An Intelligent Approach to Information Integration*

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

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 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. *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) while *clustering techniques* are employed to build the *global schema*, i.e. the *unified* view of the data to be used for query processing.

keywords: intelligent information integration, application ontology, semantic integration, mediator, heterogeneous databases, ODMG-93 standard.

1 Introduction

In the continuing quest to provide integrated access to distributed information, the problem of integrating information from heterogeneous sources is becoming more and more relevant. Some approaches have been recently proposed in the literature for the extraction and integration of structured databases [5, 6, 24] and semi-structured data [18, 22]. Data integration architectures are usually based on mediators, where knowledge about data of multiple sources is combined and stored to provide global views of the underlying data for query processing [14, 23]. While these systems follow a "structural" approach, our proposal follows a "semantic" approach to integration [4], based on these assumptions:

- for each source, its conceptual schema, i.e., meta-data, is available;
- semantic information is encoded in the schema;
- a common data model for describing the information to be shared is available;
- a partial or total unification of the source schemas is performed.

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

Many 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 of the information. Semantic heterogeneity gives origin to different kinds of conflicts, such as naming conflicts, when different names are employed to represent the same information, or structural conflicts, when different modeling constructs are used in different sources to represent the same piece of information [17, 19, 21]. When name conflicts are involved, problems of ontological nature have to be faced, to come up with a unified terminology to describe the information spread in different sources and describing the same real-world concepts. Furthermore, a different and more difficult kind of problem may arise when the same name is used, across several sources, to identify different real world concepts.
In this paper, we propose an intelligent approach to the integration of heterogeneous information, coupling functionalities to solve application ontology problems [16] (taking into account semantic heterogeneity) and system discrepancies problems.
The approach follows the "semantic" paradigm, in that conceptual schemas of each involved source are considered, and a common data model (ODM_{I^3}) and a common data language (ODL_{I^3} language) are adopted to describe sharable information. ODM_{I^3} and ODL_{I^3} are a subset of the corresponding ODMG-93 [11] ODM and ODL, respectively. A Description Logics `ocdl` (object description language with constraints [1]) is used as a kernel language and ODB-Tools as the supporting system.

The approach consists of an *extraction and analysis* phase and of a *unification* phase. In the extraction and analysis phase, a Common Thesaurus of *terminological relationships* is derived from source schemas, which constitutes the basis for identifying semantically similar classes in different source schemas using clustering techniques. In the unification phase, clusters of semantically similar classes are unified to build an integrated global schema for the analyzed sources. The goal of this process is to overcome the absence of a common shared ontology, and to semi-automatically derive a global schema, including all the concepts that belong to the source schemata to be integrated. The use of the `ocdl` Description Logics language together with hierarchical clustering techniques are the original contributions of the approach, which allow a semi-automated integration process. Description Logics allows us to reason about the validity of terminological relationships in the Thesaurus and to infer new relationships out of a set of explicit ones in an automated way. Clustering techniques allow the automated identification of schema classes in different source schemas that are candidate to be unified 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 (the syntax is in Appendix A) and we outline the approach to intelligent 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 source schemas. In Section 5 we illustrate the cluster generation process for the identification of semantically similar classes and, in Section 6, we describe the unification process for building the global schema. Finally, in Section 7 we give our concluding remarks.

2 Basic concepts of the approach

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

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

In Figure 1 the architecture of the supporting system is shown. With respect to the literature, this can be considered as an example of powerful I^3 system [7]. In the architecture, 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 translation is made by the *wrapper* for the query, from the OQL$_{I^3}$ language to a local request executed by the single sources. Above the *wrapper* there is a I^3 *mediator*, a software module that combines, integrates, and refines data received from the *wrappers*. In addition, the mediator generates the OQL$_{I^3}$ queries for the *wrappers*, starting from the query request formulated on the global schema. Using the Description Logics techniques we can generate in an automatic way the translation into local queries for a given user query (see Section 6 for an example). The mediator module is obtained by coupling a "semantic approach", based on a Description Logics component (i.e. ODB-Tools engine) and on a clustering component (i.e., ARTEMIS tool [9]), together with a minimal ODL$_{I^3}$ interface.
Let $S = \{S_1, S_2, \ldots, S_N\}$ be a set of schemas of N heterogeneous sources that have to be integrated. In order to easily communicate source descriptions between wrappers and mediator engine, we defined a description language, called ODL$_{I^3}$(for the BNF syntax of ODL$_{I^3}$ language, see appendix A), derived from ODL [11] and from the I^3 mediator language proposal, as it is described in [7]. According to recommendations of [7], and to the diffusion of the object data model (and its standard ODMG-93), ODL$_{I^3}$ is very close to 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 database, object-oriented database, file, ...). It will be the wrapper task to translate the original description language of any particular source into ODL$_{I^3}$ and to add the information needed by the mediator, such as the source name and type.

