Synthesizing an Integrated Ontology

The Mediator Environment for Multiple Information Sources (Momis) supports semiautomatic building, annotation, and extension of domain ontologies.

Domenico Beneventano and Sonia Bergamaschi
University of Modena and Reggio Emilia and IEIIT-CNR, Bologna

Francesco Guerra and Maurizio Vincini
University of Modena and Reggio Emilia

To exploit the Internet's expanding data collection, current Semantic Web approaches employ annotation techniques to link individual information resources with machine-comprehensible metadata. Before we can realize the potential this new vision presents, however, several issues must be solved. One of these is the need for data reliability in dynamic, constantly changing networks. Another issue is how to explicitly specify relationships between abstract data concepts. Ontologies provide a key mechanism for solving these challenges, but the Web's dynamic nature leaves open the question of how to manage them.

The Mediator Environment for Multiple Information Sources (Momis), developed by the database research group at the University of Modena and Reggio Emilia, aims to construct synthesized, integrated descriptions of information coming from multiple heterogeneous sources. Our goal is to provide users with a global virtual view (GVV) of information sources, independent of their location or their data's heterogeneity. Such a view conceptualizes the underlying domain; you can think of it as an ontology describing the sources involved.

The Semantic Web exploits semantic markups to provide Web pages with machine-readable definitions. It thus relies on the a priori existence of ontologies that represent the domains associated with the given information sources. This approach relies on the selected reference ontology's accuracy, but we find that most ontologies in common use are generic and that the annotation phase (in which semantic annotations connect Web page parts to ontology items) causes a loss of semantics. By involving the sources themselves, our approach builds an ontology that more precisely represents the domain. Moreover, the GVV is annotated according to a lexical ontology, which provides an easily understandable meaning to content.

In this article, we use Web documents as a representative information source to describe the Momis methodology's general application. We explore the framework's main elements and discuss how the output of the integration process can be exploited to create a conceptualization of the underlying domain. In particular, our method provides a way to extend previously created conceptualizations, rather than starting from scratch, by inserting a new source.

 Published by the IEEE Computer Society 1089-7801/03/$17.00

Information Integration with ODL_{I3}

ODL_{I3} is an extension of the Object Definition Language (www.service-architecture.com/database/articles/odmg_3_0.html), which is used to define interfaces to object types that conform to the Object Data Management Group (ODMG) object model. ODL_{I3} extends ODL with constructors, rules, and relationships that are useful in the ontology-integration process — both for handling source heterogeneity and representing the global virtual view (GVV). In particular, ODL_{I3} extends ODL with several relationships that express intra- and interschema knowledge for source schemas.[1]

- `Synonym of` (SYN) relationships are defined between two terms t_i and t_j that share meanings.
- `Broader terms` (BT) relationships are defined between two terms t_i and t_j, where ti has a more general meaning than t_j. BT relationships are not symmetric.
- `Narrower terms` (NT) relationships are the opposite of BT relationships.
- `Related terms` (RT) relationships are defined between two terms t_i and t_j that are generally used together in the same context in the considered sources.

ODL_{I3} also extends ODL by adding *integrity-constraint rules*, which declaratively express `if-then` rules at both the intra- and inter-source level. ODL_{I3} descriptions are translated into the Object Language with Complements allowing Descriptive cycles (OLCD)[2,3] in order to perform inferences that will be useful for semantic integration.

Because the ontology is composed of concepts (represented as *global classes* in ODL_{I3}) and simple binary relationships, translating ODL_{I3} into a Semantic Web standard such as RDF, DAML+OIL, or OWL is a straightforward process. In fact, from a general perspective, an ODL_{I3} concept corresponds to a class in the Semantic Web standards, and ODL_{I3} relationships translate into properties. In particular, the `is-a` ODL_{I3} relationships are equivalent to `subclas-sof` in the considered Semantic Web standards. We might recognize further specific correspondences by analyzing the syntax and semantics of each standard. For example, there is the correspondence between the ODL_{I3} interface and DAML+OIL `class`. There is also a correlation between ODL_{I3}'s simple domain attributes and the DAML+OIL `DataTypeProperty` concept. Complex domain attributes further correspond to the DAML+OIL `ObjectProperty` concept (www.w3.org/TR/daml+oil-reference/). Moreover, classes are wrapped in both approaches.

References

1. S. Bergamaschi et al., "Semantic Integration of Heterogeneous Information Sources," *Data & Knowledge Eng.*, vol. 36, no. 1, 2001, pp. 215–249.
2. D. Beneventano et al., "Consistency Checking in Complex Object Database Schemata with Integrity Constraints" *IEEE Trans. Knowledge and Data Eng.*, vol. 10, no. 4, 1998, pp. 576–598.
3. D. Beneventano, S. Bergamaschi, and C. Sartori, "Description Logics for Semantic Query Optimization in Object-Oriented Database Systems," *ACM Trans. Database Systems*, vol. 28, no. 1, 2003, pp. 1–50.

