Exploiting Schema Knowledge for the Integration of Heterogeneous Sources*

S. Bergamaschi[1], S. Castano[2], S. De Capitani di Vimercati[2],
S. Montanari[1], M. Vincini[1]

(1) University of Modena
DSI - Via Campi 213/B - 41100 Modena
e-mail: [sonia,montanar,vincini]@dsi.unimo.it

(2) University of Milano
DSI - Via Comelico, 39 - 20135 Milano
e-mail: [castano,decapita]@dsi.unimi.it

Contact Author: Sonia Bergamaschi

Abstract

Information sharing from multiple heterogeneous sources is a challenging issue which ranges from database to ontology areas. In this paper, we propose an intelligent approach to information integration which takes into account of semantic conflicts and contradictions, caused by the lack of a common shared ontology. Our goal is to provide an *integrated access* to information sources, allowing a user to pose a single query and to receive a single unified answer.

We propose a "semantic" approach for integration where the conceptual schema of each source is provided, adopting a common standard data model and language, and *Description Logics* plus *clustering techniques* are exploited.

Description Logics is used to obtain a semi-automatic generation of a Common Thesaurus (to solve semantic heterogeneities and to derive a common ontology). *Clustering techniques* are used to build the *global schema*, i.e. the *unified* view of the data to be used for query processing.

keywords: Intelligent Information Integration, semantic integration, mediators, heterogeneous databases, ODMG-93.

1 Introduction

In the continuing quest to provide uniform access to distributed information, the problem of integrating information from heterogeneous sources is becoming more and more relevant.

Main problems to be faced in integrating information coming from distributed sources are related to structural and implementation heterogeneity (including differences in hardware platforms, DBMS, data models and data languages), and to the lack of a common ontology, which leads to semantic heterogeneity. Semantic heterogeneity occurs when different names are employed to represent the same information or when different modeling constructs are used to represent the same piece of information in different sources [29, 31, 35].
Some approaches have been recently proposed in the literature for the extraction and integration

*This research has been partially funded by the Basi di Date Evolute - MURST 40% project.

of conventional structured databases [42, 10, 13] and semi-structured data [14, 30, 37] taking into account semantic heterogeneity. Data integration architectures are usually based on *mediators* [41], where knowledge about data of multiple sources is combined to provide a global view of the underlying data. Two fundamental approaches have emerged in the literature: *structural* and *semantic*. There are many projects following the 'structural approach' [3, 23]. A well-know project following the structural approach, is the TSIMMIS ("The Stanford-IBM Manager of Multiple Information Sources") project under development at the Stanford University [21, 27]. The TSIMMIS approach towards *mediators* development can be characterized as follows:

- OEM, the data description language for sources is different from conventional OO languages, as it is a *self-describing model* where each data item has an associated descriptive label, *without a strong typing system*;
- *semantic knowledge is effectively encoded in the MSL (Mediator Specification Language) rules* that perform/enforce the integration at the mediator level.

Let us recall the fundamental argument in favor of the *structural approach* (considering TSIMMIS as a target system):

- the generality and conciseness of OEM and MSL make the 'structural' approach a good candidate for the integration of widely heterogeneous and semi-structured information sources; this is an improvement, since:
  - in traditional data models, a client must be aware of the schema in order to pose a query, while in TSIMMIS the structure of the information is discovered as queries are posed;
  - a conventional OO language breaks down in such a case, unless one defines an object class for every possible type of irregular object.

On the other hand a major drawback in such on approach is that *we lose the notions of schema, class, extension* and thus:

- when integration involves a large amount of objects we cannot use optimization access techniques, developed in the database research area, based on the concepts mentioned above;
- only pre-defined queries (at the mediator level) are executable.

Many other projects have been proposed following a *semantic* approach [1, 2, 7, 10, 13, 28]. This approach can be characterized as follows:

- for each source, meta-data, i.e. conceptual schema, must be available;
- semantic information is encoded in the schema;
- a common data model as the basis for describing sharable information must be available;
- partial or total schema unification is performed.

Let us introduce some fundamental arguments in favor of a 'semantic approach' based on conventional OO data models:

1. having a schema available, all the queries consistent with that schema are supported and the semantic knowledge encoded in it permits the efficient extraction of information;

2. the schema nature of conventional OO models together with classification, aggregation and generalization primitives allows the organization of extensional knowledge;
3. the adoption of a *type as a set* semantics for a schema make it possible to check the consistency of instances with respect to their descriptions;
4. a relevant effort has been devoted to develop OO standards: CORBA [40] for object exchanging among heterogeneous systems; ODMG-93 (including ODM model and ODL language for schema description; OQL language as query language) for object oriented databases [20].

Furthermore, let us argue that the argument in favor of the structural approach can be satisfied as well with a *semantic approach* with a *weak class description notion* [7]:

- The adoption of an *open world semantics* typical of the description Logics approach [9, 12, 18, 43], for classes descriptions allows semi-structured data integration: objects of a class share a common minimal structure, but can have further additional properties.

In this paper, we propose a semantic approach to the integration of heterogeneous information. The approach follows the *semantic* paradigm, in that conceptual schemas of an involved source are considered, and a common data model (ODM$_{I^3}$) and language (ODL$_{I^3}$) are adopted to describe sharable information. ODM$_{I^3}$ and ODL$_{I^3}$ are defined as a subset of the corresponding ODMG-93 [20] ODM and ODL. A Description Logics `ocdl` (object description language with constraints [8]) is used as a kernel language and ODB-Tools as the supporting system [4, 5].

Information integration in MOMIS is based on schemas and is performed through an *extraction and analysis* process followed by a *unification* process. The extraction and analysis process is devoted to the construction of a Common Thesaurus of *terminological relationships* based on ODL$_{I^3}$ schemas descriptions and to the formation of clusters (by means of clustering techniques) of ODL$_{I^3}$ classes describing similar information in different schemas. The unification process builds an integrated global schema for the analyzed sources by integrating ODL$_{I^3}$ classes in a given cluster. Issues regarding query processing and optimization using the knowledge embedded in the global schema are briefly discussed, also w.r.t. aspects related to extensional knowledge. We developed an I^3 system, called MOMIS (Mediator envirOnment for Multiple Information Sources) implementing our approach.

The use of the `ocdl` Description Logics together with hierarchical clustering techniques are the original contributions of the approach to enhance a semi-automated integration process. Description Logics allows us to interactively set-up the Thesaurus by deriving explicit terminological relationships from ODL$_{I^3}$ schema descriptions and by inferring new relationships out of the explicit ones. Moreover, optimization of the queries against the global schema is possible using description logics. Clustering techniques allow the automated identification of ODL$_{I^3}$ classes in different source schemas that are semantically related and constitute a cluster candidate to be unified into a unique class in the global schema.

Information integration is a difficult, time-consuming, and knowledge intensive process, and the availability of an automated support is a valuable aspect, especially in case of large scale integration, as it is more and more frequent with the increasing number of information sources available in global information systems.

The paper is organized as follows. In Section 2, we introduce the ODL$_{I^3}$ language and we outline the approach to schema integration together with a running example used throughout the paper. In Section 3, we describe the construction of a Common Thesaurus of terminological relationships. In Section 4, we illustrate affinity-based techniques for the analysis of ODL$_{I^3}$ schemas. In Section 5, we illustrate the clustering process for the formulation of group of classes with affinity and, in Section 6, we describe the unification process for building the global schema. In Section 7 we show the effectiveness of the global schema w.r.t. query processing and optimization. In Section 8 we discuss previous work. Finally, in Section 9 we give our concluding remarks.

2 The MOMIS approach to information integration

In this section we describe the architecture of the MOMIS system and the phases of the approach to intelligent schema integration.

2.1 The MOMIS architecture and the ODL$_{I^3}$ language

In Fig. 1 the architecture of the MOMIS system is shown. With respect to the literature, this can be considered as an example of powerful I^3 system [15] and follows the TSIMMIS architecture [27]. Above each source lies a translator (called *wrapper*) responsible for translating the structure of the data source into the common ODL$_{I^3}$ language. In a similar way, the *wrapper* performs a translation of the query from the OQL$_{I^3}$ language to a local request to be executed by a single source. Above the *wrapper* there is a *mediator*, a software module that combines, integrates, and refines ODL$_{I^3}$ schemas received from the *wrappers*. In addition, the mediator generates the OQL$_{I^3}$ queries for the *wrappers*, starting from the query request formulated with respect to the global schema. The mediator module is obtained by coupling a *semantic approach*, based on a Description Logics component, i.e., ODB-Tools Engine, and an extraction and analysis component, i.e., Schema Analyzer and Classifier, together with a minimal ODL$_{I^3}$ interface.