According to ODL$_{I^3}$, each source schema S_i is a collection of *classes*, $S_i = \{c_{1i}, c_{2i}, \ldots, c_{mi}\}$. 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.

To obtain an intelligent schema integration, we use ODB-Tools [1, 2], a system based on Description Logics which performs schema validation and query optimization. In the following, we concentrate on mediator functionalities of the proposed architecture by discussing the approach to intelligent schema integration in more detail.

2.2 Overview of the approach

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

1. *Semi-automatic generation of a Common Thesaurus.*
   The objective of this step is the construction of a Common Thesaurus of *terminological relationships* for schema classes in different source ODL$_{I^3}$ schemas.

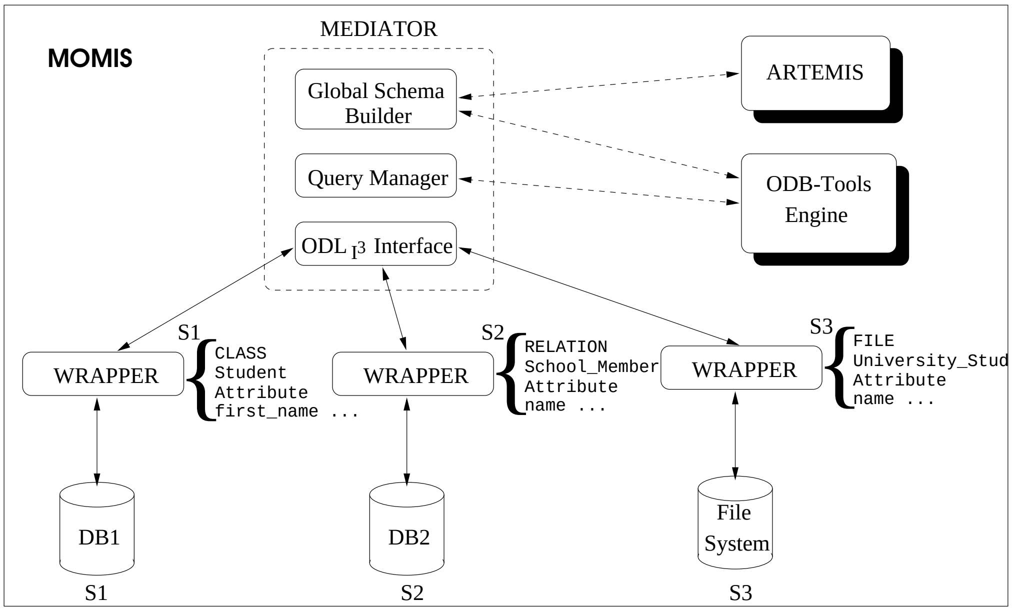

Figure 1: Architecture of the proposed I^3 system

Terminological relationships are derived in a semi-automatic way, by analyzing structure and context of classes, by using ODB-Tools and the Description Logics techniques.

2. *Evaluation of schema class affinity.*
   Terminological relationships in the Thesaurus are used to evaluate the level of *affinity* between schema classes for subsequent integration. To this end, we define proper coefficients that measure the level of affinity of schema classes based on their names and attributes [8].

3. *Cluster generation.*
   Classes with affinity are grouped together using hierarchical clustering techniques [10, 13].

