SOAP-enabled web services for knowledge management

I. Benetti*, F. Guerra, M. Vincini and S. Bergamaschi

Dipartimento di Ingegneria dell'Informazione,
Università di Modena e Reggio Emilia, Via Vignolese 905,
41100 Modena, Italia
E-mail: benetti.ilario@unimo.it E-mail: guerra.francesco@unimo.it
E-mail: vincini.maurizio@unimo.it
E-mail: bergamaschi.sonia@unimo.it
*Corresponding author

Abstract: The widespread diffusion of the World Wide Web among medium/small companies yields a huge amount of information to make business available online. Nevertheless the heterogeneity of that information, forces even trading partners involved in the same business process to face daily interoperability issues.

The challenge is the integration of distributed business processes, which, in turn, means integration of heterogeneous data coming from distributed sources.

This paper presents the new web services-based architecture of the MOMIS (Mediator envirOnment for Multiple Information Sources) framework that enhances the semantic integration features of MOMIS, leveraging new technologies such as XML web services and the SOAP protocol.

The new architecture decouples the different MOMIS modules, publishing them as XML web services. Since the SOAP protocol used to access XML web services requires the same network security settings as a normal internet browser, companies are enabled to share knowledge without softening their protection strategies.

Keywords: web-services; e-commerce; information integration; ontology.

Reference to this paper should be made as follows: Benetti, I., Guerra, F., Vincini, M. and Bergamaschi, S. (2004) 'SOAP-enabled web services for knowledge management', *Int. J. Web Engineering and Technology*, Vol. 1, No. 2, pp.218–235.

Biographical notes: Ilario Benetti is a software consultant. His research interests include information integration from heterogeneous data sources, especially e-commerce applications. He received a PhD in computer science and engineering at the University of Modena e Reggio Emilia.

Sonia Bergamaschi is a Full Professor of Databases at the Dipartimento di Ingegneria dell'Informazione, Università di Modena e Reggio Emilia. Her research interests include intelligent integration of information, knowledge representation and management in the context of very large databases that face theoretical and implementation problems. She is a member of the IEEE Computer Society and the ACM and is the coordinator of the EU IST SEWASIE Project (semantic webs and agents in integrated economies).

Francesco Guerra is a PhD candidate in information engineering at the Dipartimento di Ingegneria dell'Informazione, Università di Modena e Reggio Emilia. His research interests include integration of heterogeneous information sources, ontologies and semantic web. He received a Laurea in computer engineering from the Università di Modena e Reggio Emilia.

Maurizio Vincini is a Research Associate in information engineering at the Dipartimento di Ingegneria dell'Informazione, Università di Modena e Reggio Emilia. His research interests include the intelligent integration of information systems based on reasoning techniques, ontologies and mediator information software agents, object oriented database design and query optimisation. He received a PhD in computer science and engineering from the Università di Modena e Reggio Emilia.

1 Introduction

The widespread diffusion of the World Wide Web has not only reached big companies, but also small and medium businesses, yielding a huge amount of information available online. Nevertheless the heterogeneity of that information, forces even trading partners to face daily, interoperability issues.

The heterogeneity of networked information may bring both semantic conflicts (when companies use different terms to refer to the same concept) and structural incompatibilities (when different data representation models are adopted).

It is important to underline that existing standards could be not sufficient to achieve a complete interoperability if the legacy systems or the involved enterprise resources planning software have been released long before the introduction of those standards. Under these conditions, as well as when a common standard cannot be established, the integration of heterogeneous systems appears to be the only chance.

The challenge is the integration of distributed business processes, which, in turn, means integration of heterogeneous data coming from distributed sources. For small and medium companies, with reduced investments in information technology, it is extremely important to overcome the lack of a common ontology between business partners without modifying either the internal data storage system or the network infrastructure.

Integration has become more and more relevant inside the business application developers' community, since a new class of applications, called EAI (enterprise application integration), has been recently introduced.

In general, three ways to address conflicts between heterogeneous applications can be individuated:

- completely rewriting existing applications: it is a costly solution and it could be compromised by the evolution of standards
- abandoning the integration projects, thus renouncing the possibilities offered by interoperable applications such as the business processes automation
- adopting integration software, which in a relatively short time and with small changes to the existing systems allows a connection to be established between existing and newer applications.

Each of the proposed solutions offers advantages and disadvantages. The integration currently appears to be the most promising. Integration is flexible, since it allows users to add or remove elements at any time and scale, as it can be performed in an incremental way.

The opportunity of connecting existing systems relies on the availability of software designed to automate the integration process as much as possible. Software of this kind, usually referred to as middleware, acts as an interpreter between different IT systems.

With this in mind, we developed the MOMIS system (Mediator envirOnment for Multiple Information Sources) [1-3]. MOMIS is a mediator-based system for information extraction and integration that works with structured and semi-structured data sources. The MOMIS system obtains the semantic integration by creating a thesaurus of terminological relationships holding both at intra-source and inter-source level. At this stage MOMIS exploits both inference techniques and the WordNet lexical system and it creates a global virtual view of all the sources in the integration domain evaluating affinities among concepts in the thesaurus. Software modules that communicate using the CORBA standard compose the system.

As companies connect to the internet, the awareness of the risks of a permanent connection increases significantly. In turn, information sharing frequently involves policies: firewalls, proxy servers and encryption algorithms that ensure a reasonable level of security for both data and network connectivity.