The MOMIS system allows the automatic or semi-automatic generation of mediators from high level descriptions of the information processing they need to do. We rely on a semantic high level language (ODL$_{I^3}$) for the mediator description and on two general purpose components: cluster generator and ODB-Tools.
We obtain the following benefits:

1. The language is an extension with rules of the structural part of the ODMG-93 standard;
2. the language allows a declarative description of schemata and mapping rules;
3. the language is interpreted as a Description Logics and has an *open world* semantics approach;
4. tools for query optimization and consistency check are available at the mediator level;
5. tools for computing class affinity and clustering, and for building the global schema are available.

Furthermore, ODB-Tools is the basis of the Query Manager which allows us to generate in an automated way the translation of a global user query into local queries.

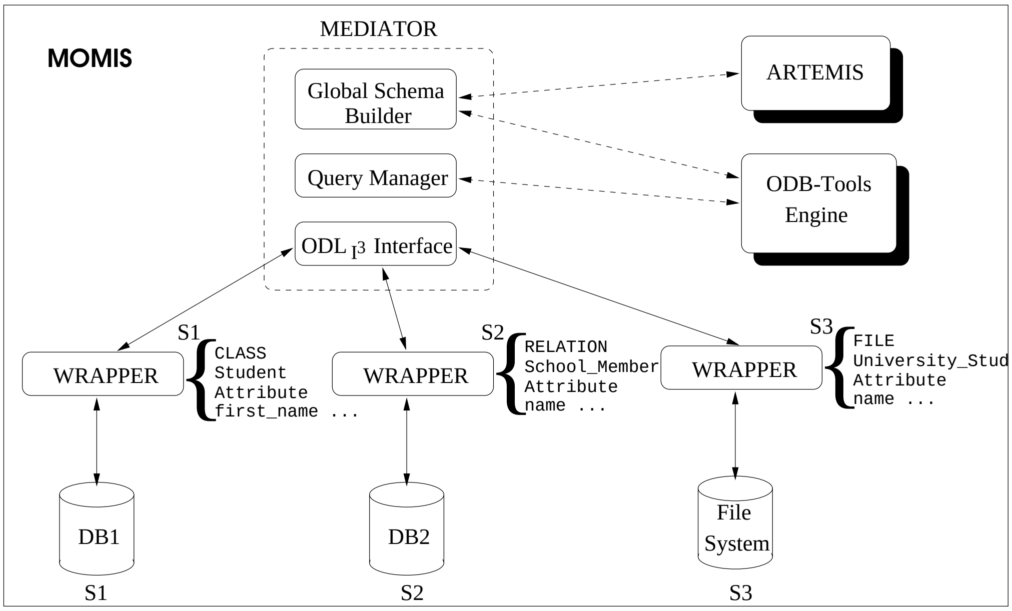

Figure 1: Architecture of the MOMIS I^3 system

In order to easily communicate source descriptions between wrappers and mediator engine, we introduce a data description language, called ODL$_{I^3}$.

According to recommendations of [15], and to the diffusion of the object data model (and its standard ODMG-93), ODL$_{I^3}$ is very close to the ODL language, supporting requirements of our intelligent information integration system. ODL$_{I^3}$ is a source independent language used by the mediator to manage the system in a common way (we suppose to deal with different source types, such as relational databases, object-oriented databases, files). The main extension w.r.t. ODL is the capability of expressing two kind of rules: *if then* rules, expressing in a declarative way integrity constraints intra and inter sources, and *mapping* rules between sources. Furthermore, ODL$_{I^3}$ allows us to express terminological relationships intra and inter sources. It will be the wrapper task to translate the data description language of any particular source into ODL$_{I^3}$ description, and to add information needed by the mediator, such as the source name and type. In general, the wrapper will provide the description of a subset of the source schema, namely the source subschema to be integrated and thus accessible for the mediator. Let $S = \{S_1, S_2, \ldots, S_N\}$ be a set of schemas of N heterogeneous sources that have to be integrated. According to ODL$_{I^3}$, each source schema S_i is a collection of *classes*,[1] $S_i = \{c_{1i}, c_{2i}, \ldots, c_{mi}\}$. A class $c_{ji} \in S_i$ is characterized by a name and a set of attributes, $c_{ji} = \langle n_{c_{ji}}, A(c_{ji})\rangle$. Each attribute $a_h \in A(c_{ji})$, with $h = 1, \ldots, n$, is defined as a pair, $a_h = \langle n_h, d_h\rangle$, where n_h is the name and d_h is the domain associated with a_h, respectively.

Thereafter, we show an example of ODL$_{I^3}$ class descriptions, showing a relation and an object class, respectively, while the syntax of the language is illustrated in Appendix A.

```
interface School_Member              interface Student : CS_Person
( source relational University       ( source object Computer_Science
  extent School_Member                 extent Students )
```

[1]In the following, for simplicity, we refer to classes, even if relations and files are supported by the language.

```
  key name )
{ attribute string first_name;
  attribute string last_name;
  attribute string faculty;
  attribute integer year; };
```

```
{ attribute integer year
  attribute set<Course> takes;
  attribute string rank; };
```

2.2 Overview of the approach

The MOMIS approach to intelligent schema integration is articulated in the following phases:

1. *Generation of a Common Thesaurus.*
   The objective of this step is the construction of a Common Thesaurus of *terminological relationships* for schema classes belonging to different source ODL$_{I^3}$ schemas. Terminological relationships are derived in a semi-automatic way, by analyzing the structure and context of classes in the schema, by using ODB-Tools and the Description Logics techniques.
2. *Affinity analysis of* ODL_{I^3} *classes.*
   Terminological relationships in the Thesaurus are used to evaluate the level of *affinity* between classes intra and inter sources. The concept of affinity is introduced to formalize the kind of relationships that can occur between classes from the integration point of view. The affinity of two classes is established by means of affinity coefficients based on class names and attributes.
3. *Clustering* ODL_{I^3} *classes* .
   Classes with affinity in different sources are grouped together in clusters using hierarchical clustering techniques. The goal is to identify the classes that have to be integrated since describing the same or semantically related information.
4. *Generation of the mediator global schema.*
   Unification of affinity clusters leads to the construction of the global schema of the mediator. A class is defined for each cluster, which is representative of all cluster's classes and is characterized by the union of their attributes. The global schema for the analyzed sources is composed of all classes derived from clusters, and is the basis for posing queries against the sources. For a semi-automatic generation of the global schema, schemata of the sources, mapping rules, and ODB-Tools are exploited

Each phase of the integration process is described in the following sections. Once the mediator global schema has been constructed, it can be exploited by the users for posing queries. The information on the global schema is used by the *Query Manager* module of MOMIS for query reformulation and for semantic optimization using ODB-Tools, as discussed in Section 7.

2.3 Running example

Fig. 2 presents the example that will be used in the remainder of this paper. We consider three different sources. The first source is a relational database, `University` (S_1), containing information about the staff and the students of a given university. There are five relations: `Research_Staff`, `School_Member`, `Department`, `Section` and `Room`.

For a given professor (in `Research_Staff`) his department (`dept_code`) and his section

University source ($\mathbf{S_1}$)

```
Research_Staff(first_name,last_name,relation,email,dept_code,section_code)
School_Member(first_name,last_name,faculty,year)
Department(dept_name,dept_code,budget,dept_area)
Section(section_name,section_code,length,room_code)
Room(room_code,seats_number,notes)
```

Computer_Science source ($\mathbf{S_2}$)

```
CS_Person(name)
Professor:CS_Person(title,belongs_to:Division,rank)
Student:CS_Person(year,takes:set⟨Course⟩,rank)
Division(description,address:Location,fund,sector,employee_nr)
Location(city,street,number,county)
Course(course_name,taught_by:Professor)
```

Tax_Position source ($\mathbf{S_3}$)

```
University_Student(name,student_code,faculty_name,tax_fee)
```

Figure 2: Example with three source schemas

(`section_code`) are stored. In the relation `School_Member` the information `name`, `year` and `faculty` about students enrolled at the university are stored.

The second source `Computer_Science` (S_2) contains information about people belonging to the computer science department of the same university, and is an object-oriented database. There are six classes: `CS_Person`, `Professor`, `Student`, `Division`, `Location` and `Course`. Information are quite similar to the first source: it stores data on professors and students, also giving the possibility to retrieve the division of a given professor. This division may be part of another department, being a logical specialization of `Department`. The class `Location` maintains the division address. With respect to students, we may know the courses they take and their year.
A third source is also available, `Tax_Position` (S_3), derived from the registrar's office. It consists of a file system, storing information about student's tax_fees. For the complete source descriptions see appendix B.