The Momis Framework

Momis is a framework for extracting information and integrating heterogeneous, semistructured information sources (www.dbgroup.unimo.it/momis/).[1] Unlike other data-integration systems that follow the local-as-view approach[2] (in which each source's content is represented by a predefined global schema), Momis implements a semiautomatic methodology that follows the global-as-view approach[2] (in which we obtain global schema from the data sources). More precisely, a view over the data sources is associated with each element of the global schema; its meaning is expressed as data residing at the source. Momis uses a modified version of the Object Definition Language, called ODL_{I3}, to describe both the input (the sources) and result of the synthesis process (GVV). (See the sidebar, "Information Integration with ODL_{I3}," for more information).

Momis generates a global schema that provides an integrated GVV comprising a set of global classes that represent the information in the underlying sources and the mappings that establish the connections among the global attributes of the global schema and source schema. Given that a GVV conceptualizes a domain, we could think of it as an ontology for the integrated sources.

Building an Ontology

Figure 1 (next page) shows the information-integration process for building the GVV for a set of Web pages. The GVV-generation process has five phases:

- *Local source schemata extraction.* Wrappers generate schemas for the involved sources and translate them into the common language ODL_{I3}.
- *Local source annotation with WordNet.* The integration designer chooses a meaning for each element of a local source schema, according to the WordNet lexical ontology (www.cogsci.princeton.edu/~wn).
- *Common thesaurus generation.* Starting from the annotated local schemata, Momis constructs a set of relationships describing inter- and intraschema knowledge about classes and attributes of the source schemata.
- *GVV generation.* The Momis methodology, applied to the common thesaurus and the local schemata descriptions, generates a global schema

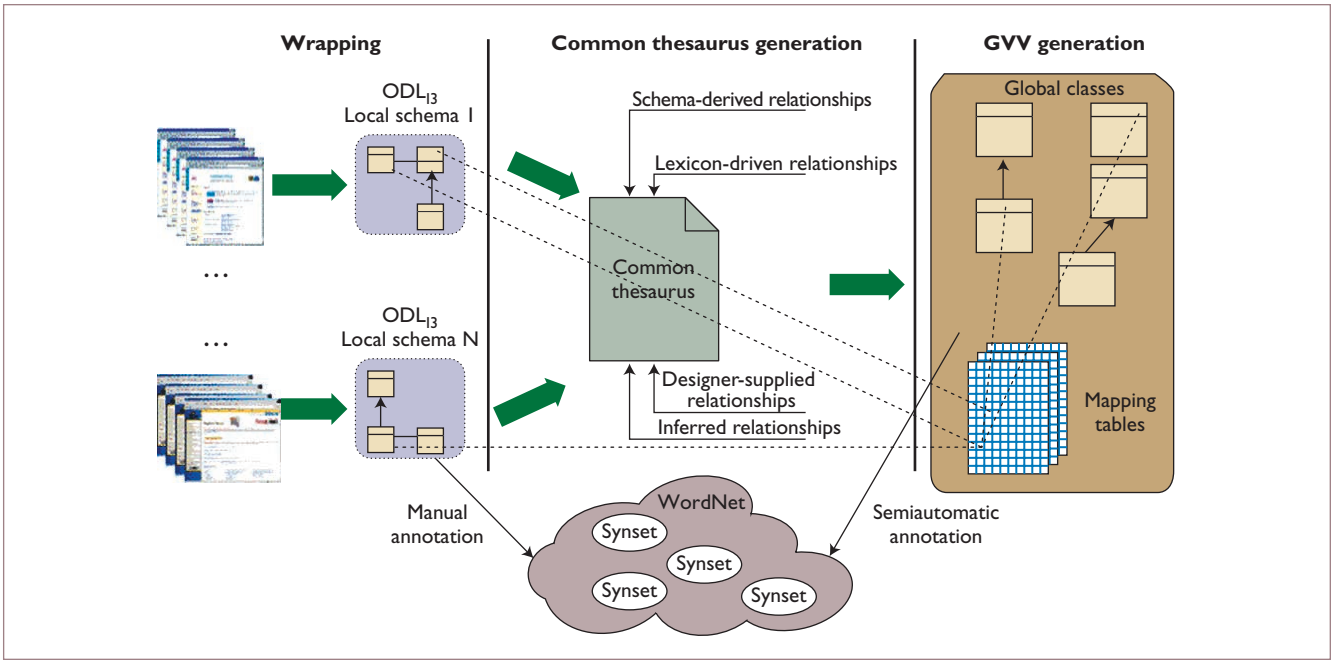

Figure 1. Overview of the ontology-generation process. Momis generates the local sources' global virtual view (GVV) via annotated local schemas derived from the WordNet lexical ontology and the common thesaurus.

