(Ss_Q) */

6: For each $X \in Ss_Q$ and $e \in Adapt(X)$ do

7: Search $Y \in S_{IAS}$ where: MSO(Y)=e

8: If Y exists then to replace X by Y in $Q^i_C \wedge Q^i_P$ ($Q^i_C \wedge Q^i_P \in O_Q$)

9: Endfor

10: to eliminate the semantic conflicts of values if they exist by using OSCV ontology

11: the quey which will be to evaluate in IAS is $Q^i_C \wedge Q^i_P$

6: End

Figure 8 Algorithm 2. Semantic evaluation of a query ontology.

Definition 12. Semantic evaluation of a query ontology. The semantic evaluation of a query enriched semantically (query ontology) $O_Q = (Q^i_C \wedge Q^i_P, ,OC^i(x), \{eOC^i(x)\}, \{pOC^i(x)\}, \{cOC^i(x)\}, OP^i(x, y), \{eOP^i(x, y)\}, \{pOP^i(x, y)\}, \{cOP^i(x, y)\}, csQ)$ is defined by Algorithm 2 (Fig. 8):

Example 7. Given the previous query

$Q = CS(x) \wedge \text{artistN}(x, \text{“artist1”}) \wedge \text{dateC}(x, y)$

The semantic evaluation of its query ontology, on the source of supplier 1 is done by the application of Algorithm 2. The steps are as follows:

– Calculation of the similarities between the query ontology and the ontology of supplier 1
– Calculation of the set CO_{QIAS} = {Concert, Artist, Person, Date,...}
– Calculation of the sub schema Ss_Q = {$CS(x)$,artistN(x, “artist1”),dateC(x,y)}
– Calculation of the function Adapt :$Ss_Q \rightarrow O_{IAS}$. Its values are :{Adapt(SC(x)) = Concert, Adapt(artistN(x,“artist1”)) = Musician, Adapt(dateC(x,y)) = Date}
– Semantic evaluation of $Adapt(Ss_Q)$, which requires the calculation of the function MSO reverse. Hence: {MSO^{-1}(Concert) = SS1, MSO^{-1} (Musician) = nam, MSO^{-1}(Date) = seance}.
– Concerning the semantic conflicts of values, the attribute *seance* uses the same format like *dateC*, else it is necessary

to take into account the change of the results and to convert the format using the OSCV ontology (a transformation function).

Finally, the query which will be carried out on the level of the source of supplier 1 is as follows:

$Q = SS1(x) \wedge nam(x, \text{“artist1”}) \wedge seance(x, y)$.

Definition 13. Mapping rules. A schema mapping is a triplet (S1, S2, M) [2], where: S1 is the source schema; S2 is the target schema; M the mapping between S1 and S2, i.e. a set of assertions $q_s \mapsto q_T$, with q_s and q_T are conjunctive queries over S1 and S2, respectively, having the same set of distinguished variables x, and $\mapsto \in \{\subseteq, \supseteq, \equiv\}$.

In our approach S1 is the schema of an IAC, and S2 is the schema of an IAS. Our approach generates automatically the mappings between the two agents (IAC and IAS). For this reason, the algorithm which describe the comparison of ontologies of the schemas of IAC and IAS is in the following figure (Fig. 9).

Example 8. The following example generates mappings between the consuming schema and the supplier schema 2 by using Algorithm 3. The steps are as follows:

– Calculation of the comparaison function *Comp* between the consuming ontology and supplier ontology 2, where : *Comp*(Date) = {Date, Englishdate}, *Comp*(song) = melody, *Comp*(price) = value, *Comp*(place) = {place, soldPlace, availablePlace}, *Comp*(PlaceNumber) = {place, soldPlace, AvailablePlace},...
– Calculation of CO_{IACIAS}, having the following set of value sCO_{IACIAS} = {Date, song, price, concert, ...}
– Calculation of Ss_{IAC} and Ss_{IAS}: Ss_{IAC} = {nbC, artistN, dateC, Pfree, Psold, Pprice}, Ss_{IAS} = {numCons, NamArtist, ConsDate, soldP, totalP, Tprice}