3 Generation of a Common Thesaurus

The goal of this phase is the construction of a Thesaurus of terminological relationships describing common knowledge about ODL_{I^3} classes and attributes describing the source schemas. For this reason, it is called Common Thesaurus. The following kinds of terminological relationships are specified in the Common Thesaurus:

- SYN (Synonym-of), defined between two terms t_i and t_j, with $t_i \neq t_j$, that are considered synonyms, i.e., that can be interchangeably used in every considered source, without changes in meaning. An example of SYN relationship in our example is ⟨`Section` SYN `Course`⟩.
- BT (Broader Terms), or hypernymy, defined between two terms t_i and t_j such as t_i has a broader, more general meaning than t_j. An example of BT relationship in our example is ⟨`CS_Person` BT `Student`⟩. BT relationship is not symmetric. The opposite of BT is NT (Narrower Terms), that is t_i BT $t_j \rightarrow t_j$ NT t_i.

- RT (Related Terms), or positive association, defined between two terms t_i and t_j that are generally used together in the same context. For example, we can have the following relationship ⟨`Student` RT `Course`⟩.

Discovering terminological relationships encoded in ODL$_{I^3}$ schemas is a semi-automatic process, enforced by the interaction between ODB-Tools and the designer. The whole process, that leads to the definition of a Common Thesaurus starting from the the ODL$_{I^3}$ source schema descriptions, is shown in Fig. 3 and is articulated in the following steps:

1. *Automated extraction of relationships from ODL$_{I^3}$ schemas*: exploiting ODB-Tools capability and semantically rich schema descriptions, BT, NT, and RT can be automatically discovered. In particular, by translating ODL$_{I^3}$ into `ocdl` descriptions, ODB-Tools infers BT/NT relationships between classes from generalization hierarchies, and RT relationships from aggregation hierarchies, respectively. Other RT relationships are extracted from source schemas to represent the aggregation between a class and each associated attribute. With relational source schemas, RT relationships are also extracted representing foreign keys.
2. *Integration/revision of relationships*: by interacting with the tool, the designer can supply additional terminological relationships not extracted in the previous step. Example of terminological relationships that can be interactively supplied are those regarding synonyms and domain-specific knowledge in general.
3. *Validation of relationships*: in this step, ODB-Tools is employed to validate terminological relationships involving attribute names in the Thesaurus. Validation is based on the compatibility of domains associated with attributes. In this way, *valid* and *invalid* terminological relationships are distinguished. In particular, let $a_t = \langle n_t, d_t \rangle$ and $a_q = \langle n_q, d_q \rangle$ be two attributes. The following checks are executed on attribute's name relationships using ODB-Tools:
   - $\langle n_t$ SYN $n_q \rangle$: the relationship is marked as valid if d_t and d_q are equivalent, or if one is more specialized than the other;
   - $\langle n_t$ BT $n_q \rangle$: the relationship is marked as valid if d_t contains or is equivalent to d_q;
   - $\langle n_t$ NT $n_q \rangle$: the relationship is marked as valid if d_t is contained in or equivalent to d_q.
4. *Inferring new relationships*: starting from the valid explicit relationships obtained in the previous steps, a new set of terminological relationships is inferred by ODB-Tools. The Thesaurus containing both explicit and inferred relationships constitutes the so-called common Thesaurus for the analyzed source schemas.

Example 1 Fig. 3 represents the inputs and the outputs of any step of the Thesaurus construction process, referred to our University example. The list of relationships inferred from explicit ones is shown in Fig. 4[2], while a graphical representation of the Common Thesaurus for $\mathtt{S}_1$, $\mathtt{S}_2$, and $\mathtt{S}_3$ is reported in Fig. 5. Here thick arrows represent BT/NT relationships, thin arrows represent RT relationships, dashed arrows represent inferred relationships, while solid ones represent extracted/supplied relationships.

[2]For the sake of simplicity, only relationships between class names are reported in Fig. 4.

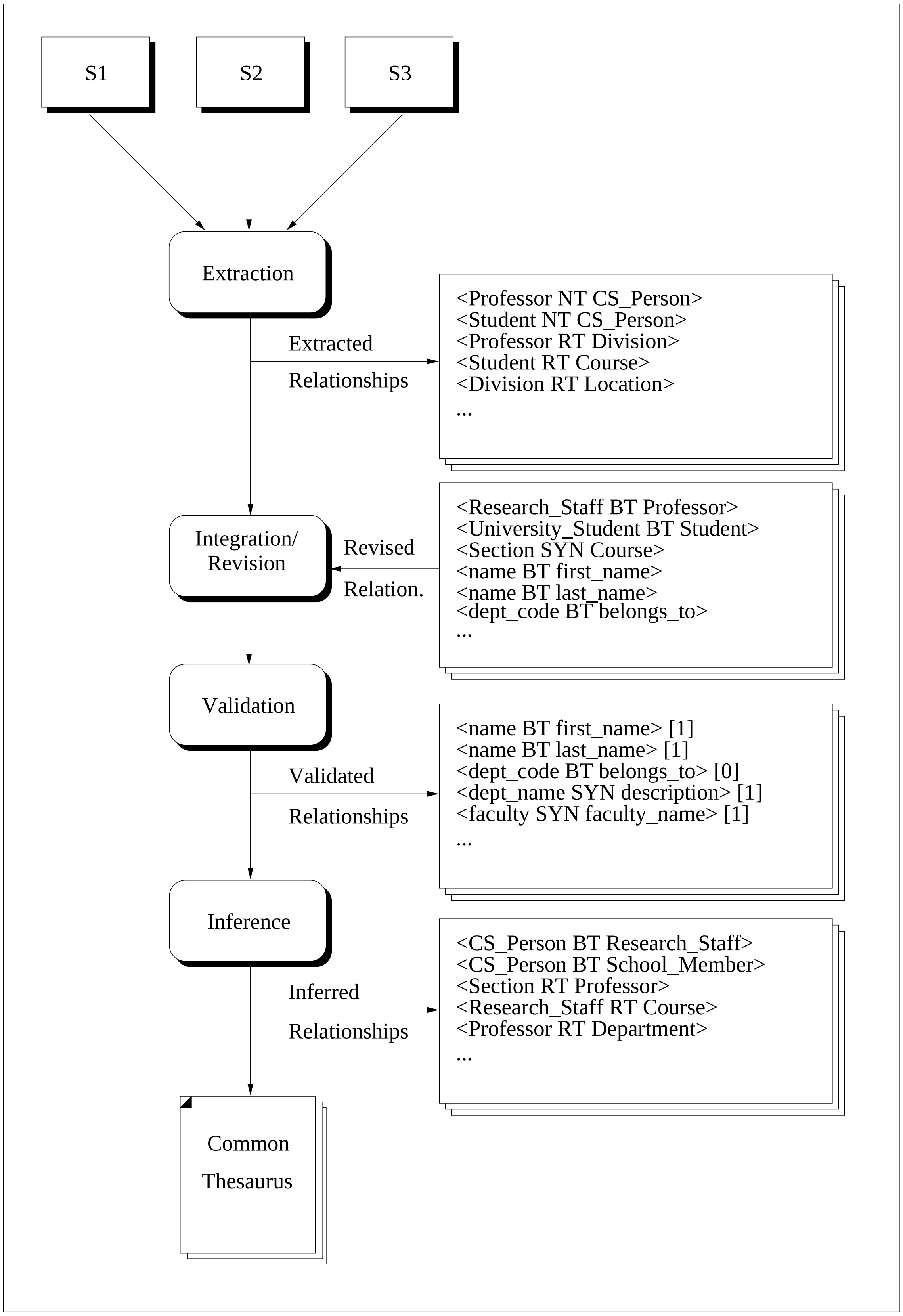

Figure 3: The generation process of a Common Thesaurus.

Explicit relationships (Step 1,2)	**Inferred relationships (Step 4)**
⟨CS_Person BT Student⟩	⟨CS_Person BT Research_Staff⟩
⟨CS_Person BT Professor⟩	⟨CS_Person BT School_Member⟩
⟨School_Member BT Student⟩	⟨Section RT Professor⟩
⟨Research_Staff BT Professor⟩	⟨Research_Staff RT Course⟩
⟨Section SYN Course⟩	⟨Professor RT Department⟩
⟨Department BT Division⟩	⟨Professor RT Section⟩
⟨Student RT Course⟩	⟨Course RT Room⟩
⟨Course RT Professor⟩	⟨Student RT Section⟩
⟨Research_Staff RT Department⟩	⟨CS_Person BT University_Student⟩
⟨Research_Staff RT Section⟩	
⟨Professor RT Division⟩	
⟨Division RT Location⟩	
⟨University_Student BT Student⟩	
⟨Section RT Room⟩	

Figure 4: Explicit/Inferred relationships

4 Affinity analysis of ODL$_{I^3}$ classes

To integrate the ODL$_{I^3}$ schemas of the different sources into a global schema, we need techniques for identifying the classes that describe the same or semantically related information in different source schemas. For this purpose, we compare ODL$_{I^3}$ classes by means of *affinity coefficients* which allow us to determine the level of semantic relationship between two classes. In particular, we compare ODL$_{I^3}$ classes with respect to their names (using a *Name Affinity coefficient*) and attributes (using a *Structural Affinity coefficient*), in order to evaluate their level of semantic relationship (using a *Global Affinity coefficient*).