Table 1. WordNet word forms and meanings.

	F1	F2	F3	...	Fn
M1	E1,1	E1,2			
M2		E2,2			
M3			E3,3		
...				...	

and sets of mappings with local schemata.

- *GVV annotation.* Exploiting the annotated local schemata and the mappings between local and global schemata, the Momis system semiautomatically assigns a meaning to each element of the global schema.

The above five phases are described in the following sections. The SI-Designer tool, with a graphical interface, supports the integration designer in all the GVV-generation-process phases.

Local Source Schemata Extraction

The first step when building an ontology with the Momis framework is to construct a semantic representation, or conceptual schema, of the information source using ODL_{I3}. We assign each source a wrapper that logically converts the underlying data structure, if there is one, into the ODL_{I3} information model. The wrapper architecture and interface are crucial because wrappers are the focal point for managing the data sources' diversity.

For conventional structured information sources (for example, relational and object-oriented databases), a schema description is always available and can be directly translated. For semistructured information sources (for example, Web pages and XML documents), a schema description is not directly available. In fact, a basic characteristic of semistructured data is that they are "self-describing" — that is, the information associated with the schema is specified in the data.[4] To manage a semistructured source, a specific wrapper has to implement an automatic methodology to extract and explicitly represent the source's conceptual schema. In the Momis framework, we developed a wrapper to translate XML and document type definition (DTD) files into ODL_{I3}.

To manage information in HTML format, which does not separate data structure from layout, we needed another extraction step. We used a commercial tool, Lixto,[5] to translate Web page content (data and data structure) into an XML file. We used the DTDs in Figure 2, built with the Lixto-generated wrapper, to acquire the ODL_{I3} descriptions shown in Figure 3.

We tested several research and commercial tools, including RoadRunner[6] and Andes,[7] but selected Lixto as the most suitable for our approach because it provides a fully visual and interactive interface that assists the user in semiautomatically creating a wrapper program.

```
University Site (UNI)
 <!ELEMENT UNI(People*)>
  <!ELEMENT People(Research_Staff*|
  School_Member*)>
  ...
  <!ELEMENT Research_Staff(name,
  e-mail, Section*, Article*)>
  <!ELEMENT Section(name, year.
  period)>
  <!ELEMENT Article(title, year,
  journal, conference)>
  <!ELEMENT School_Member(name,
  e-mail)>
  <!ELEMENT name (#pcdata)> ...
```

```
Computer Science Site (CS)
  <!ELEMENT CS(Person*)>
  ...
  <!ELEMENT Person(Professor*|
  Student*)>
  <!ELEMENT Professor(first_name,
  last_name, e-mail, Publication*)>
  <!ELEMENT Student(name, e-mail)>
  <!ELEMENT Course(denomination, Professor)>
  <!ELEMENT Publication(title, year,
  journal, editor)>
  <!ELEMENT School_Member(name,
  e-mail)>
  <!ELEMENT name (#pcdata)>...
```

Figure 2. Document type definition (DTD) fragments used to represent source schemas. Momis uses these DTDs built with a Lixto-generated wrapper to translate source content for Web pages from a university and a computer science department into XML files.

```
University Site (UNI)
...
Interface Research_Staff
(Source Un_site.dtd)
{
 attribute string name;
 attribute string email;
 attribute set < Section > section;
 attribute set < Article > article;
}

Interface Article
(Source Un_site.dtd)
{
 attribute string title;
 attribute string journal;
 attribute string conference;
 attribute string year;
}
```

```
Computer Science Site (CS)
...
Interface Professor
(Source Sc_site.dtd)
{
attribute string first_name;
attribute string last_name;
attribute string email;
attribute set < Publication > publication;
}

Interface Publication
(Source Sc_site.dtd)
{
attribute string title;
attribute string year;
attribute string journal;
}
```

Figure 3. Pieces of the university (UNI) and computer science (CS) sources in ODL_{I3}. Momis uses the XML/DTD wrapper to translate the generated DTDs into ODL_{I3} descriptions.

Local Source Annotation with WordNet

The first idea behind exploiting a lexical ontology in Momis concerns the integration process: terms for describing schemas or structures in information sources hold exploitable semantics. The second idea concerns using the integration process's results as domain ontology: for external users and applications to use our ontology, each item must have a well-known meaning. We chose the WordNet database as our lexical ontology.

For lexical semantics, WordNet starts from a conventional association between word forms — the way they are pronounced or written — and the concept or meaning they express. Lexical matrix M (shown in Table 1) synthesizes the association between each word's form (columns) and meaning (rows). Rows represent synonym sets (synsets); columns show word forms (lemma). In linguistics, a lemma is a word and all inflected forms. The lemma for *go*, for example, consists of *go*, *goes*, *going*, *went*, and *gone*. The WordNet database contains 146,350 lemma, organized into 111,223 synsets.