In this paper we propose a new web service-based architecture in order to enhance the semantic integration features of the MOMIS, leveraging new technologies such as XML web services and SOAP (simple object access protocol) [4,5].

The semantic integration carried out by MOMIS does not affect the structure of the sources to be integrated. The new architecture decouples the different modules of MOMIS, publishing them as XML web services. Enabling XML web services within the original MOMIS architecture [6] means, on the one hand, exploiting the benefits brought by real platform independence, low overhead service integration and, on the other hand, splitting the integration process over powerful distributed software modules. Since the SOAP protocol used to access XML web services requires the same network security settings as a normal internet browser, companies are enabled to share knowledge without softening their protection strategies, hence the business processes integration becomes a cost-effective task.

The paper is organised as follows. Section 2 introduces some of the concepts on which web services are based; Section 3 describes how web service-based architecture improves our integration system; Section 4 shows a running example, and finally Section 5 offers some concluding remarks.

2 Web services at a glance

It is a largely widespread opinion that web services will be the fundamental building blocks in the move to distributed computing on the internet. In fact, enterprises are moving their existing applications to the web and consequently a complete infrastructure to manage the specific issue introduced by the open platform is needed [7].

Several definitions of web services are been provided. In our opinion, a web service may be thought of as a self-contained, modular application that can be described,

published, located and invoked over a network, generally, the World Wide Web. Essentially web service architecture may describe three roles [4,8]:

- *Service provider*: from a business perspective, this is the owner of the service. From an architectural perspective, this is the platform that hosts access to the service.
- *Service requestor*: from a business perspective, this is the business that requires certain functions to be satisfied. From an architectural perspective, this is the application that is looking for and invoking or initiating an interaction with a service. The service requestor role can be played by a browser driven by a person or a program without a user interface, for example another web service.
- *Service registry*: this is a searchable registry where service providers publish their service descriptions. Service requestors find services and obtain binding information (in the service descriptions) for services during development for static binding or during execution for dynamic binding. For statically bound service requestors, the service registry is an optional role in the architecture, because a service provider can send the description directly to service requestors. Likewise, service requestors can obtain a service description from other sources besides a service registry, such as a local file, FTP site, website, advertisement and discovery of services (ADS) or discovery of web services (DISCO).

Existing applications can be integrated more rapidly, easily and less expensively, since web services reduce what is absolutely required for interoperability to the minimum. Integration occurs at a higher level in the protocol stack, based on messages centred more on service semantics and less on network protocol semantics, thus enabling real platform and language independence. These characteristics are ideal for connecting business functions across the web both between enterprises and within enterprises. They provide a unifying programming model so that application integration inside and outside the enterprise can be done with a common approach, leveraging a common infrastructure. The integration and application of web services can be done in an incremental manner, by using existing languages and platforms and by adopting existing legacy applications.

Previous platforms and architectures relying on distributed computing (CORBA, DCOM, Java RMI) have yielded systems where the coupling between various components is too tight to be effective for low overhead, every time and everywhere B2B applications over the internet. These approaches require too much agreement and shared context among business systems from different organisations to be reliable for open, e-business cross-platform.

2.1 The SOAP approach

SOAP is a lightweight protocol for exchange of information in a decentralised, distributed environment. It is an XML-based protocol that consists of three parts: an envelope that defines a framework for describing what is in a message and how to process it, a set of serialising rules for expressing instances of application-defined data types and a convention for representing remote procedure calls and responses.

All SOAP messages are encoded using XML. A major design goal for SOAP is simplicity and extensibility. This means that there are several features from traditional messaging systems and distributed object systems that are not part of the core SOAP

specification. SOAP defines a message-processing model but does not itself define any application semantics, such as a programming model or implementation-specific semantics.

The SOAP specification also defines the relationships between HTTP messages and SOAP. This HTTP binding is important because HTTP is supported by almost all modern operating systems. The HTTP binding is optional, but almost all SOAP implementations support it. The HTTP transport binding for SOAP makes it attractive for industrial uses. Since most organisations are familiar with HTTP and already have it incorporated into their network infrastructure, SOAP fits right in without the complex changes to the network or firewalls that many other protocols require.

One of the most relevant uses of SOAP is to enable XML web services. An XML web service is a function that is exposed through a SOAP interface so that other SOAP-based applications on the web can call it to access the service.

WSDL (web services description language) [9] is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information. As communications protocols and message formats are standardised in the web community, it becomes increasingly possible and important to be able to describe the communications in some structured way. WSDL addresses this need by defining an XML grammar for describing network services as collections of communication endpoints capable of exchanging messages. WSDL service definitions provide documentation for distributed systems and serve as a recipe for automating the details involved in applications communication. A WSDL file is an XML document that describes a set of SOAP messages and how the messages are exchanged. Since WSDL is XML, it is readable and editable, but in most cases, it is generated and consumed by software. SOAP introduces the following advantages with regard to the communication mechanism used by the CORBA architecture:

- While IIOP, ORPC, are binary protocols, SOAP is a text-based protocol. Using XML for data encoding gives SOAP some unique capabilities. For example, due to the readability of an XML file, it is much easier to debug applications based on SOAP than on a binary stream. Vice-versa, the SOAP protocol it is not optimised to transfer huge data sources.
- Due to the communications among the different SOAP machines uses the HTTP protocol, no further configurations are needed in order to overcome firewalls and others protections.
- Because it is based on a vendor-agnostic technology, namely XML, HTTP and simple mail transfer potocol (SMTP), SOAP appeals to all vendors.