The evaluation of the affinity coefficients relies on the terminological relationships stored in the Common Thesaurus. For affinity evaluation purposes, we consider the Common Thesaurus as an associative network of terms [26], where nodes correspond to terms and labeled edges between nodes to terminological relationships. Two terms have affinity if they are connected through a path in the Thesaurus. To compute a quantitative measure of affinity between terms, a strength $\sigma_{\Re}$ is assigned to each type of terminological relationship $\Re$ in the Thesaurus, with $\sigma_{\text{SYN}} \geq \sigma_{\text{BT/NT}} \geq \sigma_{\text{RT}}$. In the following, when necessary, we use notation $\sigma_{ij_{\Re}}$ to denote the strength of the terminological relationship $\Re$ for terms t_i and t_j in the Thesaurus. In our experimentation, we use $\sigma_{\text{SYN}} = 1$, $\sigma_{\text{BT}} = \sigma_{\text{NT}} = 0.8$ and $\sigma_{\text{RT}} = 0.5$. The level of affinity of two terms depends on the length of the path, on the type of relationships involved in this path, and on their strength, according to the following *affinity function*.

Definition 1 (Affinity Function) The affinity function $A_{thes}(t_i, t_j)$ of two terms t_i, t_j, returns the measure of the strength of the relationship involved in the shortest path defined between t_i and t_j in the Thesaurus as follows:

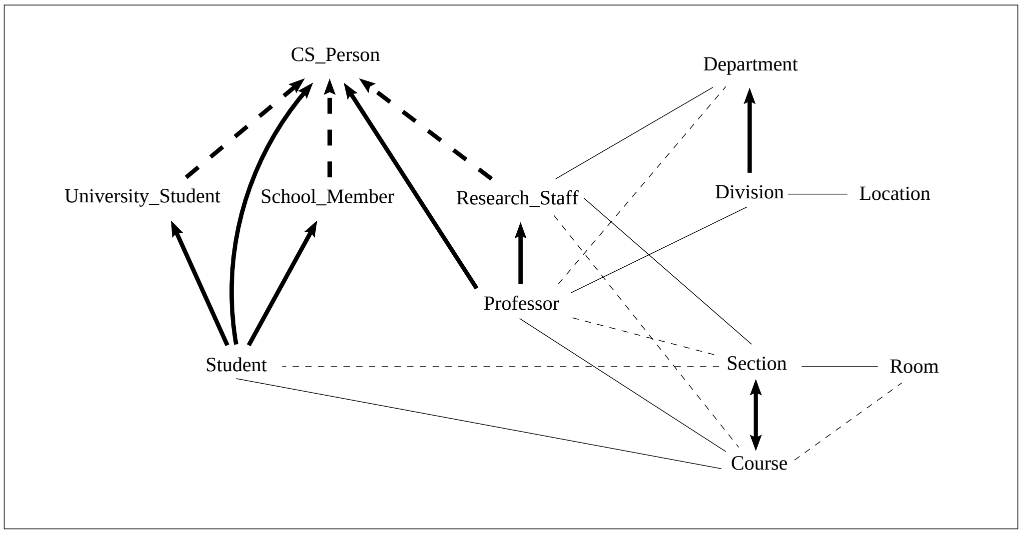

Figure 5: Common Thesaurus for $\mathbf{S_1}$, $\mathbf{S_2}$, and $\mathbf{S_3}$.

$$A_{thes}(t_i, t_j) = \begin{cases} 1 & \text{if } t_i = t_j \\ \sigma_{i1_{\Re}} \cdot \sigma_{12_{\Re}} \cdot \ldots \cdot \sigma_{(k-1)j_{\Re}} & \text{if } t_i \rightarrow^k t_j \\ 0 & \text{otherwise} \end{cases}$$

where notation $t_i \rightarrow^k t_j$ denotes the shortest path of length k, with $k \geq 1$, between t_i and t_j in the Thesaurus.

For any pairs of terms, $A_{thes} \in [0, 1]$. The affinity of two terms is 0 if a path does not exist between them in the Thesaurus, and it is 1 if the terms coincide. In remaining cases, the longer the path defined between two terms, the lower the affinity of these terms in the Thesaurus. For a given path length, the higher the strength of the involved relationships, the greater the affinity of the considered terms.

Example 2 Consider the Common Thesaurus illustrated in Fig. 5. A path exists between `School_Member` and `University_Student`, that is, `School_Member` $\rightarrow^{\mathrm{NT}}$ `CS_Person` $\rightarrow^{\mathrm{BT}}$ `University_Student`. Therefore, A_{thes}(`School_Member`, `University_Student`) $= 0.8 \cdot 0.8 = 0.64$.

Definition 2 (Term affinity) Two terms t_i, t_j have affinity, denoted by $t_i \sim t_j$, if their affinity function returns a value greater than or equal to a defined threshold $\alpha > 0$, that is, $t_i \sim t_j \leftrightarrow A_{thes}(t_i, t_j) \geq \alpha$

For example, suppose that $\alpha = 0.4$.

In this case, because of A_{thes}(`Student`, `University_Student`) $= 0.64$, we can conclude that `Student` $\sim$ `University_Student`.

4.1 Affinity coefficients

In this section, we give the definitions of the *Name Affinity coefficient*, *Structural Affinity coefficient*, and *Global Affinity coefficient* for two ODL_{I^3} classes c and c' belonging to sources S and S' respectively.

The Name Affinity coefficient measures classes affinity with respect to their names, while the structural affinity considers class attributes. These coefficients are defined as follows.

Definition 3 (Name Affinity coefficient) The Name Affinity coefficient of two classes c and c', denoted by $NA(c$,$c')$, is the measure of the affinity of their names n_c and $n_{c'}$ in the Thesaurus as follows.

$$NA(c,c') = \begin{cases} A_{thes}(n_c, n_{c'}) & \text{if } n_c \sim n_{c'} \\ 0 & \text{otherwise} \end{cases}$$

Example 3 Consider the `University_Student` and `School_Member` classes. We have that $NA(\texttt{University\_Student}, \texttt{School\_Member}) = 0.64$

Definition 4 (Structural Affinity coefficient) The Structural Affinity coefficient of two classes c and c', denoted by $SA(c,c')$, is the measure of the affinity of their attributes computed as follows:

$$SA(c,c') = \frac{2\cdot \mid \{(a_t, a_q) \mid a_t \in A(c), a_q \in A(c'), n_t \sim n_q\} \mid}{\mid A(c) \mid + \mid A(c') \mid} \cdot F_c$$

$$\begin{aligned} F_c &= \tfrac{|\{x \in C| \ flag(x)=1\ \}|}{|C|} \\ \mathrm{C} &= \{(a_t, a_q) \mid a_t \in A(c), a_q \in A(c'),\ n_t \sim n_q\} \end{aligned}$$

where notation $flag(x) = 1$ stands for a positive result and C is the set of validable attribute pairs.

The Structural Affinity coefficient is evaluated using the Dice's function, refined by a control factor F_c, which returns an affinity value in the range $[0,1]$ proportional to the number of attributes that have affinity in the considered classes.

The F_c term realizes a domain check on each terminological relationship between attributes in the Thesaurus (the check is the same as described in the terminological validation phase, section 3): its value will be the ratio of positive checks to number of checkable attributes. The greater the number of attributes with affinity in the considered classes, and the greater the number of positive control results, the higher the Structural Affinity coefficient for the classes. The value 0 indicates the absence of attributes with affinity in the considered classes, while the value 1 indicates that all attributes defined in both classes have affinity.

Example 4 Consider the source schemas in Fig. 2. Consider now class `Student` in S_2 and class `University_Student` in S_3. We have that $SA(\texttt{Student}, \texttt{University\_Student}) = \frac{2\cdot 3}{4+4} \cdot \frac{1}{1} = 0.75$.

Definition 5 (Global Affinity coefficient) The Global Affinity coefficient of two classes c and c', denoted by $GA(c,c')$, is the measure of their affinity computed as the weighted sum of the Name and Structural Affinity coefficients as follows:

$$GA(c,c') = \begin{cases} w_{NA} \cdot NA(c,c') + w_{SA} \cdot SA(c,c') & \text{if } NA(c,c') \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

where weights w_{NA} and w_{SA}, with $w_{NA}, w_{SA} \in [0,1]$ and $w_{NA} + w_{SA} = 1$, are introduced to assess the relevance of each coefficient in computing the global affinity value.

Procedure Hierarchical Clustering
/* **Input:** K classes to be analyzed */

1. Compute all pairwise Global Affinity coefficients $GA(c, c')$.
2. Place each class into a cluster of its own.
3. **Repeat**

   Select the pair c_h, c_k of current clusters
   with the highest affinity coefficient in M, $M[h, k] = max_{i,j} M[i, j]$;
   Form a new cluster by combining c_h, c_k;
   Update M by deleting the rows and columns corresponding to c_h and c_k;
   Define a new row and column for the new cluster.

   until rank of M is greater than 1.

end procedure.