Entry E1,1 implies that word form F1 can express word meaning M1. If there are at least two entries in the same column, the corresponding word form is *polysemous* — that is, it can be used to represent more than one meaning, (exactly two in this case); if there are at least two entries in the same row, the two word forms are synonymous.

Given a word form F, its *i*-th meaning will be denoted by F#*i*. For example, the word form "course" has eight meanings in WordNet; the first is `course#1 = "education imparted in a series of lessons or class meetings."`

Annotation Phase

In the local source annotation phase, the integration designer manually chooses the appropriate

WordNet meaning for each conceptual schema element provided by the wrappers. Because a word form can have different meanings in WordNet, the annotation phase includes two steps: *word form choice* and *meaning choice*.

In first step, the WordNet *morphologic processor*, a component of the Momis system, aids the designer by suggesting a corresponding word form for each local element (class or attribute) of the local schema. More precisely, the morphologic processor *stems* a term — converts it to a common root form — and checks whether it exists in the WordNet database.

In the meaning-choice step, the designer can choose to map an element to zero, one, or more meanings (called *senses* in WordNet) belonging to the selected word form. Notice that designers can choose only among existing WordNet meanings: WordNet does not allow external application to extend forms with new meanings.

For a compound term, such as `shipment_received_date`, the morphologic processor extracts the component terms as the word forms — the word forms `shipment`, `received`, and `date`, in this case. The integration designer then selects the most appropriate meanings among those proposed for the single word forms. If a term is ambiguous or unavailable as a word form (if it is an abbreviation, for example), or if the proposed word form is not satisfactory, the designer can either choose another word form or search manually for a meaning. If the integration designer doesn't find an appropriate meaning in WordNet, the term is considered *unknown*, and the system derives no lexicon relationship. (See the discussion on lexicon-derived relationships in the next section.)

Summarizing via the annotation phase, the integration designer assigns each local element (class or attribute) of the local schema a name (LEN). In our example, `shipment_received_date` is an original name (of a class or attribute) and a set (that might be empty if the term is unknown) of local element meanings (LEM_i) defined by the disjunction of the term's meaning set:

$$LE = \langle LEN, \{LEM_1, \ldots, LEM_k\} \rangle, \; k=0$$

For example,

```
CS.Course          = < course, {course#1} >
UNI.Professor      = < professor,
   {professor#1} >
UNI.School_Member     = < student,
   {student#1} >
UNI.School_Member.name       = < name,
   {name#1} >
```

where

```
course#1 = "education imparted in a
   series of lessons or class meetings"
professor#1 = "someone who is a member of
   the faculty at a college or universi-
   ty"
Student#1 = "a learner who is enrolled in
   an educational institution"
name#1 = "a language unit by which a per-
   son or thing is known"
```

Momis uses the annotation phase's output to construct the lexicon relationships for the common thesaurus, which describes intra- and interschema knowledge in the form of SYN (synonym of), BT (broader terms), NT (narrower terms), and RT (related terms) relationships.

Common Thesaurus Generation

The common-thesaurus construction process incrementally adds four types of relationships.

Schema-derived relationships. Momis automatically extracts these relationships, which are expressed at the intraschema level, by analyzing each schema separately. For example, when analyzing XML data files, Momis generates BT and NT relationships from paired element identifiers (IDs) and ID references (IDREFs), and generates RT relationships from nested elements. Further extraction rules can be applied to other data models. For example, we extract intraschema RT relationships from foreign keys — sets of attributes that express a reference from one relation to another — in relational source schemas. When a foreign key is also a primary key, in both the original and referenced relation, Momis extracts BT and NT relationships, which are derived from inheritance relationships in object-oriented schemas.

Lexicon-derived relationships. These originate from the schema annotations for the lexical ontology. WordNet defines a large variety of semantic relations between its meanings. Momis derives lexical relationships between local sources terms from semantic relationships defined in WordNet between meanings. It generates these relationships using the following WordNet constructs:

- *synonymy* (similar relation) corresponds to a SYN relationship in ODL_{I^3};
- *hypernymy* (super-name relation) corresponds

to a BT relationship;
- *hyponymy* (sub-name relation) corresponds to an NT relationship;
- *holonomy* (whole-name relation) corresponds to an RT relationship;
- *meronymy* (part-name relation) corresponds to an RT relationship in ODL_{I3}, whereas RT relationships express "generic" connections between two terms and thus map both holonomy and meronymy; and
- *correlation* (two terms share the same hypernym) corresponds to a RT relationship in ODL_{I3}.

Unknown terms do not add lexicon-derived relationships to the common thesaurus, but an incorrect annotation can insert faulty relationships.

Designer-supplied relationships. To capture specific domain knowledge, designers can supply new relationships directly. This operation is crucial because the new relationships are forced to belong to the common thesaurus. If a meaningless or incorrect relationship is inserted, the subsequent integration process can produce a wrong global schema.