3 The MOMIS system

3.1 Semantic integration of heterogeneous data sources

The MOMIS system is a framework for integration and querying of distributed and heterogeneous data sources. MOMIS exploits the semantics expressed by the conceptual schemata, or metadata, of the data sources to be integrated, to create a global virtual view of all the sources. The MOMIS system allows the external applications to perform

queries directly to the integrated global schema. The system, originally based on an I^3 architecture [9], consists of the three functional elements: a common data model, one data wrapper for each data source involved by the integration, and a mediator.

For a semantically rich representation of source schemas and object patterns, MOMIS uses an object-oriented language called ODL_{I3}. ODL_{I3}. is close to the ODL language and can be used to describe heterogeneous schemas of structured and semi-structured data sources. ODL_{I3} extends ODMG-ODL with intentional and extensional relationships expressing intra-schema and inter-schema knowledge for the source schemas. In particular ODL_{I3} extends ODL with the following relationships:

- syn (synonym of) is a relationship defined between two terms ti and tj (where ti tj) that are synonyms in every involved source
- bt (broader terms) is a relationship defined between two terms ti and tj, where ti has a broader, more general meaning than tj. bt relationships are not symmetric. The opposite of bt is nt (narrower terms)
- rt (related terms) is a relationship defined between two terms ti and tj that are generally used together in the same context in the considered sources.

The data wrappers run over the data sources to be integrated to translate the conceptual schema of each source into a common ODL_{I3} format. Therefore, the wrappers are responsible for translating the queries over the global virtual view into queries expressed in a language compliant with those of the sources and for exporting the results to the mediator.

Finally, the mediator consists of two modules: the global schema builder and the query manager. The global schema builder processes and integrates ODL_{I3} descriptions received from the wrappers to create the global virtual view. To accomplish this, the mediator combines the reasoning capabilities of the description logics with affinity-based clustering techniques [6]. The query manager module performs query processing and optimisation. The user's applications interact with MOMIS by querying the global view using the ODL_{I3} language, which is a subset of OQL-ODMG. The query manager assists this phase by generating the OQL_{I3} queries for the wrappers. Using mapping-description techniques, the query manager generates the queries automatically by formulating and optimising the generic OQL_{I3} queries into different sub queries, one for each involved data source and synthesises a unified global result.

The original contribution of MOMIS is related to the availability of a set of techniques for the designer to face common problems that arise when integrating pre-existing information sources, containing both semi-structured and structured data. MOMIS provides the capability of explicitly introducing many kinds of knowledge for integration, such as integrity constraints, intra-source and inter-source intensional and extensional relationships and designer supplied domain knowledge. A common thesaurus, which has the role of a shared ontology of the source, is built in a semi-automatic way. The common thesaurus is a set of intra-schema and inter-schema intensional and extensional relationships, describing inter-schema knowledge about classes and attributes of sources' schemas; it provides a reference on which to base the identification of classes, candidate to integration and subsequent derivation of their global representation.

MOMIS supports information integration in the creation of an integrated view of all sources (global virtual view) in a way that is automated as much as possible and performs revision and validation of the various kinds of knowledge used for the integration. To this end, MOMIS combines reasoning capabilities of description logics with affinity-based clustering techniques, by exploiting a common ontology for the sources constructed using lexical knowledge from WordNet and validated integration knowledge.

The global virtual view is expressed by using XML standard, to guarantee the interoperability with another open integration system prototype. The aforementioned functionalities of the MOMIS system are also available with web-interface [6].

3.2 The integration process

The MOMIS framework consists of semiautomatic and distributed software tools, which require an integration domain expert (the integration designer), to revise the results automatically computed by the mediator and even to add new knowledge, refining the global virtual view.

An incremental process provides the global schema. The first step is the creation of a thesaurus of lexical relationships. MOMIS extracts the relationships within a single local schema (intra-schema) by using inference techniques. Then, an ontology shared by the different local schemas (inter-schema relationships) is built by using the WordNet system [10], which identifies lexical relationships between inter-schema concepts on the basis of their meaning. At this stage, the correct meaning for each significant term within the local schemas has to be indicated, choosing from those suggested by the WordNet lexical system. This task is called annotation. The annotation can be performed at each single wrapper (local annotation) or at mediator level (centralised annotation). The annotation process in conjunction with WordNet properties (synonymy, polysemy, hypernymy, olonomy, correlation) allows the MOMIS system to define new lexicon/terminological relationships among the ODL_{I3} classes and attributes.

Both the inter-schema and the intra-schema relationships are stored in the shared ontology, which, in turn, is defined common thesaurus and validated by the mediator. The integration designer is allowed to insert new intensional or extensional relationships into the common thesaurus to capture specific knowledge about the integration domain. The designer can delete insignificant inferred relationships as well. Finally, inference capabilities of ODB-tools [11] are exploited to obtain a new set of structural/terminological relationships by using subsumption, (i.e. generalisation) and equivalence computation.