Figure 6: Hierarchical clustering procedure

Weights in $GA(c, c')$ allow the analyst to differently stress the impact of each coefficient in the evaluation of the global affinity value. In our experimentation, we considered both types of equally relevant affinity and we set $w_{NA} = w_{SA} = 0.5$.
In the Global Affinity evaluation, if the terminological affinity coefficient of two classes is 0, this means that the classes do not describe similar concepts and their structural affinity is not further evaluated. Consequently, also the $GA()$ coefficient is equal to zero.

Example 5 Let us return to the Name and Structural Affinity coefficients of Example 3, and Example 4, respectively. The Global Affinity coefficient of `University_Student` and `School_Member` is computed as follows:

$$GA(\texttt{University\_Student}, \texttt{School\_Member}) = 0.5 \cdot 0.64 + 0.5 \cdot 0.75 = 0.695$$

5 Clustering ODL$_{I^3}$ classes

When analyzing N schemas, a high number of classes need to be compared for affinity evaluation. To identify all the classes having affinity in the considered schemas, we employ a hierarchical clustering technique, which classifies classes into groups at different levels of affinity, forming a tree [24].

Hierarchical clustering procedure (see Fig. 6) exploits a matrix M of rank K being k the total number of ODL$_{I^3}$ classes to be analyzed. An entry $M[h, k]$ of the matrix represents the affinity coefficient $GA()$ between classes c_h and c_k. Clustering is iterative and starts by placing each class in a cluster by itself. Then, at each iteration, the two clusters having the greatest $GA()$ value in M are merged. M is updated at each merging operation by deleting the rows and the columns corresponding to the merged clusters, and by inserting a new row and a new column for the newly defined cluster. The $GA()$ value between the newly defined cluster and each remaining cluster is computed, by keeping the maximum $GA()$ value among the values of every merged clusters and each remaining cluster in the matrix. The procedure terminates when only one cluster is left and produces as the output an affinity tree.

Fig. 7 shows the affinity tree resulting from applying the clustering procedure to our set of classes.

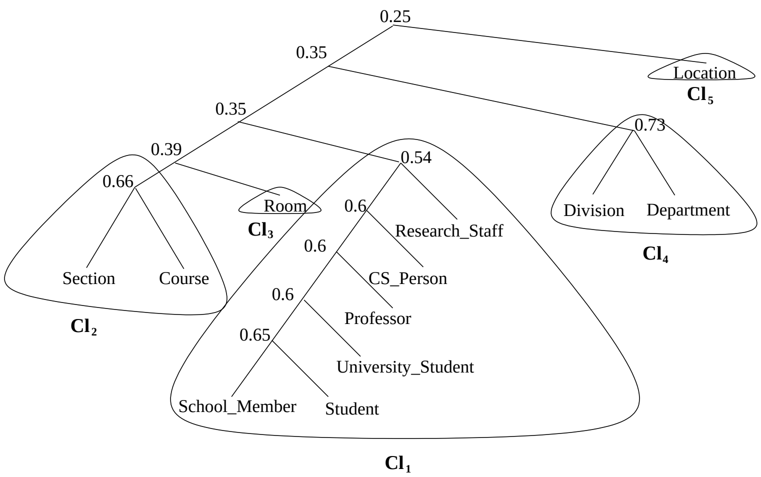

Figure 7: Affinity tree of S_1, S_2, and S_3

6 Generation of the mediator global schema

In this section we present the process which leads to the definition of the mediator global schema, that is the mediator view of data stored in local sources. Starting from the affinity tree produced with clustering, we define, for each cluster in the tree, a global class $global\_class_i$ representative of the classes contained in the cluster (i.e., a class providing the unified view of all the classes of the cluster).

The generation of the $global\_class_i$ is interactive with the designer. Let Cl_i be a cluster in the affinity tree. First, the Global Schema Builder component of MOMIS associates with the $global\_class_i$ a set of global attributes, corresponding to the union of the attributes of the classes belonging to Cl_i, where the attributes with affinity are unified into a unique global attribute in $global\_class_i$. The attribute unification process is performed automatically for what concerns the names of attributes with affinity, according to the following rules:

- for attributes that have name affinity due to SYN relationships only, only one term is selected and assigned to the corresponding global attribute in $global\_class_i$;
- for attributes that have name affinity due to BT and NT relationships, a name which is a broader term for all of them is selected and assigned to the corresponding global attribute in $global\_class_i$.

For example, the output of this attribute unification process for the Cl_1 of Fig. 7, is the following set of global attributes:

```
Cl1 = ( name, rank, title, dept_code, year, takes, relation,
        email, student_code, tax_fee, section_code, faculty )
```

Additional information has to be provided for completing the global class definition. In particular, MOMIS asks the designer to interactively supply information regarding :

1. the $global\_class_i$ name; the tool proposes a set of candidate names for the global class, by exploiting the terminological relationships defined in the Common Thesaurus for the classes in Cl_i. Based on these names, the designer can decide the most suitable name for $global\_class_i$. In our example, the designer decides to assign the name `University_Person` to the global class defined from Cl_1.

2. *mappings* between the global attributes of $global\_class_i$ and the corresponding attributes of the classes of Cl_i. In particular, if a global attribute is obtained from more than one attribute of the same class in the Cl_i, the designer is asked to specify the type of *correspondence* to be set for the global attribute. In MOMIS, we provide the following options:
   - *and* correspondence: this specifies that a global attribute corresponds to the union of the attributes of a given class $c_h \in Cl_i$. For example, the global attribute `name` of `University_Person` corresponds to both `first_name` and `last_name` attributes of `School_Member` in Cl_1. By specifying the *and* correspondence for `name`, the designer states that the values of both `first_name` and `last_name` attributes have to be considered when `School_Member` is involved.
   - *or* correspondence: this specifies that a global attribute corresponds to at most one attribute of a given class $c_h \in Cl_i$;
3. default values to be assigned to global attributes in correspondence of a given class $c_h \in Cl_i$. A default value is always verified for the global attribute when evaluated on the class c_h. For example, the designer knows that the global attribute `rank` of of `University_Person` has value 'Professor' when evaluated on `Research_Staff`, even if this information is not stored in any attribute of `Research_Staff`. This can be specified by setting a default value for `rank` with respect to `Research_Staff`.
4. new attributes for the global class.

A global class $global\_class_i$ is specified in ODL_{I^3} and the information on attribute mappings and default values is represented in form of *rules*, following the syntax of ODL_{I^3} (see Appendix A). An example of ODL_{I^3} specification for the global class `University_Person` is shown in Fig. 6. As we can see from this figure, for each attribute, in addition to its declaration, mapping rules are defined, specifying both information on how to map the attribute on the corresponding attributes of the associated cluster and on possible default/null values defined for it on cluster classes. For example, for the global attribute `name`, the mapping rule specifies the attributes that have to be considered in each class of the cluster Cl_1. In this case, an *and* correspondence is defined for `name` for the class `University.Research_Staff` (we use dot notation for specifying the source of a given class belonging to the cluster). A mapping rule is defined for the global attribute `rank` to specify the value to be associated with `rank` for the instances of `University.Research_Staff` and `University.School_Member`.

In MOMIS, a mapping table is maintained for each defined global class, storing the information on its mapping rules. As an example, mapping tables for the global classes extracted from clusters Cl_1 and Cl_4 of Fig. 7 are shown in Fig. 9.

The global schema of the mediator is composed of the global classes defined for all the clusters of the affinity tree. Once all global classes have been defined following this process, they are revised to check their mutual consistency. In particular, their attributes expressing relationships are checked to define relationships among global classes. For example, if an attribute with a correspondence exist in all classes belonging to the cluster associated with this global class and if a global class has been defined for classes referenced by such attribute, then a relationship is defined between these two global classes.

```
interface University_Person
(extent Research_Staffers, School_Members, CS_Person
        Professors, Students, University_Students
 key    name
{ attribute string name
     mapping_rule  (University.Research_Staff.first_name and
                   University.Research_Staff.last_name)
                   (University.School_Member.first_name and
                   University.School_Member.last_name),
                   Computer_Science.CS_Person.name
                   Computer_Science.Professor.name
                   Computer_Science.Student.name
                   Tax_Position.University_Student.name;
  attribute string rank
     mapping_rule  University.Research_Staff = 'Professor',
                   University.School_Member = 'Student',
     ... }
```

Figure 8: Example of global class specification in ODL_{I^3}

7 Query Reformulation

In this section, we describe how it is possible to exploit the information on the global schema for global processing. When the user submits a query on the global schema, MOMIS takes the query and produces a set of subqueries that will be sent to the single information sources. According to other semantic approaches [1, 2], this process consists of two main phases:

- semantic optimization;
- query plan formulation.