Inferred relationships. Momis exploits description logic techniques from ODB-Tools[8] to infer new relationships by applying subsumption computation to "virtual schemas" obtained by interpreting BT and NT as subclass relationships and RT as domain attributes. A class C1 subsumes a class C2 if the description of C2 implies the description of C1; subsumption computation is performed, in our context, by a syntactic comparison of class descriptions. For example, if `FEMALE NT PERSON`, then the class C1 with description {attribute children `PERSON`} subsumes the class C2 with description {attribute children `FEMALE`}.

Proposed relationships. In the example described in Figures 2 and 3, Momis automatically obtained and proposed the following relationships:

`CS.Professor NT CS.Person` [schema-derived]
`CS.Student NT CS.Person` [schema-derived]
`UNI.School_Member NT CS.Person` [lexicon-derived]
`UNI.Article NT CS.Publication` [lexicon-derived]
`UNI.Research_Staff NT CS.Person` [inferred]
`UNI.Research_Staff RT UNI.Article` [inferred]

If the designer accepts and confirms these relationships, they are included in the common thesaurus.

Global Virtual View Generation

The GVV consists of a set of global classes; Momis defines a mapping table to connect the global attributes of each with the source schemas' local attributes. To build the global classes, Momis must identify ODL_{I3} classes that describe the same or semantically related concepts in different sources. Therefore, the system defines *affinity coefficients*[9] for all possible pairs of ODL_{I3} classes, based on their relationships in the common thesaurus. Affinity coefficients determine the degree of matching between two classes, based on their names (*name affinity* coefficient) and attributes (*structural affinity* coefficient). Momis then calculates the linear combination of the two to create the *global affinity* coefficient, which a hierarchical clustering algorithm[10] uses to classify ODL_{I3} classes. The clustering procedure's output is an affinity tree, in which ODL_{I3} classes are the leaves and intermediate nodes have associated affinity values. Momis interactively computes the integration clusters from the affinity tree using a threshold-based mechanism for which the integration designer sets the parameters.

The generation of global classes from selected clusters is a synthesis activity that Momis performs interactively with the designer. It builds a global class GC_i and a definition for each cluster Cl_i. Once it constructs the global classes, Momis creates a corresponding global attribute set for each.

This process requires interactions with the integration designer and consists of two phases. First, the system automatically associates a set of global attributes with GC_i, corresponding to the union of local attributes for the classes belonging to Cl_i. Using the common thesaurus lattice that contains SYN, BT, and NT relationships among local attributes, the system then proposes restrictions that the designer could impose on the global attribute set.

For each global class, a persistent mapping table MT (like the one in Table 2) stores all the generated mappings. The first column in the table represents the selected global class's global attributes; the other columns represent the local classes that belong to the global class. Rows represent the global attributes. An element MT[GA][LC] represents the set of attributes of the local class LC that are mapped to the global attribute GA. The GA attribute value is a *mapping function* of the values assumed by the set of attributes MT[GA][LC]. Some simple and frequent cases of this mapping function are:

- *Identity*. The GA value is equal to the local attribute LA value; we denote this case as MT[GA][LC] = LA.

Table 2. Mapping table for global class Global2.

Global2	UNI.Article	CS.Publication
Title	Title	Title
Year	Year	Year
Journal	Journal	Journal
Conference	Conference	Null
Editor	Null	Editor

- *Concatenation*. The GA value is obtained as a concatenation of the values assumed by a set of local attributes LA_i for the local class LC; we denote this case as MT[GA][LC] = LA_1 and ... and LA_n.
- *Constant*. The GA assumes into the local class LC a constant value set by the designer; we denote this case as MT[GA][LC] = const.
- *Undefined*. The GA is undefined for the local class LC; we denote this as MT[GA][LC] = null.

In our university Web page example, the integration process gives rise to three global classes:

```
Global1: (UNI.Section, CS.Course)
Global2: (UNI.Article, CS.Publication)
Global3: (UNI.Research_Staff,
UNI.School_Member, CS.Professor,
   CS.Student)
```

Table 2 shows an example mapping table for the global class `Global2`. For simplicity, the names of local and global attributes are the same in this example, but they can differ in practice. For more information on the process of generating GVVs, see the Momis project homepage (www.dbgroup.unimo.it/Momis/).

GVV Annotation

GVV annotation assigns a global element name (GEN) and a set of global element meanings (GEM_i; a class or attribute meaning given by the disjunction of its set of meanings), to each global element (GE; class or attribute):

$$GE = \langle GEN, \{GEM_1, \ldots, GEM_p\}\rangle, p\rangle = 0$$

With the Momis project, we have developed a semiautomatic methodology for annotating a GVV.