Algorithm 3. Generation mapping rules
Require: a schema, ontology and schema-ontology mapping of IAC and IAS
Calculation of the function *Comp:* $O_{IAC} \rightarrow O_{IAS}$
Calculation of the set CO_{IACIAS}
Calculation of the sub-schema Ss_{IAC}, Ss_{IAS}
Calculation of the function *Adapt:* $Ss_{IAC} \rightarrow O_{IAS}$
/* Mappings generation */
For each $X \in Ss_{IAC}$ do
Search $Y \in Ss_{IAS}$ such as:
If *Adapt*(*X*) =*e1* and MSO_{IAS} (*Y*) =*e2* and *e1, e2* $\in$
O_{IAS} and if e1 $\equiv$ *e2* then generate the rule *X* $\equiv$ *Y* else
if e1 $\subseteq$ *e2* then generate the rule *X* $\subseteq$ *Y* else if e1 $\supseteq$ *e2*
then generate the rule *X* $\supseteq$ *Y*
EndFor
End

Figure 9 Generation mapping rules.

– Calculation of the function *Adapt*. *Adapt*(nbC) = concert, *Adapt*(artistN) = Artist, *Adapt*(dateC) = {Date, EnglishDate}, *Adapt*(Pfree) = {AvailablePlace, place}, *Adapt*(Psold) = {FreePlace, place}, *Adapt*(Pprice) = value.
– The generation of the mappings: *dapt*(nbC) = concert, and MSO_{IAS}(numCons) = concert, then the mapping « nbC ≡ numCons » is generated. In the same way, by applying the algorithm for the others, as a result the following mappings are generated: artistN ≡ NamArtist, dateC ≡ ConsDate, Pfree ⊆ totalP, Psold ≡ soldP, Pprice ≡ Tprice.

The generated mappings will be used to reformulate the queries written in terms of the consuming schema and respecting the elimination the semantic conflicts of values, by using ontology OSCV and information available about the system supplier 2 (its context). For example, given the query $Q1 = CS(x) \wedge artistN(x, \text{“artist1”}) \wedge \text{Pprice}(x, y)$ reformulated by our approach to the query. $Q1' = SS2(x) \wedge NamArtist(x, \text{“artist1”}) \wedge \text{Tprice}(x, y)$. Such reformulation can be considered as an equivalent one: $Q1 \equiv Q1'$.

The query $Q2 = CS(x) \wedge nbC(x, y) \wedge artistN(x, \text{“artist1”}) \wedge Pfree(x, z)$ reformulated to the query $Q2' = SS2(x) \wedge numCons(x, y) \wedge NamArtist(x, \text{“artist1”}) \wedge totalP(x, z)$.

It is the minimal reformulation containing $Q2$. There is a lack of information at the level of the system supplier.

Definition 14. Query reformulation. Let Q_i be a query in schema S_i and Q_j be a query in schema $S_j$described by classes and properties in the mapping Mij.

- Q_j is an equivalent reformulation of Q_i if $Q_j \subseteq Q_i$and $Q_i \subseteq Q_j$, which is noted by$Q_j \equiv Q_i$.
- Q_j is a minimally-containing reformulation of Q_i if $Q_i \subseteq Q_j$ and there is no other query Q'_j such that $Q_i \subseteq Q'_j$ and $Q'_j \subseteq Q_j$.
- Q_j is a maximally-contained reformulation of Q_i if $Q_j \subseteq Q_i$ and there is no other query Q'_j such that $Q_j \subseteq Q'_j$ and $Q'_j \subseteq Q_i$.

To find the approximate query reformulation we use the mapping rules M (Definition 13), we substitute the terms of Q_i by their correspondents [02].

5. Description of the various types of agents

In this section we describe the roles of the agents of the GAACSM architecture.

5.1. Intelligent agents

They are mono-domain agents. They acquire information coming from other agents. They are gradually adapted and enriched from their internal KBs and the evolution coming from the environment. The IA roles are multiple:

(1) The execution of the users or applications queries.
(2) The comparison of ontologies and the automatic generation of the mapping rules of the schemas between an IAC and another IAS.