7.1 Semantic Optimization

In this phase, MOMIS mediator operates on the query by exploiting the semantic optimization techniques [36] supported by ODB-Tools [4, 5], in order to reduce the query access plan cost. The query is replaced by a new one that incorporates any possible restriction which is not present in the original query but is logically implied by the query on the global schema. The transformation is based on logical inference from content knowledge (in particular on the integrity constraints rules) of the mediator global schema, shared among the classes belonging to the cluster.
Let us consider the request: "Retrieve the professors whose workplace has a research budget greater then $ 60.000", corresponding to the following query:

```
select name
from University_Person
where rank = 'Professor'
and works.budget > 60000
```

Let us suppose that in our `University` domain exists a relationship between the research fund and the application area, stating that all the workplaces with a budget greater than $

University_Person	name	rank	works	faculty	...
Research_Staff	first_name *and* last_name	'Professor'	dept_code	null	...
School_Member	first_name *and* last_name	'Student'	null	faculty	...
CS_Person	name	null	null	'Computer_Science'	...
Professor	name	rank	belongs_to	'Computer_Science'	...
Student	name	rank	null	'Computer_Science'	...
University_Student	name	'Student'	null	faculty_name	...

Workplace	name	area	employee_nr	budget	...
Department	dept_name	dept_area	null	budget	...
Division	description	sector	employee_nr	fund	...

Figure 9: University_Person and Workplace mapping tables

60.000 belongs to the Scientific area. Thus in the ODL_{I^3}, we can define the following constraint rule on the global schema:

```
rule R1 forall X in Workplace:  (X.budget > 60000) then X.area = 'Scientific';
```

The mediator executes the semantic expansion of the input query by applying rule `R1`. The resulting query is the following:

```
select name
from University_Person
where rank = 'Professor'
and works.budget > 60000
and works.area = 'Scientific'
```

Semantic expansion is performed in order to add predicates in the "where clause": this process makes query plan formulation more expensive (because a heavier query has to be translated for each interesting source) but single sources' query processing overhead can be lighter in case secondary indexes on added predicates exist in the involved sources. The query optimization algorithm included in ODB-Tools and presented in [6] is polynomial. Our experiments shows that the overhead of optimization is very small compared to the overall query processing cost. On a set of 8 queries performed on 10 different database instances, the query optimization gave an average reduction in execution time of 47%.

7.2 Query Plan Formulation

Once the mediator has produced the optimized query, a set of subqueries for the local information sources will be formulated. For each information source, by using the mapping table associated with each global class, the mediator has to express the optimized query in terms of local schemas. In order to obtain each local query, the mediator checks and translates every predicate (*boolean factor*) in the where clause. In particular, a *Check* algorithm (shown in Fig. 10) is used to determine the local queries to be generated starting for the clustered associated with the global class queried by the user.

```
Check algorithm
/* Input: Cluster Cl_j with its k classes */

begin procedure:
        for each class c_h ∈ Cl_j do
            for each boolean factor do
                if ( global attribute ∈ boolean factor) corresponds to:
                1. null value in c_h: no local query is generated on c_h;
                2. default value in c_h: if the value specified in the boolean factor differs from the default one
                                         then no local query is generated on c_h;
end procedure.
```

Figure 10: Check algorithm

Referring to our example, the step 1 of the *Check* algorithm will exclude the `School_Member`, `CS_Person`, `Student` and `University_Student` elements and the step 2 the `School_Member` and `University_Student`, so that we derive the following queries for sources S_1 and S_2, respectively:

```
S1: select first_name, last_name
    from Research_Staff as R, Department as D
    where R.dept_code = D.dept_code
    and D.budget > 60000
    and D.dept_area = 'Scientific'
```

```
S2: select name
    from Professor as P
    where P.rank = 'Professor'
    and P.belongs_to.fund > 60000
    and P.belongs_to.sector = 'Scientific'
```

If integrity constraints are provided also for a local source, it is possible to perform semantic query optimization with ODB-Tools on the corresponding local query too, before sending the query to the source. For example, if the following rule is provided for the `Computer_Science` (S_1) source:

```
rule R2 forall X in Division:  (X.fund > 60000) then X.employee_nr >20;
```

`R2` can be used the rule to optimize the local query `S2`, giving raise to the following query:

```
S2': select name
     from Professor as P
     where P.rank = 'Professor'
     and P.belongs_to.fund > 60000
     and P.belongs_to.sector = 'Scientific'
     and P.belongs_to.employee_nr > 20
```

This transformation could be useful if an index is available for the `employee_nr` attribute.

7.3 Query Optimization using extensional knowledge

Another kind of knowledge which could be useful for query optimization is inter-schema knowledge of extensional type. It is defined in terms of *inter-schema properties*, specifying mutual relationships between class instances of corresponding databases.

Let $EXT(c)$ be the set of instances (extension) of class $c \in Cl_i$. Example of inter-schema properties that can be specified for two classes c and c' at the extensional level are the following:

1. Equivalence. Two classes c and c' are equivalent at the extensional level, denoted by $c \equiv_{ext} c'$ if and only if $EXT(c) \equiv EXT(c')$;
2. Containment. Two classes c and c' are contained one another at the extensional level, denoted by $c \subset_{ext} c'$, if and only if $EXT(c) \subset EXT(c')$;

3. Overlapping. Two classes c and c' are overlapping at the extensional level, denoted by $c \cap_{ext} c'$, if and only if $EXT(c) \cap EXT(c') \neq \emptyset$ and $c \subset_{ext} c'$ is not verified;
4. Disjointness. Two classes c and c' are disjoint at the extensional level, denoted by $c\ \emptyset_{ext}\ c'$, if and only if $EXT(c) \cap EXT(c') = \emptyset$.

Referring to our example, let us suppose that the designer is able to state this extensional property: `School_Member` $\subset_{ext}$ `University_Student` for classes in cluster Cl_1.
The mediator may exploit this knowledge during the query processing phase to minimize the data accesses required. In particular, if the user is looking for the name of the students attending 'Economics' faculty, the following query will be submitted:

```
select name
from University_Person
where rank = 'Student'
and faculty = 'Economics'
```

In this case, the query reformulation step determines two local queries to be sent to $\mathtt{S}_1$ and $\mathtt{S}_3$ sources related to `School_Member` and `University_Student` classes respectively. By exploiting the introduced extensional property the mediator knows that all student instances of the first class (`School_Member`) are also stored (and thus retrieved) in the second one (`University_Student`). Consequently, only the following query is considered and will be sent to the $\mathtt{S}_3$ source:

```
select name
from University_Student
where faculty_name = 'Economics'
```

while no query is sent to the source $\mathtt{S}_1$.

8 Related work

Much efforts have been spent on the problem of accessing information distributed over multiple sources both in the AI-oriented database community and in the more traditional database community. Some of the most important works to integrate distributed information sources are the TSIMMIS project [21, 27], the GARLIC project [19, 33], and the SIMS project [1, 2]. The TSIMMIS project uses pattern matching techniques to match user queries against predefined queries with stored query plan, and a self-describing model (OEM [32]) to represent the objects retrieved from the local sources. Both SIMS and GARLIC attempt to define a global schema, trying to support every possible user query instead of a predefined subset of them. In particular, the GARLIC approach builds up on a complex wrapper architecture to describe the local sources with an OO language (GDL, an extension of ODMG-SQL that adds support for path expressions, nested collections and methods), and on the definition of Garlic Complex Objects to manually unify the local sources classes. Eventually, the SIMS project explores the use of Description Logics for describing information sources. This approach requires constructing a general domain model that encompasses the relevant parts of the database schemata, and relating each of the local database model to this general domain model. Then, for each query submitted to the system, an optimized query plan is computed using AI planning techniques (based on the LOOM knowledge representation language).

Other researches have concentrated more specifically issues related to the identification and resolution of the semantic inconsistencies between schemas by exploiting the definition of correspondence between schema elements. Some approaches rely on terminological analysis of schema elements. In [13], a Summary Schema Model (SSM) has been proposed to provide support to identify semantically similar entities. According to an existing taxonomy, the SSM is used as an abstract and concise description of all data available in the sources. Two entities are semantically related if they are mapped to the same concept of the taxonomy. In [25], the main purpose consists in defining techniques based on fuzzy logic to derive terminological relationships between schema elements.

Other approaches consider also attributes and domains of schema elements [11, 22]. In [11], relations, attributes, and integrity constraints in source schema are properly expressed using logics-based to unified knowledge of all the elements in the sources. In [22] two systems are illustrated developed to provide support to designer in establishing attribute correspondences problem. The first tool, Data Element Tool-Based Analysis (DELTA) finds candidate attribute correspondences by exploiting textual similarities between data element definitions. The second tool, SemInt uses schema information (e.g, data types, length, keys) along with statistics (e.g., average, maximum, variance) describing the data element values. Moreover, the results of applying these tools to some databases of the Air Force environment are discussed. Finally, other approaches have been proposed for resolution of semantic heterogeneity. These approaches consider federated architecture, where notion of federated schema is given [34, 38].