Global Class Annotation

To semiautomatically associate an annotation with each global class, we consider the set of its "broadest" local classes (BLC_{GC}), with respect to the relationships included in the common thesaurus. (A class C is broader than a class C′ if C BT C′ or C′ NT C.) This set is defined as

$$BLC_{GC} = \{LC \in GC \mid \neg\exists y \in GC, (LC\ NT\ y) \vee (y\ BT\ LC)\}$$

For the meanings in Table 3, the designer has to use BLC_{GC} to annotate the global class by name choice and meaning choice.

Name choice. To identify each global class and its contents, the integration designer selects a name to serve as a label — particularly to identify the global class's role. The system suggests a list of possible names, but the designer can choose one that is not in the list. Therefore, a name might not be a WordNet word form. In Table 3, the designer selected the name `course` and `section` for GC_1; for GC_3, the designer chose the more significant name `university_member` over the generic proposed name `person`.

Meaning choice. For each global class, the system proposes a meaning derived from the union of the meanings of the local class names BLC_{GC}. The designer can change this set by removing some meanings or adding others. This is a crucial operation because it assigns a universally understandable meaning to each global class — that is, to each item of the ontology.

Global Attribute Annotation

We use the same approach for assigning names and meanings to attributes of a global class GC. For a given global attribute GA of GC, we consider the set of local attributes, that Momis maps into the global attribute GA on the basis of the mapping table MT. This set is denoted by LA_{GA} (local attributes mapped to GA) and is defined as:

$$LA_{GA} = \{\ LA \mid \exists LC \in GC,\ LA \in LC \wedge MT[GA][LA] \neq null\ \}$$

The set of broadest local attributes mapped to GA is denoted by BLA_{GA} and defined as:

$$BLA_{GA} = \{\ LA \in LGA \mid \neg\exists y \in LGA,\ (LA\ NT\ y) \vee (y\ BT\ LA)\}$$

On the basis of BLA_{GA}, the designer annotates the GA in the same manner as global classes. Moreover, we could use the mapping function described in the "Global Virtual View" section to develop a specific

Table 3. University global virtual view (GVV) annotation.

Global class	Local classes of GC	Broadest local class of GC	Name	Meaning
GC_1	CS.Course, UNI.Section	CS.Course, UNI.Section	course or section	course#1
GC_2	CS.Publication, UNI.Article	CS.Publication	publication	publication#1
GC_3	CS.Professor, CS.Person, UNI.School_Member, UNI.Research_Staff, CS.Student	CS.Person	University_Member	person#1

policy to automatically select meanings. For example, if GA were obtained as the concatenation of LA_1 and LA_2, the automatically selected meaning might be a hypernymy meaning of both LA_1 and LA_2.

Adding a New Source

Researchers have proposed many solutions to the challenge of supporting an ontology's evolution. In the "Related Work in Ontology Dynamics" sidebar, we briefly present the two major approaches: the *evolution approach*[11] tries to face the problem of dynamics in its total complexity, and the *versioning approach*[12] uses different versions of ontologies to reduce the problem's complexity. Alternatively, we propose using a single ontology that we update to keep it consistent with the sources that define it.

In Momis, the GVV must change whenever new sources are added or deleted or when existing sources change. Because the integration process is expensive for both the designer and the system, we propose a methodology that builds on the efforts that generated the initial GVV, rather than restarting the integration process from scratch. The GVV annotation's lexicon-based knowledge can greatly simplify the process of integrating a new source.

We approach the process as managing the integration of two schemata. The system treats the GVV's global classes as local classes and integrates them with the new source's local classes. The process introduces the following notation:

- gcNew is a global class of the new integrated schema. It has a name gcNewName and a set of global attributes $gcNewAtt_i$.
- gcOld is a global class of the old integrated schema. It has a name gcOldName and a set of global attributes $gcOldAtt_j$.
- lcNew is the local class of the new source. It has a name lcNewName and a set of local attributes $lcNewAtt_k$.

According to the Momis integration methodology, we must create a common thesaurus. In this case, the common thesaurus contains schema-derived relationships extracted from the new source and intraschema lexicon-derived relationships obtained from annotating the new source. Furthermore, we have to semantically enrich the GVV global classes using this semiautomatic annotation method. Interestingly, the GVV annotation lets us discover interschema lexical relationships, thus enriching the common thesaurus.

The next step is cluster generation, which is followed by the creation of the global classes and mapping tables. This phase provides mapping rules between GCs and new or old local classes. By integrating the old integrated schema and the new source we obtain a new integrated schema whose global classes gcNew comprise both old global classes $gcOld_1$ and new local classes $lcNew_j$:

$$gcNew = \{gcOld_1, \ldots, gcOld_p, lcNew_1, \ldots, lcNew_n\}$$

A new global class gcNew is expressed as a set of local classes (new or old) by substituting $gcOld_i$ with the respective old local classes $lcOld_{ik}$.

$$gcNew = \{lcOld_{11}, \ldots, lcOld_{1z}, \ldots, lcOld_{p1}, \ldots, lcOld_{pn}, lcNew_1, \ldots, lcNew_n\}$$

With global class generation, we observe that, using the same clustering parameters, an old global class, $lc_1,\ldots,lc_i,\ldots,lc_n$, changes only if the integration process inserts one or more new local classes ($lcNew_i$) into the global class. We therefore find three possible cases.