(3) It enriches the query semantically by using the ontology and the schema-ontology mapping.
(4) It translates the query Q coming from other IACs into query Q' expressed in the proper language of the local base of the supplier.
(5) Filtering of the results.

5.2. Routing agents

They are multi-domain agents, gathering the nearest semantically domain. The roles of a RA are

1. To gather the nearest semantically intelligent agents in a net contacts to be used as suppliers (Figs. 10 and 11). Algorithm 4 calculates the semantic proximity rate of an IA, compared to the domain of an AR. This last gathers several IAs near semantically. Algorithm 5 selects the RA which has the best proximity rate for a IA. This last decides to choose the selected RA if its proximity rate is higher than a predefined threshold.
2. To ensure the dynamic query resolution, and to communicate with other routing agents in order to execute the queries ontologies.

Algorithm 4. Semantic proximity rate

Require: O_{newIA} a ontology describes the domain name of the new IA. Θ is a set of ontologies of the other IASs in the list of a RA.

```
Sim←0 ;
for each O_i ∈ Θ do /* i varies from 1 to N */
  Sim←Sim+SimProx( O_newIA , O_i )
EndFor
Sim← Sim/N
semantic proximity rate is Sim
end.
```

Figure 10 Semantic proximity rate.

Agorithme 5. Nearest semantically routing agent

Require: the intelligent agent receives answers of the all routings agents; *L* is the list of these agents contains its semantic proximity rate. The *ml* indicates the minimum level where an RA is near semantically to an IA.

```
NSRA←0;
while L<>∅ do
extract an element e from the list L
if  e.rate > NSRA  and e.rate > ml then
     NSRA ← e.rate
   Save the routing agent who has this rate
  Endif
endWhile
return the routing agent saving
End
```

Figure 11 Algorithm 5. Nearest semantically routing agent.

3. To record/eliminate dynamically the agents which take part in the cooperation.
4. To search IAs containing information to which the domain is the nearest to the one of an intelligent agent domain (consuming).

6. Queries processing

The query processing is divided into several steps, and during this process, the multi-agents system uses a set of protocols. The principal steps are (Fig. 12):

6.1. Static query resolution

The static resolution is applied to the systems have been already discovered.

Step 1: Query validation, the IAC checks the validity of the query, i.e., whether it is written in schema mediation terms or not
Step 2: Query reformulation: the query is divided into a recombining query of the results and sub queries intended for the IAS which contain data necessary to the execution of the query. The decomposition of the query is done by the use of the mapping rules. The IAC applies the cooperation protocol of static query resolution.
Step 3: Recombining of the results: the IAC executes the recombining query for the results.

Two measures of complexity are used for query processing problems: query complexity and data complexity. The query complexity measures the query answering time in terms of the size of the query Q, holding the other inputs fixed. High query complexity (NP-complete or worse), which is quite common for practical database languages, is not considered a serious impediment to implementation, because queries are generally considered to be very small compared with the size of the data. Data complexity measures the running time in terms of the size of the data. Since the data can be quite large, data complexity is by far the more important measure. The result shows that the data complexity of answering queries in GLAV is no harder than in LAV [48]. Our static query resolution uses the GLAV mappings (Fig. 9), so the answering conjunctive queries in GLAV is in LogSpace in data complexity [45].

Algorithm 6. Query processing

```
Given L the list of discovered agents and their mappings
If QueryValidation() then
1:if  L <> empty then
        - QueryReformulation()
        - StaticRecombiningResults()
 2: Dynamic query resolution
        - SemanticEnrichmentQuery()
        - TransmissionSemanticallyEnrichedQuery()
        - SemanticEvaluation()
        - DynamicRecombiningResults()
```

Figure 12 Algorithm 6. Query processing.

6.2. Dynamic query resolution

The dynamic resolution makes it possible to take into account the appearance of new IASs. The principal steps are

Step 1: Semantic enrichment of a query. The IAC enriches the query semantically by using the ontology and the links schema-ontology which are in its own knowledge base (Definition 11).
Step 2: Transmission of the semantically enriched query. The IAC applies the cooperation protocol of dynamic query resolution. So it transmits the semantically enriched query to the routing agent which is nearest semantically. This latter sends it to all IASs of its net contacts.
Step 3: Semantic evaluation of the semantically enriched query (Algorithm 2). Each IAS answers according to its capacity to treat the query:

(1) To compare elements of the query with its ontology. The elements of the query and its ontology are compared by using a semantic distance. The identified elements as equivalent are retained.