MOMIS uses relationships in the common thesaurus to evaluate the level of affinity between objects both at intra-schema and inter-schema level. The concept of affinity is introduced to formalise the kinds of relationship that can occur between objects. MOMIS groups together in the same cluster classes having affinity in different sources, by using hierarchical clustering techniques. The goal is to identify the classes that will form the global schema: for each cluster in the tree, a global class representative of the classes contained in the cluster is defined via interactive process with the designer. First MOMIS associates to the global class a set of global attributes corresponding to the union of the attributes of the classes belonging to the cluster, where the attributes are automatically unified into a unique global attribute by exploiting terminological relationships. In short, we can say that the global attributes are obtained in two steps:

- union of the attributes of all the classes belonging to the cluster
- fusion of the 'similar' attributes; in this step redundancies are eliminated in a semi-automatic way taking into account the relationships stored in the common thesaurus.

To complete global class definition, information on local/global attribute mappings and default values is provided by the designer in the form of declarative mapping rules. For each global class a persistent mapping table, storing all the intensional mappings is generated; it is a table whose columns represent the set of the local classes, which belong to the cluster and whose rows represent the global attributes.

An element *MT*[*L*][*ag*] represents how the global attribute *ag* is mapped into the local class *L*. Each element MT[L][ag] of the table is a mapping function of the values assumed by the set of attributes MT[L][ag]. Some simple and frequent cases of this mapping function are:

- *MT*[*L*][*ag*] = *al*: the global attribute *ag* maps into the *al* local attribute
- *MT*[*L*][*ag*] = al_1 and al_2 and al_n: this is used when the value of the *ag* attribute is the concatenation of the values assumed by a set of attributes al_i belonging to the same local class *L*
- *MT*[*L*][*ag*] = `case of` *al* $\texttt{const}_1$: al_1, …$\texttt{const}_n$: al_n: this situation occurs when the *ag* global attribute can assume one value in a set of al_i belonging to the same local class *L* and the value choice depends on a third attribute, *al*, from the same class, which act as a selector
- *MT*[*L*][*ag*] = `const`: in this case a global attribute value does not refer to any local attribute and a constant value is set by the designer (see the Rank attribute)
- *MT*[*L*][*ag*] = `null`: in this case no attribute of the class *L* corresponds to the global attribute *ag*

The global schema consists of all the classes derived from clusters and it is the basis for submitting queries against the sources.

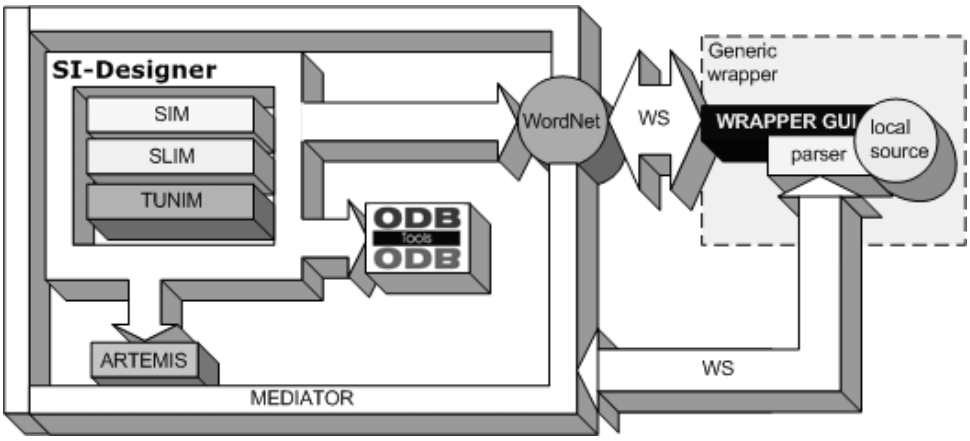
3.3 Software at work

MOMIS is developed as a distributed system, where each node represents a local data source to be integrated and the nodes are connected to the mediator that is a central point to information access for the final user.

Each node is encapsulated into a wrapper conceived as a web services provider to make available machine-processable information for the mediator, while the mediator also provides a web service for the wrapper.

The web services are implemented by following SOAP protocol, so that the MOMIS architecture consists of information interchange between the mediator and the wrapper by SOAP Client/Server platform (see Figure 1).

Figure 1 MOMIS architecture



Each wrapper makes the ODL_{I3} data description of its underlying source available, by exposing as SOAP Server the appropriate method (getDescription), which returns the ODL_{I3} schema as XML string. A running wrapper performs this translation in unattended mode, always making the information available. At present relational, object oriented, XML data source formats are supported. In addition, the wrapper permits the WordNet annotation directly at local source, by the interaction with the Wordnet dictionary accessed as a web service exposed by the mediator

For a local annotation, the wrapper must provide a SOAP client to invoke those methods and, in turn, to find out the meanings within the WordNet's lexicon. Unlike the translation into ODL_{I3}, the annotation is a user driven task. When the local annotation has been accomplished, the user issues the annotated schema. The information becomes available through the getAnnotatedDescription method, which joins the terms of the ODL_{I3} description with the address of their meanings within the WordNet database.

The mediator will then evaluate the lexical relationships that hold between the returned WordNet terms and insert them into the common thesaurus.

3.4 The role of XML web services

Enabling XML web services within the original MOMIS architecture means, on the one hand, extending the MOMIS capabilities through the benefits brought by real platform independence, low overhead service integration and, on the other hand, it causes a significant improvement of the capabilities of the integration process.

> "Moreover, this extended architecture decouples costs of the common thesaurus generation task"

The WDS acts on a single data source. Once the wrapper has generated the ODL_{I3} description of the local schema, the WDS uses the SOAP client located on the wrapper associated with the data source to access the WordNet web service running at mediator level. The WordNet web service allows the WDS to perform a precise annotation of the local schema by assigning the correct WordNet meaning to each term within the local schema. The WDS differs from the very integration designer since he is supposed to

supply a detailed knowledge about a specific data source rather than a global experience about the whole integration domain.