Scenario 1

A new global class gcNew is composed of only one old global class gcOld and one or more new local classes $lcNew_i$. For example,

$$gcNew = \{gcOld, lcNew_1, \ldots, lcNew_i, \ldots, lcNew_n\}$$

The gcNew might have new GAs generated from the new local classes' semantic contributions. Momis defines new mapping rules between a global attribute and its corresponding local attributes. In this case, GAs belonging to gcOld ($gcOldAtt_i$) can map both local classes of the old global class and new local classes. New global attributes can map only new local classes (null mappings).

The meaning of old GAs must be enriched with the meanings of the new local classes mapped by these attributes, and the meaning of new global attributes must be set according to the rules defined in the global attributes annotation.

Scenario 2

A global class of the new integrated schema is composed of only new local classes.

$$gcNew = \{lcNew_1, \ldots, lcNew_i, \ldots, lcNew_n\}$$

In this situation, the GVV is extended without interfering with the previous one.

As stated, the gcNew has a name (gcNewName) and a set of new global attributes ($gcNewAtt_i$) that each map only new local attributes. The names and meanings of the global attributes are defined following the rules stated in the global attributes annotation.

Scenario 3

A global class of the new integrated schema is composed of more than one global class of the GVV and at least one local class of the new source we are integrating.

$$gcNew = \{gcOld_1, \ldots, gcOld_p, lcNew_1, \ldots, lcNew_i, \ldots, lcNew_n\}$$

In this case, the process modifies the previous GVV; side effects can influence the applications that exploit the previously created GVV as a domain ontology. These applications must change their annotations according to the new GVV. The new global class gcNew has a name gcNewName and a set of new global attributes $gcNewAtt_i$.

Conclusions

Momis supports the semiautomatic building, annotation, and extension of domain ontologies by integrating the schemas of information sources, such as Web documents. Because the Momis ontology is tailored to represent the involved sources, the generated ontology is particularly sensitive to changes in the sources. In particular, in this article we described the methodology we adopted to extend a created ontology by inserting new sources. We faced two distinct problems: the system overload in maintaining the built ontologies and the effects of inserting new sources that could modify their existing ontologies.

Our approach attempts to solve both issues. By exploiting the semantically annotated results of previous integration processes, our methodology is less expensive than starting from scratch, but it does have some limitations. Mistakes in the previous integration process can propagate to the new GVV, for example. Moreover, our methodology tries to "adapt" the previously built-up ontology to the "new context" and so might not perfectly represent all the sources at the same level of specificity.

Current work on Momis is primarily focused on

- managing dynamics by including updating and deleting operations;
- improving the annotation phase by developing a new interface to WordNet that lets designers create new meanings and face multilingual aspects; and
- exploiting different technologies that let Momis modules communicate with each other.

Momis is a Java application based on a Corba architecture, but we have created prototypes for both an agent-based and Web service-based architecture.

The Momis methodology for extending an existing ontology was developed within the Semantic Web Agents in Integrated Economies (Sewasie) European research project (www.sewasie.org). Sewasie's goal is to design and implement an advanced search engine that enables intelligent access to heterogeneous data sources on the Web via semantic enrichment, providing the basis for structured secure Web-based communication. To achieve this goal, Sewasie creates a virtual network based on Sewasie information nodes (SINodes), which consist of managed information sources, wrappers, and a metadata repository. SINodes metadata represent GVVs of the overall information sources that each manage. We are exploring how to use the Momis methodology to create and maintain the GVV of an SINode.

Acknowledgments

This work is supported in part by the EC's 5th Framework IST program through the Sewasie project in the Semantic Web Action Line. The Sewasie consortium includes the University of Modena and Reggio Emilia (Sonia Bergamaschi, coordinator), the Universities of Aachen RWTH (M. Jarke), Roma La Sapienza (M. Lenzerini, T. Catarci), Bolzano (E. Franconi), and the

Related Work in Ontology Dynamics

Researchers have proposed many solutions to the challenge of supporting an ontology's evolution, but two major methods have emerged. The *ontology evolution*[1] approach employs the timely adaptation of an ontology as well as the consistent propagation of changes. A modification in one part of the ontology can create subtle inconsistencies in other parts of the same ontology, dependent ontologies, and applications. When a change occurs, it is vital to ensure the consistency of the ontology and all dependent artifacts. The Karlsruhe Ontology and Semantic Web framework (Kaon), for example, is based on this approach (http://kaon.semanticweb.org).