(2) The query is rewritten in terms of the equivalent elements of its ontology (then interpreted on its schema) to take into account the semantic conflicts of values (each intelligent agent has library of functions for the conversion of the types) (Definition 14).

(3) The answer is sent latter to the routing agent, indicating the manner of treating the query, so that this letter can build recombining queries of the results.

If no IAS answers, the routing agent sends the query to the other routings agents of other domains and if there are answers the routing agent updates its net contacts.

Stage 4: Results recombining: the routing agent recomposes the results obtained by IASs. Then it sends the final result to the IAC, this latter recomposes the results of static and dynamic query resolution.

Considering as elementary the operation of calculating the similarity between two elements e1 and e2. Our dynamic query processing has polynomial data complexity.

7. Technical aspects and prototype implementation

Our implementation is based on three class libraries: OntoSim [39], Alignment API [40] based on OWL API [41], and Jade [42]. OntoSim provides many similarities measurements between character strings. Alignment API allows to integrate new methods of similarities measurement (between two OWL ontologies) by implementing a Java interface. Jade (Java Agent Development Framework) [42,43] is used for the construction of the multi agents systems and the realization of applications in conformity with FIPA specifications [26]. The cooperation protocols are implemented using the Jade platform. Concerning the local information systems, the local database of the consuming system and the database of the supplier system 1 are established under the Access DBMS and the Windows XP operating system. The database of the supplier system 2 is implemented in XML files and the same operating system. The scalability and the performances of the transport system

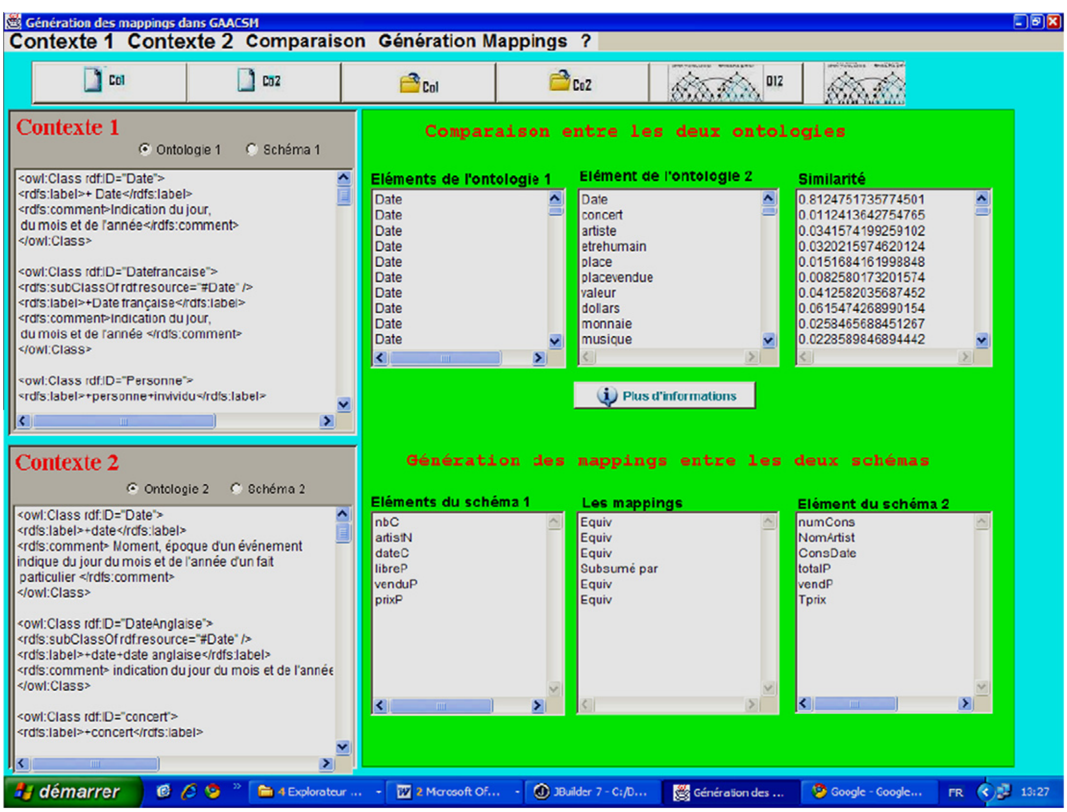


Figure 13 Automatic mapping generation.