The common thesaurus generation starts after each local schema description has been translated into ODL_{I3} and, if possible, annotated by the WDS.

One of the most relevant advantages coming from the introduction of the WDS is releasing the integration designer from annotating each local schema. The uniqueness of the WordNet database, in addition, prevents ambiguity even in the presence of many data sources, (i.e. many different WDSs). The generation of the common thesaurus becomes a more rapid task since the integration designer deal with annotated sources.

Another major advantage becomes valuable in the presence of non-meaningful terms within the local schema. In theses cases the direct experience of a WDS is fundamental to the correct conversion of abbreviations, acronyms and conventional words into meaningful terms. Furthermore, if a local schema is expressed in a language other than English, a local specialist may be more precise in the translation of the terms from their original language into English than the integrator designer. Notice that the translation would be required since the WordNet morphological processor assumes that all words are expressed in English.

The wrappers, within a web services-based architecture, are service providers exposing both the ODL_{I3} description and the annotation of the local schema. Nevertheless the wrapper must include a SOAP compliant module acting as a web services requestor in order to access the WordNet web service and to provide the annotation of the local schema. A graphical user interface is also required to allow the WDS to easily browse the meanings available in WordNet and to assign them to the terms of the local schema.

A generic wrapper becomes a more complex software with respect to the wrappers described in the earlier MOMIS versions. The introduction of the web services within the original I^3 architecture involves a trade-off between the complexity and the versatility of the wrappers with reference to optional implementation of a SOAP client module for the local annotation of the schema. The more complex the wrapper application, the wider the contribution to the whole integration process.

A lightweight wrapper designed to convert the local schema into the ODL_{I3} format and to publish it as a web service, would be perfectly compliant with the rest of the system. It would be a straightforward solution, (considering the variety of web services publishing tools available) well tailored for meaningful data source.

4 Running example

Let us introduce the following example to illustrate both the integration process and the exploitation of the new architecture. Let us suppose that an industrial group had to run financial statistics about two controlled companies called CompA and CompB. CompA evaluates its financial performances by directly accessing the information about the invoices of a given year. This information is stored in a relational database. CompB creates an XML file, year by year, with data that could be useful in evaluating statistics.

This over-simplified example aims to illustrate the benefits coming from the exploitation of the distributed knowledge provided by the WDS rather than a complex integration scenario. The proposed integration domain requires a wrapper for relational databases and another one for XML sources. As the CompB file includes only

meaningful terms, a wrapper's domain specialist is not required for the local schema annotation. Therefore, a simple ODL_{I3} parser for XML source files would be the recommended wrapper for the CompB source. The CompA database's schema holds plenty of acronyms and conventional words. Thus, only a local expert would be able to explain the meaning of each term, or, in other words, to annotate the local schema. A local annotation is typically very effective under those conditions. Therefore, let us focus on the CompA wrapper.

The considered wrapper basically performs two steps. First it establishes a connection to CompA's schema and builds the corresponding ODL_{I3}*3* description on the basis of a fixed set of translation rules. The translated schema is stored in an environment variable of the wrapper, as an XML string. This value remains unchanged unless the wrapper is stopped and restarted.

When the mediator retrieves the ODL_{I3} schema through the getDescription web method, the web services inside the wrapper has only to read the environment variable and to assign it as return value of the method.

Table 1

CompA's database	*CompB's XML schema*
DOCH (ID, DT, DD, CID)	<?xml version="1.0" encoding="utf-8"?>
DOCR (DID,RD,IID,Q)	<xs:schema targetNamespace=http://tempuri.org/fnSchema.xsd
ITM (ID, DSC, PR, UM)	xmlns:xs="http://www.w3.org/2001/XMLSchema">
CST(ID, NM, USA)	<xs:complexType name="InvoiceStat">
	<xs:sequence>
	<xs:element name="ItemID" type="xs: integer "/>
	<xs:element name="ItemDesc" type="xs: string "/>
	<xs:element name="CustomerID" type="xs: integer "/>
	<xs:element name="CustName" type="xs: string "/>
	<xs:element name="Date" type="xs: date "/>
	<xs:element name="Price" type="xs:float"/>
	<xs:element name="Quantity" type="xs:float"/>
	<xs:element name="UnitMeas" type="xs:float"/>
	</xs:sequence>
	</xs:complexType>
	</xs:schema>

The second main purpose of the wrapper is to support the local annotation. To enable a local expert (WDS) to locally annotate the schema, the wrapper links up with the WordNet dictionary which is only stored at mediator level.

The annotation includes two steps: the base form choice and the meaning choice. The former requires the WDS to select the word form, (i.e. the way in which the word is written) from the list of suitable base forms supplied by the WordNet morphologic processor. This is accomplished by invoking the checkWord web method on the mediator, which returns an acknowledgment only if the word, has been found within WordNet. If a base form is not found (as we could expect in the CompA case) the WDS

can directly introduce it. In our example, the 'customer' base form would replace the CST term. Likewise, the WDS introduces meaningful base forms for each term of the local schema.