9 Conclusions and future work

In this paper, we have presented an intelligent approach to schema integration for heterogeneous information sources. It is a semantic approach based on a DescriptionLogics component (ODB-Tools engine) and a cluster generator module together with a minimal ODL_{I^3} interface module. In this way, generation of the global schema for the mediator is a semi-automated process. ODL_{I^3} language is available and basic functionalities related to Thesaurus construction and clustering have been implemented. We are developing on unification functionalities in the framework of the architecture presented in Section 2. Future research work will be devoted to a deeper study of query processing and optimization processes, and to the development of corresponding Query Manager functionalities. In particular, inter-schema extensional properties relevant to perform the information integration process in view of query processing will be investigated. Moreover, the impact of this topic on clustering will be studied in order to enhance the classification process with semantic features.

References

[1] Y. Arens, C.Y. Chee, C. Hsu, and C. A. Knoblock, "Retrieving and Integrating Data from Multiple Information Sources," in *International Journal of Intelligent and Cooperative Information Systems. Vol. 2, No. 2. Pp. 127-158*, 1993.

[2] Y. Arens, C. A. Knoblock and C. Hsu, "Query Processing in the SIMS Information Mediator," in *Advanced Planning Technology, editor, Austin Tate, AAAI Press, Menlo Park, CA*, 1996.

[3] Y.J. Breibart et al. "Database Integration in a Distributed Heterogeneous Database System," in *Proc. 2nd Intl IEEE Conf.on Data Engineering, Los Angeles, CA*, 1986.

[4] D. Beneventano, S. Bergamaschi, C. Sartori, M. Vincini, "ODB-Tools: a Description Logics Based Tool for Schema Validation and Semantic Query Optimization in Object Oriented Databases," in *Proc. of Int. Conf. on Data Engineering, ICDE'97*, Birmingham, UK, April 1997.

[5] D. Beneventano, S. Bergamaschi, C. Sartori, M. Vincini, "ODB-QOptimizer: a Tool for Semantic Query Optimization in OODB", in *Proc. of Fifth Conference of the Italian Association for Artificial Intelligence (AI*IA97)*, Rome 1997.

[6] D. Beneventano, S. Bergamaschi and C. Sartori "Semantic Query Optimization by Subsumption in OODB," in *Proc. of the Int. Workshop on Flexible Query-Answering Systems (FQAS96)*, Roskilde, Denmark, may 1996, pp. 167-185

[7] S. Bergamaschi "Extraction of Informations from highly Heterogeneous Sources of Textual Data," in *Cooperative Information Agents, First International Workshop, CIA' 97 Proceedings. Lecture Notes in Computer Science*, Kiel, Germany, February 26-28, 1997.

[8] S. Bergamaschi and B. Nebel, "Acquisition and Validation of Complex Object Database Schemata Supporting Multiple Inheritance," *Applied Intelligence: The International Journal of Artificial Intelligence, Neural Networks and Complex Problem Solving Technologies*, 4:185–203, 1994.

[9] S. Bergamaschi and C. Sartori, "On Taxonomic Reasoning in Conceptual Design," in *ACM Transactions on database Systems*, 17(3):385–422, September 1992.

[10] S. Bergamaschi, C. Sartori, "An Approach for the Extraction of Information from Heterogeneous Sources of Textual Data," in *Proceedings of the 4th KRDB Workshop*, Athens, Greece, August 1997.

[11] J.M. Blanco, A. Illarramendi, A. Goni, "Building a Federated Relational Database System: An Approach Using a Knowledge-Based System," *Int. Journal of Intelligent and Cooperative Information Systems*, Vol.3, No.4, 1994, pp.415-455.

[12] A. Borgida, R.J. Brachman, D.L. McGuinness and L.A. Resnik, "CLASSIC: A Structural Data Model for Objects," in *SIGMOD*, pages 58-67, Portland, Oregon, 1989.

[13] M.W. Bright, A.R. Hurson, S. Pakzad, "Automated Resolution of Semantic Heterogeneity in Multidatabases," *ACM Transactions on Database Systems,* Vol.19, No.2, June 1994, pp.212-253.

[14] P. Buneman, S. Davidson, M. Fernandez and D. Suciu, " Adding a Structure to Unstructured Data," in *Proc. of Int. Conf. on Database Theory*, Delphi, Greece, January 1997, pp. 336-350.

[15] P. Buneman, L. Raschid, J. Ullman, "Mediator Languages - a Proposal for a Standard," Report of an I^3/POB working group held at the University of Maryland, April 1996. *ftp://ftp.umiacs.umd.edu/pub/ONRrept/medmodel96.ps.*

[16] S. Castano, V. De Antonellis, "Semantic Dictionary Design for Database Interoperability," in *Proc. of Int. Conf. on Data Engineering, ICDE'97*, Birmingham, UK, April 1997.

[17] S. Castano, V. De Antonellis, M.G. Fugini, B. Pernici, "Conceptual Schema Analysis: Techniques and Applications," accepted for publication on *ACM Transactions on Database Systems*, 1997.

[18] D. Calvanese, G. De Giacomo and M. Lenzerini, "Structured Objects: Modeling and Reasoning", *in Proc. of Int. Conference on Deductive and Object-Oriented Databases*, 1995.

[19] M.J.Carey, L.M. Haas, P.M. Schwarz, M. Arya, W.F. Cody, R. Fagin, M. Flickner, A.W. Luniewski, W. Niblack, D. Petkovic, J. Thomas, J.H. Williams and E.L. Wimmers, "Towards Multimedia Information System: The Garlic Approach", *IBM Almaden Research Center*, San Jose, CA 95120.

[20] R. Cattel (ed.), *The Object Database Standard: ODMG-93*, Morgan Kaufmann, 1996.

[21] S. Chawathe, H. Garcia Molina, J. Hammer, K. Ireland, Y. Papakostantinou, J.Ullman, and J.Widom, "The TSIMMIS project: Integration of Heterogeneous Information Sources," *in IPSJ Conference, Tokyo, Japan*, 1994. ftp://db.stanford.edu/pub/chawathe/1994/tsimmis-overview.ps.

[22] C. Clifton, E. Housman, A. Rosenthal, "Experience with a Combined Approach to Attribute-Matching Across Heterogeneous Databases," in *IFIP DS-7 Data Semantics Conf.*, Switzerland, 1997

[23] U. Dayal and H. Hwuang, "View Definition and Generalization for Database Integration in a Multidatabase System," In *Proc. IEEE Workshop on Object-Oriented DBMS - Asilomar, CA*, 1986.

[24] B. Everitt, *Cluster Analysis*, Heinemann Educational Books Ltd, Social Science Research Council, 1974.

[25] P. Fankhauser, M. Kracker, E.J. Neuhold, "Semantic vs. Structural Resemblance of Classes," *SIGMOD RECORD*, Vol.20, No.4, December 1991, pp.59-63.

[26] *Associative Networks*, N.V. Findler Ed., Academic Press, 1979.

[27] H. Garcia-Molina et al., "The TSIMMIS Approach to Mediation: Data Models and Languages," in *NGITS workshop*, 1995. *ftp://db.stanford.edu/pub/garcia/1995/tsimmis-models-languages.ps.*

[28] J. Hammer and D. McLeod, "An Approach to Resolving Semantic Heterogeneity in a Federation of Autonomous, Heterogeneous Database Systems," *Intl Journal of Intelligent and Cooperative Information Systems*, 2:51–83, 1993.

[29] W. Kim, I. Choi, S. Gala, M. Scheevel, "On Resolving Schematic Heterogeneity in Multidatabase Systems," *Distributed and Parallel Databases,* Vol.1, No.3, 1993, and in *Modern Database Systems-The Object Model, Interoperability and Beyond*, W. Kim (Editor), ACM Press, 1995.

[30] A.Y. Levy, A. Rajaraman, J.J. Ordille, "Querying Heterogeneous Information Sources Using Source Descriptions," in Proc. of *22th VLDB Conference*, Mumbai (Bomaby), 1996.

[31] S.E. Madnick, "From VLDB to VMLDB (Very MANY Large Data Bases): Dealing with Large-Scale Semantic Heterogeneity," in *Proc. of the 21th Int. Conf. on Very Large Databases*, Zurich, Switzerland, September 1995, pp.11-16.

[32] Y. Papakonstantinou, H. Garcia-Molina, J. Widom, "Object Exchange Across Heterogrnrous Information Sources", in *Data Engineering Conf.*, March, 1995.

[33] M.T. Roth, P. Scharz, "Don't Scrap It, Wrap it! A Wrapper Architecture for Legacy Data Sources", in *Proc. of the 23rd Int. Conf. on Very Large Databases*, Athens, Grrece, 1997.