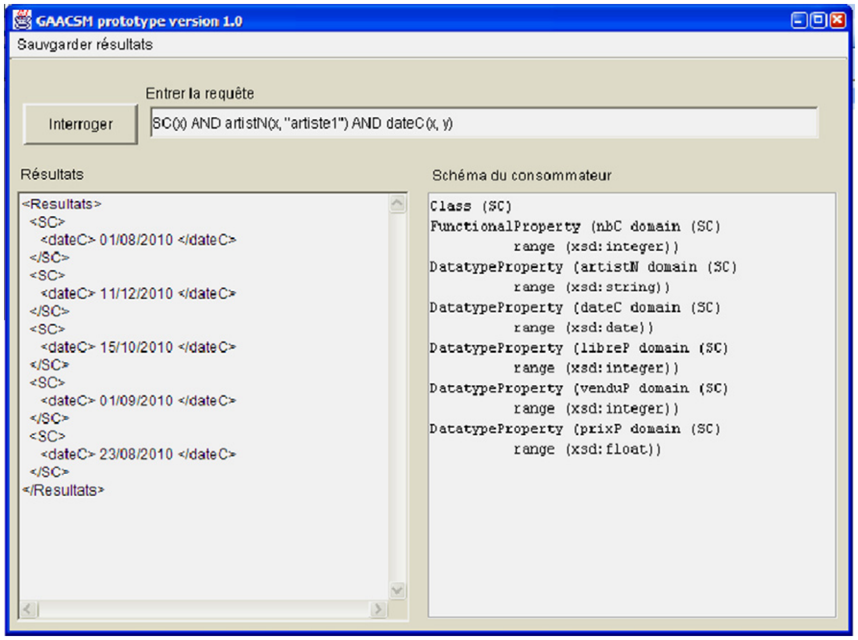


Figure 14 A simple example.

of Jade message were treated in [27,28]. The obtained results confirm the fact that Jade deals well with the scalability according to several scenarios intra or inter framework. The Fig. 13 presents an example of comparison between two ontologies of the consuming system and the supplier system 2. Fig. 14 presents the graphical interface, an example of query and the obtained results. In this example, the IAS1 is discovered by agent IAC. This last applies the schema mediation in order to reformulate the query. The IAC applies the context mediation for other agents, which are not yet discovered (IAS2). It communicates with the agent RA.

8. Conclusion and future research

In this paper, we presented an intelligent and an efficient query processing for semantic mediation of information systems. We proposed a generic multi agent architecture supporting our approach. The main advantage of our query answering is its robustness with regard to the evolution of systems, adaptation to the changes of environment and solves the most various data conflicts in a dynamic way. The developed prototype shows us the functionality of architecture suggested. As prospect we try to slacken our data mediation towards service mediation in general and to use intelligent

methods to reduce ontologies to be compared not to influence the scalability of architecture suggested.

References

[1] Sciore E, Siegel M, Resenthal A. Using semantic values to facilitate interoperability among heterogeneous information systems. ACM Trans Database Syst 1994;19(2):254–90.

[2] Akahani J, Hiramatsu K, Satoh T. Approximate query reformulation for ontology integration. In: Semantic integration workshop (SI-2003) second international semantic web conference, Sanibel Island, Florida, USA, October 20, 2003.

[3] Jouanot F. DILEMMA: vers une coopération de systèmes d'informations basée sur la médiation sémantique et la fusion d'objets. université de bourgougne, novembre 2001.

[4] Schneider M, Bejaoui L, Bertin G. A semantic matching approach for mediating heterogeneous sources. metadata and semantic. US: springer; 2009, ISBN 978-0-387-77744-3, pp. 537–548.