Figure 2 illustrates the ODL_{I3} format for the CST class, once the correct forms have been selected by the WDS. The latter is the meaning choice. The designer can relate a name to one, more than one, or no meaning. The choice of not relating a name to any meaning can be made for different reasons: the concept is too complex and it cannot be expressed with one word; it belongs to the tops, i.e. to generic concepts.

The meanings are retrieved by invoking the getSense web method, exposed by the mediator. When called for a base form, the function puts into a bi-dimensional array of values all the meanings associated to the form and the corresponding logical address within the WordNet's data files. The WDS is then allowed to choose on or more meanings. The wrapper stores, for each base form in the local schema, only the logical address of the meaning indicated by the WDS. In the case of CompA, the WDS associates the first meaning to the name attribute of the customer class: it turns the wrapper, keep the value corresponding logical address in the running version of WordNet (in this case 12548).

Figure 2 The CompB's XML schema

```
Interface    customer {
Attribute    integer     code;
Attribute    string      name;
Attribute    string      address;
}
```

At the end of the annotation, the WDS is required to save his choices and to issue the schema. From this point on, the mediator is allowed to invoke the getAnnotatedDescription on the wrapper to exploit the local annotation. Notice that any semantic ambiguity between concepts among different schemas will be removed by the annotation since a third party-supplied morphology, (i.e. the WordNet dictionary) is trustfully shared by each source.

The starting point of the exploited lexical semantics derives from the existence of a conventional association between the words form and the concept/meaning they express; such association is of the many-to-many kind and gives rise to the following properties:

- synonymy: property of a concept/meaning which can be expressed with two or more words
- polysemy: property of a single word having two or more meanings.

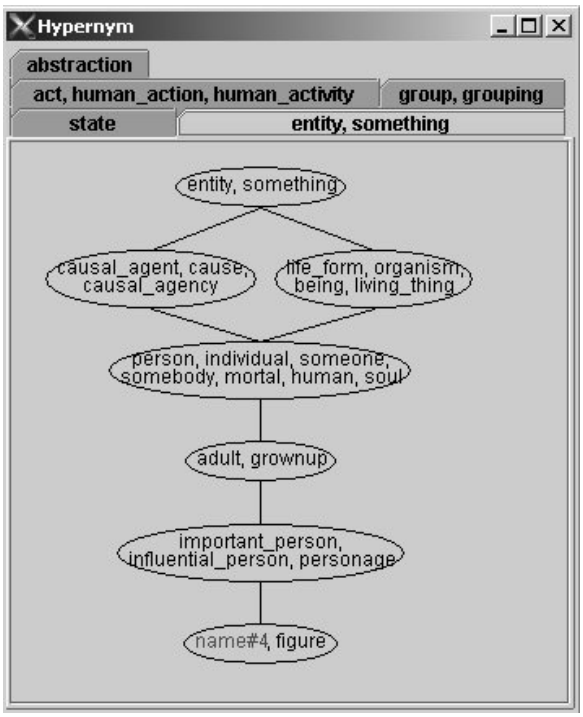
As the WDS selects one or more meanings from those found in WordNet, starting from the chosen base form, all the words that are related to the same name, share the same base form.

For example, all the 15 meanings that WordNet relates to the [name] base form are obtained. Selecting them all, i.e. considering 15 words for the [cutomer.name] attribute, we could obtain 'wrong' results, which are not suitable within the examined context.

That is the WDS experience becomes fundamental to making the correct choice. The annotation activity could be significantly long in terms of time, even for the WDS. The semi-automatic approach supplied by the wrapper reduces the complexity of the task, in fact, a 'difficult' problem, (i.e. finding the relations between all words), is divided into many 'easy' ones, choosing the meaning of each terms from a list.

Furthermore the wrapper provides a graphical representation of the generalisation hierarchy of the meanings in order to help the WDS in the most difficult choices (see Figue 3 for the [name] base form).

Figure 3 The [name] hypernym tree



Once the mediator has gathered the local schemas, the common thesaurus generation starts. Lexical relationships are extracted in the following order:

- *Schema-derived relationships*: extracted by analysing each ODL_{I3} schema separately. In particular, intra-schema relationships are extracted when an attribute of a class refers to another belonging to a different class in the same source.

  CompA.DOCR RT CompA.DOCH

  CompA.CST RT CompA.DOCH

  CompA.ITM RT CompA.DOCR

- *Lexicon derived relationship*: extracted exploiting the lexical relationships existing between terms in WordNet. If annotated schemas are gathered from the wrappers (see CompA case) the mediator has only to run the extraction algorithm. Otherwise (see CompB source in our example) the integration designer must first annotate the local schema and then extract the relationships.

  CompB.Invoice_Stat NT CompA.DOCH → (Invoice NT Document)

  CompA.DOCR NT CompA.DOCH → (line NT head)