[34] M.P. Reddy, B.E. Prasad, P.G. Reddy, A. Gupta, "A Methodology for Integration of Heterogeneous Databases," *IEEE Trans. on Knowledge and Data Engineering,* Vol.6, No.6, December 1994, pp.920-933.

[35] F. Saltor, E. Rodriguez, "On Intelligent Access to Heterogeneous Information," in *Proceedings of the 4th KRDB Workshop*, Athens, Greece, August 1997.

[36] S. Shenoy and M. Ozsoyoglu, "Design and Implementation of a Semantic Query Optimizer," *IEEE trans. Knowl. and Data Engineering*, 1(3):344-361, September 1989.

[37] S. Shenoy et al. "The Rufus System: Information Organization for Semistructured Data," in *Proc. VLDB Conference*, Dublin, Ireland, 1993

[38] A.P. Sheth and J.P. Larson, "Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases," *ACM Computing Surveys,* Vol. 22, No. 3, September 1990.

[39] M. Siegel and E. Sciore and S. Salveter, "A Method for Automatic Rule Derivation to Support Semantic Query Optimization," in *ACM Transactions on Database Systems*, Vo. 17, Num. 4, pag. 563-600, dec. 1992.

[40] AA. VV. "The Common Object Request Broker: Architecture and Specification," Technical report, Object Request Broker Task Force, 1993. Revision 1.2, Draft 29, December.

[41] G. Wiederhold, "Mediators in the architecture of Future Information Systems," *IEEE Computer*, Vol. 25, 1992, pp.38-49.

[42] G. Wiederhold et al., "Integrating Artificial Intelligence and Database Technologies," *Journal of Intelligent Information Systems, Special Issue: Intelligent Integration of Information*, Vol. 6, Number 2/3, June 1996.

[43] W.A. Woods and J.G. Schmolze, "The kl-one family," in *F.W. Lehman, editor*, published as a Special issue of *Computers & Mathematics with Applications*, Volume 23, Number 2-9.

A The ODL$_{I^3}$ description language

The following is a BNF description for the ODL$_{I^3}$ description language.
We included the syntax fragments which differ from the original ODL grammar, referring to this one for the remainder.

⟨interface_dcl⟩	::=	⟨interface_header⟩ {[⟨interface_body⟩]};
⟨interface_header⟩	::=	**interface** ⟨identifier⟩
		[⟨inheritance_spec⟩]
		[⟨type_property_list⟩]
⟨inheritance_spec⟩	::=	: ⟨scoped_name⟩ [,⟨inheritance_spec⟩]
⟨type_property_list⟩	::=	**(** [⟨source_spec⟩] [⟨extent_spec⟩] [⟨key_spec⟩] [⟨f_key_spec⟩] **)**
⟨source_spec⟩	::=	**source** ⟨source_type⟩ ⟨source_name⟩
⟨source_type⟩	::=	**relational** | **nfrelational** | **object** | **file**
⟨source_name⟩	::=	⟨identifier⟩
⟨extent_spec⟩	::=	**extent** ⟨string⟩
⟨key_spec⟩	::=	**key**[**s**] ⟨key_list⟩
⟨f_key_spec⟩	::=	**foreign_key** ⟨f_key_list⟩
. . .		
⟨attr_dcl⟩	::=	**[readonly] attribute**
		⟨domain_type⟩ ⟨attribute_name⟩
		[⟨fixed_array_size⟩] [⟨mapping_rule_dcl⟩]
⟨mapping_rule_dcl⟩	::=	**mapping_rule** ⟨rule_list⟩
⟨rule_list⟩	::=	⟨rule⟩ | ⟨rule⟩**,**⟨rule_list⟩
⟨rule⟩	::=	⟨local_attr_name⟩ | **‘**⟨identifier⟩**’**
		⟨and_expression⟩ | ⟨or_expression⟩
⟨and_expression⟩	::=	**(** ⟨local_attr_name⟩ **and** ⟨and_list⟩ **)**
⟨and_list⟩	::=	⟨local_attr_name⟩ | ⟨local_attr_name⟩ **and** ⟨and_list⟩
⟨or_expression⟩	::=	**(** ⟨local_attr_name⟩ **or** ⟨or_list⟩ **)**
⟨or_list⟩	::=	⟨local_attr_name⟩ | ⟨local_attr_name⟩ **or** ⟨or_list⟩
⟨local_attr_name⟩	::=	⟨source_name⟩**.**⟨class_name⟩**.**⟨attribute_name⟩
. . .		
⟨relationships_list⟩	::=	⟨relationship_dcl⟩**;** | ⟨relationship_dcl⟩**;** ⟨relationships_list⟩
⟨relationships_dcl⟩	::=	⟨local_attr_name⟩ ⟨relationship_type⟩ ⟨local_attr_name⟩
⟨relationship_type⟩	::=	**SYN** | **BT** | **NT** | **RT**
. . .		
⟨rule_list⟩	::=	⟨rule_dcl⟩**;** | ⟨rule_dcl⟩**;** ⟨rule_list⟩
⟨rule_dcl⟩	::=	**rule** ⟨identifier⟩ ⟨rule_pre⟩ **then** ⟨rule_post⟩
⟨rule_pre⟩	::=	⟨forall⟩ ⟨identifier⟩ **in** ⟨identifier⟩ **:** ⟨rule_body_list⟩
⟨rule_post⟩	::=	⟨rule_body_list⟩

⟨rule_body_list⟩	::=	**(** ⟨rule_body_list⟩ **)** | ⟨rule_body⟩ |
		⟨rule_body_list⟩ **and** ⟨rule_body⟩ |
		⟨rule_body_list⟩ **and** **(** ⟨rule_body_list⟩ **)**
⟨rule_body⟩	::=	⟨dotted_name⟩ ⟨rule_const_op⟩ ⟨literal_value⟩ |
		⟨dotted_name⟩ ⟨rule_const_op⟩ ⟨rule_cast⟩ ⟨literal_value⟩ |
		⟨dotted_name⟩ **in** ⟨dotted_name⟩ |
		⟨forall⟩ ⟨identifier⟩ **in** ⟨dotted_name⟩ **:** ⟨rule_body_list⟩ |
		exists ⟨identifier⟩ **in** ⟨dotted_name⟩ **:** ⟨rule_body_list⟩
⟨rule_const_op⟩	::=	$= \mid \geq \mid \leq \mid > \mid <$
⟨rule_cast⟩	::=	**(**⟨simple_type_spec⟩**)**
⟨dotted_name⟩	::=	⟨identifier⟩ | ⟨identifier⟩**.**⟨dotted_name⟩
⟨forall⟩	::=	**for all** | **forall**

B ODL$_{I^3}$ sources descriptions

`UNIVERSITY source:`

```
interface Research_Staff
( source relational University
  extent Research_Staffers
  keys first_name, last_name
  foreign_key dept_code, section_code )
{ attribute string first_name;
  attribute string last_name;
  attribute string relation;
  attribute string e_mail;
  attribute integer dept_code;
  attribute integer section_code; };

interface School_Member
( source relational University
  extent School_Members
  keys first_name, last_name )
{ attribute string  first_name;
  attribute string  last_name;
  attribute string  faculty;
  attribute integer year; }

interface Department
( source relational University
  extent Departments
  key dept_code )
{ attribute string dept_name;
  attribute integer dept_code;
  attribute integer budget;
  attribute string dept_area; };

interface Section
( source relational University
  extent Sections
  key section_code
  foreign_key room_code )
{ attribute string section_name;
  attribute integer section_code;
  attribute integer length;
  attribute integer room_code; };

interface Room
( source relational University
  extent Room
```

```
  key room_code )
{ attribute integer room_code;
  attribute integer seats_number;
  attribute string notes; };
```

COMPUTER_SCIENCE source:

```
interface CS_Person
( source object Computer_Science
  extent CS_Persons
  key name )
{ attribute string name; };
```

```
interface Professor : CS_Person
( source object Computer_Science
  extent Professors )
{ attribute string title;
  attribute Division belongs_to;
  attribute string rank; };
```

```
interface Student : CS_Person
( source object Computer_Science
  extent Students )
{ attribute integer year;
  attribute set<Course> takes;
  attribute string rank; };
```

```
interface Division
( source object Computer_Science
  extent Divisions
  key description )
{ attribute string description;
  attribute Location address;
  attribute integer fund;
  attribute integer employee_nr;
  attribute string sector; };
```

```
interface Location
( source object Computer_Science
  extent Locations
  keys city, street, county, number)
{ attribute string city;
  attribute string street;
  attribute string county;
  attribute integer number; };
```

```
interface Course
( source object Computer_Science
  extent Courses
  key course_name )
{ attribute string course_name;
  attribute Professor taught_by; };
```

Tax_Position source:

```
interface University_Student
( source file Tax_Position
  extent University_Students
  key student_code )
{ attribute string name;
  attribute integer student_code;
```

```
attribute string faculty_name;
attribute integer tax_fee; };
```