The other leading approach, *ontology versioning*, can be defined as the ability to handle changes in ontologies by creating and managing different variants.[2] Such methodologies must be able to distinguish and recognize versions, and include procedures for updating and changing ontologies. This approach is used with the Simple HTML Ontology Extensions (Shoe; www.cs.umd.edu/projects/plus/SHOE/), a small extension to HTML that lets Web page authors annotate their documents with machine-readable knowledge.

References

1. B. Motik et al., "User-Driven Ontology Evolution Management," *Proc. 13th Int'l Conf. Knowledge Eng. and Knowledge Management* (EKAW 02), LNCS 2473, Springer, 2002, pp. 285–300.
2. M. Klein and D. Fensel, "Ontology Versioning on the Semantic Web," *First Int'l Semantic Web Working Symp.*, Stanford University Press, 2001, pp. 75–91.

companies IBM Italia (G. Vetere), Thinking Networks AG (C. Engels), and CNA (A. Tavernari) as user organizations.

References

1. S. Bergamaschi et al., "Semantic Integration of Heterogeneous Information Sources," *Data & Knowledge Eng.*, vol. 36, no. 1, 2001, pp. 215–249.
2. M. Lenzerini, "Data Integration: A Theoretical Perspective," *Proc. 21st Symp. Principles of Database Systems* (PODS), ACM Press, 2002, pp. 233–246.
3. N. Guarino, "Formal Ontology in Information Systems," *Int'l Conf. Formal Ontology in Information Systems* (FOIS 98), IOS Press, 1998, pp. 3–15.
4. S. Abiteboul, P. Buneman, and D. Suciu, *Data on the Web: From Relations to Semistructured Data and XML*, Morgan Kaufmann, 2000.
5. R. Baumgartner, S. Flesca, and G. Gottlob, "Visual Web Information Extraction with Lixto," *Proc. Very Large Database Conf.* (VLDB 01), Morgan Kaufmann, 2001, pp. 119–128.
6. V. Crescenzi, G. Mecca, and P. Merialdo, "RoadRunner: Automatic Data Extraction From Data-Intensive Web Sites," *Proc. Special Interest Group on Management of Data Conf.* (SIGMOD 02), ACM Press, 2002, p. 624.
7. J. Myllymaki, "Effective Web Data Extraction with Standard XML Technologies," *Proc. 10th Int'l World Wide Web Conf.*, ACM Press, 2001, pp. 689–696.
8. D. Beneventano et al., "ODB-QOptimizer: A Tool For Semantic Query Optimization in OODB," *Proc. Int'l Conf. Data Eng.* (ICDE 97), IEEE CS Press, 1997, p. 578.
9. S. Castano, V. De Antonellis, and S. De Capitani di Vimercati, "Global Viewing of Heterogeneous Data Sources," *IEEE Trans. Data and Knowledge Eng.*, vol. 13, no. 2, 2001, pp. 277-297.
10. B. Everitt, *Cluster Analysis*, Heinemann, 1974.
11. B. Motik et al., "User-Driven Ontology Evolution Management," *Proc. 13th Int'l Conf. Knowledge Eng. and Knowledge Management* (EKAW 02), LNCS 2473, Springer, 2002, pp. 285–300.
12. M. Klein and D. Fensel, "Ontology Versioning on the Semantic Web," *Proc. 1st Int'l Semantic Web Working Symp.*, Stanford Univ. Press, 2001, pp. 75-91.

Francesco Guerra is a PhD candidate in information engineering at the University of Modena and Reggio Emilia. His main research interests include integration of heterogeneous information sources, ontologies, and the Semantic Web. Guerra received a Laurea in computer engineering from the University of Modena and Reggio Emilia. Contact him at guerra.francesco@unimore.it.

Maurizio Vincini is a research associate in information engineering at the University of Modena and Reggio Emilia. His research interests include intelligent information integration, reasoning techniques, ontologies, object-oriented database design, and query optimization. Vincini received a PhD in computer science and engineering from the University of Modena and Reggio Emilia. Contact him at vincini.maurizio@unimore.it.

Domenico Beneventano is a professor of engineering at the University of Modena and Reggio Emilia. His research current interests include intelligent information integration, particularly with structured and semistructured data. Beneventano received a Laurea in electronic engineering and a PhD in computer science and electronic engineering, both from the University of Bologna. Contact him at beneventano.domenico@unimore.it.

Sonia Bergamaschi is a full professor in the Department of Information Engineering at the University of Modena and Reggio Emilia and coordinator of the EU Semantic Web and Agents in Integrated Economies (Sewasie) project. Her research interests include intelligent information integration, knowledge representation, and management with very large databases. She is a member of the IEEE Computer Society and the ACM. Contact her at bergamaschi.sonia@unimore.it.