[5] Busse S, Kutsche R, Leser U, Weber H. Federated information systems: concepts, terminology and architectures. Technical Report Nr. 99-9, Berlin University, 1999.

[6] Hakimpour F, thesis: Using Ontologies to Resolve Semantic Heterogeneity for Integrating Spatial Database Schemata, Universität Zürich Zürich, 2003.

[7] Palopoli L, Terracina G, Ursino D. DIKE: a system supporting the semi-automatic construction of cooperative information systems from heterogeneous databases. Softw. Pract. Exper. 2003; 33(9):847–84.

[8] Leclercq E, Benslimane D, Yétongnon K. ISIS: a semantic mediation model and an agent based architecture for GIS interoperability. In: Proceedings of the international database engineering and applications symposium (IDEAS'99), Montreal, CANADA, Published by IEEE Computer Society Press, ISBN 0-7695-0265-2, 1999. p. 81–92.

[9] Arens Y, Hsu C. Query processing in the SIMS information mediator. In: Tate A, editor. Advanced planning technology, Menlo Park, CA, AAAI Press, 1996.

[10] Bressan S, Goh C, Levina N, Madnick S, Shah A, Siegel M. Context knowledge representation and reasoning in the context interchange system. Appl Intell 2000;13(2):165–80.

[11] Bayardo Jr RJ et al. InfoSleuth: agent-based semantic integration of information in open and dynamic environments. Microelectronics and Computer Technology Corporation, 1997.

[12] Mena E, Kashyap V, Sheth A, Illarramendi A. OBSERVER: an approach for query processing in global information systems based on interoperation across pre-existing ontologies. Int J Distrib Parallel Datab DAPD 2000;8(2):223–72.

[13] Do H, Rahm E. COMA – a system for flexible combination of schema matching approaches, VLDB 2002, 2002. p. 610–21.

[14] Madhavan J, Philip A, Bernstein, Rahm E. Generic schema matching with cupid, VLDB 2001, 2001. p. 49–58.