  CompB.Invoice_Stat.Date SYN CompA.DOCH.Date

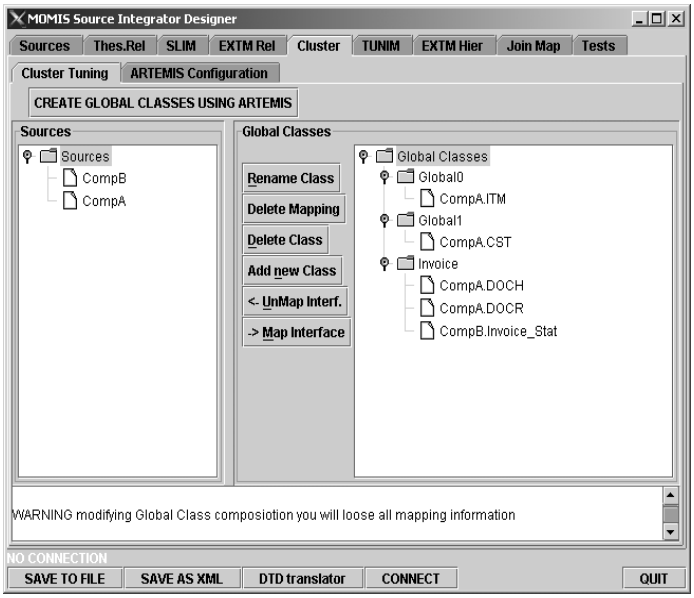
- *Inferred relationships:* holding at intra-schema level, are inferred by exploiting inference capabilities of a description logics-based component called ODB-Tools.

  CompA.CST RT CompB.Invoice_Stat

  CompA.ITM RT CompB.Invoice_Stat

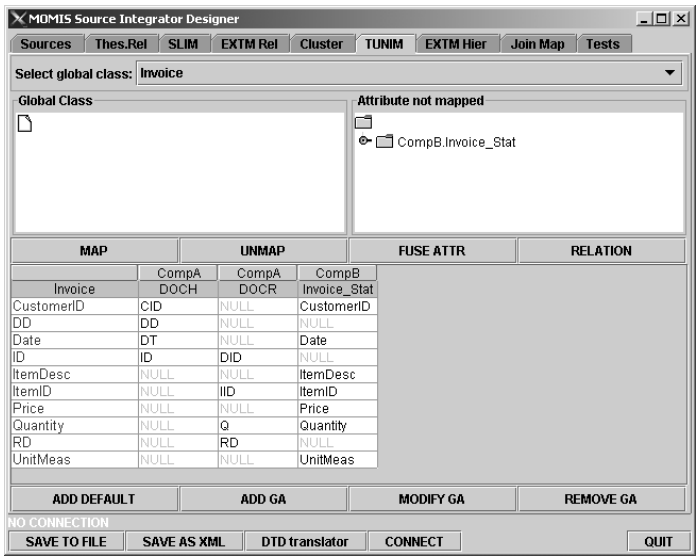
All these relationships are added to the Common Thesaurus and thus considered in the subsequent phase of construction of Global Schema. Figure 4 shows the global classes obtained for the considered integration domain.

Figure 4 The global classes



The mapping table in Figure 5 illustrates the correspondence between the global attributes and the attributes of the local schemas.

Figure 5 The mapping table



5 MOMIS and knowledge management

The integration process supported by MOMIS can be viewed from a KM perspective. To do so, we have re-conceptualised the mediation process using the well-known Nonaka knowledge creation model [12].

Local schemata are the starting point. They offer information that, coming from different sources, is not directly comparable. Resolving the syntactic and semantics inconsistencies existing between the various sources is the main goal of the MOMIS integration process that can be subdivided into two sub-processes.

The first, automatically performed, is conducted by software components such as wrappers, WordNet and so on. Using the Nonaka terminology this process can be regarded as a knowledge combination process, since it applies the explicit knowledge embedded in the software tools to the explicit knowledge contained in the catalogues to produce new explicit knowledge [13]. Such new information is organised in a different way (as far as the logical structure, the semantics and the syntax are concerned) from the initial one. Nevertheless, it still includes mistakes and inconsistencies that make it useless.

The second entails the decisive contribution of the integration designer and the WDSes. It can be considered as a knowledge externalisation process, since the designers apply their tacit knowledge about the business domain to the information generated in the previous phase, to create new information, that will form the global virtual view, i.e. the outcome of the integration process.

The last step concerns the search from the virtual catalogue and the utilisation of the retrieved information by the end-user. Also this process can be read according to the Nonaka taxonomy. In particular it can be considered a knowledge internalisation process, since the end-user applies his tacit knowledge on the information contained in the virtual catalogue to take business decisions.

The whole mediation process creates new knowledge and, consequently, produces business value. The amount of the generated knowledge (and value) can be in the first approximation estimated as the difference between the efficiency and efficacy [14] of the end-user decision, made on the basis of the various catalogues taken separately and starting from the unique virtual catalogue.

6 Related work

In the area of heterogeneous information integration, many projects based on mediator architectures have been developed. The mediator-based TSIMMIS project [15] follows a `structural' approach and uses a self-describing model (OEM) to represent heterogeneous data sources and the MSL (mediator specification language) rule to enforce source integration and pattern matching techniques to perform a predefined set of queries based on a query template. Differently from our integration approach proposal, in TSIMMIS only the predefined queries may be executed and for each source modification a manually mediated rules rewriting must be performed.

The SIMS [2] proposes to create a global schema definition by exploiting the use of description logics, (i.e. the LOOM language) for describing information sources. The use of a global schema allows both GARLIC and SIMS projects to support every possible user queries on the schema instead of a predefined subset of them.

The information manifold system [15] provides a source independent and query independent mediator. The input schema of information manifold is a set of descriptions of the sources and the integrated schema is mainly defined manually by the designer, while in our approach it is tool-supported.

Infomaster [16] provides integrated access to multiple distributed heterogeneous information sources, giving the illusion of a centralised, homogeneous information system. The main difference of this project, with regard to our approach, is the lack of a tool aid support for the designer in the integration process.