[15] Girardi R. An analysis of the contributions of the agent paradigm for the development of complex systems. In: (SCI 2001) and (ISAS 2001), Orlando, Florida, 2001.
[16] Purvis M, Cranefield S, Bush G, Carter D, McKinlay B, Nowostawski M, Ward R. The NZDIS project: an agent-based distributed information systems architecture. In: Proceedings of the Hawai'i international conference on system sciences, Maui, Hawaii, January 4–7, 2000.
[17] Suwanmanee S, Benslimane D, Thiran P. OWL based approach for semantic interoperability. In: Proceedings of the 19th international conference on advanced information networking and applications, AINA 05, 2005.
[18] Mohsenzadeh M, Shams F, Teshnehlab M. Comparison of schema matching systems, WEC (2), 2005. p 141–7.
[19] Deborah L, McGuinness, Frank van Harmelen. Owl web ontology language, Technical report, 2004.
[20] Wache H, Vögele T, Visser U, Stuckenschmidt H, Schuster G, Neumann H, Hübner S.Ontology-based integration of information – a survey of existing approaches. In: Stuckenschmidt H, editor. IJCAI-01 workshop: ontologies and information sharing, 2001. p. 108–17.
[21] Jouanot F, Cullot N, Yetongnon K. Context comparison for object fusion CAiSE 2003, LNCS 2681, 2003. p. 536–51.
[22] Rubén Tous, Thesis: data integration with XML and semantic web technologies. Universitat Pompeu Fabra, Barcelona, ISBN:9783836471381, June 2008.
[23] Goasdoué F, Rousset M. Querying distributed data through distributed ontologies: a simple but scalable approach. IEEE Intell Syst 2003:60–5.
[24] Tom N, Saw H. OWL-based approach for semantic teroperating and accessing heterogeneous XML sources, 0-7803-9521-2/06/ $20.00 §2006 IEEE, 2006.
[25] Talens G, Boulanger D, Séguran M. Domain ontologies evolutions to solve semantic conflicts. In: Collard M, editor. ODBIS 2005/2006, LNCS 4623, Springer-Verlag, Berlin, Heidelberg, 2007. p. 51–67.
[26] Online document: FIPA Communicative Act Library Specification, 2000.
[27] Cortese E, Quarta F, Vitaglione G. Scalability and performance of jade message transport system. In: Presented at AAMAS workshop on AgentCities, Bologna, 16th July, 2002.
[28] M.agali Séguran, Thesis : Résolution des conflits sémantiques dans les systèmes d'information coopératifs: proposition d'un modèle d'interaction entre agents, Université Jean Moulin, Lyon3, 2003.
[29] Kim W, Woo Choi D, Park S. Agent based intelligent search framework for product information using ontology mapping. J Intell Inf Syst 2008;30:227–47. DOI 10.1007/s10844-006-0026-8, Springer Science + Business Media, LLC, 2008.
[30] Mougin F, Burgun A, Bodenreider O, Chabalier J, Loréal4 O, Le Beux1 P. Automatic methods for integrating biomedical data sources in a mediator-based system, DILS 2008, LNBI 5109, Springer-Verlag, Berlin, Heidelberg, 2008. p. 61–76.
[31] Hansen HL, Ingvaldsen K. Dynamic schema integration the ACRoS prototype THESIS, The Faculty of Social Sciences, Department of Information Science and Media Studies, May 2005.
[32] Gal A, Segev A. Agent oriented data integration. In: Akoka J et al., editors. R Workshops 2005, LNCS 3770, Springer-Verlag, Berlin, Heidelberg, 2005. p. 98–108.
[33] Peter haase, dissertation, semantic technologies for distributed information systems, universitatsverlag karlsruhe, ISBN 978-3-86644-100-2, 2007.
[34] Ning Zhong, Jiming Liu, Setsuo Ohsuga, Jeffrey Bradshaw. Intelligent Agent Research and Development 2nd Asia-Pacific Conference on IAT , ISBN 981-02-4706-0 World Scientific Publishing Co. Pty. Ltd., 2001.
[35] Carbonell JG, Siekmann J. Lecture notes in artificial intelligence, Agent-Oriented Information Systems III 7th International Bi-ConferenceWorkshop, AOIS 2005 Utrecht, ISBN-10 3-540-48291-1. Springer, Berlin, Heidelberg, NewYork, 2006.
[36] Bach TL, Dieng-Kuntz R, Gandon F. On ontology matching problems (for building a corporate semantic web in a multi-communities organization). In: Proceedings of ICEIS 2004. Porto, Portugal, 14–17 April 2004.
[37] Bach TL, Dieng-Kuntz R. Measuring similarity of elements in OWL ontologies. AAAI'2005 workshop on contexts and ontologies: theory, practice and applications, Pittsburgh, Pennsylvania, USA, July 2005.
[38] Ai-Ping Li, Yan Jia, Quan-Yuan Wu. A study on organizational knowledge in multi-agent system. JCIT: J Converg Informat Technol 2007;2(1):61–5.
[39] http://www.gforge.inria.fr/projects/ontosim/.
[40] http://www.alignapi.gforge.inria.fr/lib.html.
[41] http://www.owlapi.sourceforge.net/.
[42] http://www.jade.tilab.com/.
[43] Bellifemine F, Caire G, Greenwood D. Developing multi-agent systems with JADE. John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester, West Sussex, PO19 8SQ, England, Copyright, 2007.
[44] Jouanot F. Un modele sémantique pour l'interoperabilité de systemes d'information, laboratoire LE2I, INFORSID, 2000.
[45] Calvanese D. Knowledge Bases and Databases, Part 3: Information Integration Faculty of Computer Science Master of Science in Computer Science AY, 2009.
[46] Nagy M, Vargas-Vera M. Towards an automatic semantic data integration: multi agent framework approach, The open Universiy united kingdom, ISBN 978-953-7619-54-1, 2010.
[47] Trajkovski G. Developments in intelligent agent technologies and multi-agent systems: concepts and applications, DOI: 10.4018/ 978-1-60960-171-3, ISBN13: 978-1-60960-171-3, 2011.
[48] Friedman M, Alon Levy, Todd Millstein. Navigational plans for data integration, AAAI-99 Proceedings, 1999.