Also inside the multi-agent system community some work has been done in the direction of integration systems. For its similarities with the goal of the MOMIS system the Infosleuth system [17] deserves a particular mention. Infosleuth is a system designed to actively gather information by performing diverse information management activities. InfoSleuth agents enable a loose integration of technologies allowing:

- extraction of semantic concepts from autonomous information sources
- registration and integration of semantically annotated information from diverse sources
- temporal monitoring, information routing and identification of trends appearing across sources in the information network.

Infosleuth bases its data analysis on given ontologies, explicitly given by humans (rather than building them) and provides visibility of data related only to the specified queries, while our approach aims at building ontologies related with the integration domain.

Another important experience is the RETSINA multi-agent infrastructure for in-context information retrieval [18]. In particular the LARKS description language is defined to realise the agent matchmaking process (both at syntactic and semantic level)

by using several different filters: context, profile, similarity, signature and constraint matching.

7 Conclusions

In this paper we presented the new web services-based architecture of the MOMIS framework. The architecture enhances the semantic integration features of the MOMIS leveraging new technologies such as XML web services and the SOAP protocol.

The XML-based format makes SOAP human-readable, i.e. useful for debugging purposes and quick implementation [19]. In addition it is based on HTTP and can be implemented with little effort on top of exiting libraries and is supported by computer industry leaders (Microsoft, IBM, SUN), while CORBA (previously adopted by MOMIS) requires huge software packages and does not provide a commonly accepted bootstrapping mechanism. Furthermore decoupling the MOMIS modules brings a significant improvement in semantic integration, as both the knowledge supplied by the WDS and an overall cut of the infrastructure requirements make the business processes integration a low-overhead, cost-effective task.

Acknowledgments

This paper has been partially supported by MIUR within the D2I and the 'Agenti software e commercio elettronico: profili giuridici, tecnologici e psico-sociali ' projects.

References and Notes

1 http://www.dbgroup.unimo.it/Momis

2 Bergamaschi, S., Beneventano, D., Castano, S. and Vincini, M. (2001) 'Semantic integration of heterogeneous information sources', *J. of Data and Knowledge Engineering*, March, Vol. 36, No. 3, pp.215–249.

3 Benetti, I., Beneventano, D., Bergamaschi, S., Guerra, F. and Vincini, M. (2002) 'An information integration framework for e-commerce', *IEEE Intelligent Systems Magazine*, January/February.

4 IBM Web Service Architecture Team (2000) *Web Services Architecture Overview – The Next Stage of Evolution for E-Business*, September.

5 W3C (2002) 'Simple Object Access Protocol (SOAP) 1.2', *W3C Working Draft*, 26 June.

6 Beneventano, D., Bergamaschi, S., Bianco, D. Guerra, F. and Vincini, M. (2002) 'SI-Web: a web-based interface for the MOMIS project', *SEBD 2002*, Isola d'Elba, Italy.

7 Curbera, F. Duftler, M., Khalaf, R., Nagy, W., Mukhi, N. and Weerawarana, S. (2002) 'Unravelling the web services web: an introduction to SOAP', *WSDL and UDDI*, IEEE Internet Computer, March-April, pp.86-93.

8 Kreger, H. (2001) **'**Web services conceptual architecture', *(WSCA 1.0)*, May IBM Software Group, Available from: http://www-3.ibm.com/software/solutions/webservices/pdf/WSCA.pdf

9 W3C (2001) 'Web services description language', *(WSDL) 1.1*, W3C Note, 15 March.

10 Miller, G. (1995) 'Wordnet: a lexical database for English', *Communications of the ACM*, Vol. 38, No. 11, pp.39–41.

11 Beneventano, D., Bergamaschi, S., Sartori, C. and Vincini, M. (1997) 'ODB-tools: a description logics based tool for schema validation and semantic query optimization in object oriented databases', *Proceedings of IEEE Int. Conference on Data Engineering (ICDE-97)*, Birmingham, UK.

12 Nonaka I. and Takeuchi H, (1995) *The Knowledge Creating Company*, Oxford University Press, New York.

13 Nonaka and Takeuchi [12] precisely state that reconfiguration of existing information through sorting, adding, combining and categorising of explicit knowledge can lead to new knowledge.

14 In very short, efficiency is related to time and efforts spent in retrieving workable information; efficacy, instead, concerns the "quality" of the decision taken on the basis of the retrieved information.

15 Li, C., Yerneni, R., Vassalos, V., Garcia-Molina, H., Papakonstantinou, Y,. Ullman, J. and Valiveti, M. (1998) 'Capability based mediation in TSIMMIS', *SIGMOD 98 Demo*, Seattle, June. 16

16 Genesereth, M.R., Keller, A.M. and Duschka, O. (1997) 'Infomaster: an information integration system, *Proceedings of 1997 ACM SIGMOD Conference*, May.

17 Nodine, M.H., Fowler, J. Ksiezyk, T., Perry, B., Taylor, M.C. and Unruh, A. (2000) 'Active information gathering in Infosleuth^{TM}', *IJCIS*, Vol. 9, Nos. 1-2, pp.3-28.

18 Sycara, K. (1999) 'In-context information management through adaptive collaboration of intelligent agents', *Intelligent Information Agents*, pp.78-99.

19 Haustein, S. (2001) 'Semantic web languages: RDF vs. SOAP serialisation', *Proceedings of the Second International Workshop on the Semantic Web – SemWeb'2001*, Hongkong, China, 1 May.