4. *Mediator schema generation.*
   Unification of affinity clusters leads to the construction of the global schema of the mediator. An integrated 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 integrated classes derived from clusters, and is the basis for posing queries against the sources. In this phase, `ocdl` and ODB-Tools are exploited for a semi-automatic generation of the global schema.

2.3 Running example

Figure 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

University source ($\mathbf{S_1}$)

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

Computer_Science source ($\mathbf{S_2}$)

```
CS_Person(first_name,last_name)
Professor:CS_Person(title,belongs_to:Office,rank)
Student:CS_Person(year,takes:set⟨Course⟩,rank)
Office(description,address:Location)
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

`Room`.
For a given professor (in `Research_Staff`) his department (`dept_code`) and his section (`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`, `Office`, `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 office of a given professor. This office may be part of another department, being a logical specialization of `Department`. The class `Location` maintains the office 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 office. It consists of a file system, storing information about student's tax_fees. As for the ODL$_{I^3}$ classes `School_Member` and `Student`, we give an example of ODL$_{I^3}$ descriptions (for the complete source descriptions see appendix B).

```
interface School_Member
 ( source relational University
   extent School_Member
   key name )
{   attribute string name;
    attribute string faculty;
    attribute integer year; };
```

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

3 Semi-automatic generation of a Common Thesaurus

The goal of this phase is the construction of a Thesaurus of terminological relationships describing common knowledge about classes and attributes specified in source schemas. For this reason, it is called Common Thesaurus. The following kinds of terminological relationships are contained 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 source schemas is a semi-automatic process, enforced by the interaction between ODB-Tools and the designer. It is articulated in the following steps:

1. *Automated extraction of relationships from source schemas*: exploiting ODB-Tools capacity and semantically rich schema descriptions, a basic set of 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 of its attributes. With relational source schemas, RT relationships are extracted also from foreign keys.
   Example 1 Consider the S_1 and S_2 sources; some of the automatically derived relationships are the following:
   ⟨`Professor` NT `CS_Person`⟩
   ⟨`Student` NT `CS_Person`⟩
   ⟨`Professor` RT `Office`⟩
   ⟨`Student` RT `Course`⟩
   ⟨`Office` RT `Location`⟩
   ⟨`Course` RT `Professor`⟩
   ⟨`Research_Staff` RT `Department`⟩
   ⟨`Research_Staff` RT `Section`⟩
   ⟨`Section` RT `Room`⟩

2. *Revision/integration of relationships*: the designer can interact with the tool to supply additional terminological relationships not discovered in the previous step. Example of terminological relationships that can be interactively supplied are those regarding synonyms and domain-related knowledge in general.
   Example 2 On the example sources, the designer supplies the following relationships for classes and attributes:
   ⟨`Research_Staff` BT `Professor`⟩
   ⟨`School_Member` BT `Student`⟩
   ⟨`University_Student` BT `Student`⟩
   ⟨`Department` BT `Office`⟩
   ⟨`Section` SYN `Course`⟩
   ⟨`name` BT `first_name`⟩
   ⟨`name` BT `last_name`⟩
   ⟨`dept_code` BT `belongs_to`⟩
   ⟨`dept_name` SYN `description`⟩
   ⟨`section_name` SYN `course_name`⟩
   ⟨`faculty` SYN `faculty_name`⟩

3. *Validation of relationships*: in this step, ODB-Tools is employed to validate terminological relationships defined for attributes 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:

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 Office⟩	⟨Professor RT Section⟩
⟨Student RT Course⟩	⟨Professor RT Course⟩
⟨Course RT Professor⟩	⟨Course RT Room⟩
⟨Research_Staff RT Department⟩	⟨Student RT Section⟩
⟨Research_Staff RT Section⟩	⟨CS_Person BT University_Student⟩
⟨Professor RT Office⟩	
⟨Office RT Location⟩	
⟨Section RT Room⟩	

Figure 3: Explicit/Inferred relationships

- $\langle n_t$ 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 is equivalent to d_q.

Example 3 Referring to the Thesaurus defined in Example 2, we report the output of the validation phase: for each relationship, value [1] for the control flag denotes a valid relationship while value [0] an invalid one.

⟨name BT first_name⟩	[1]
⟨name BT last_name⟩	[1]
⟨dept_code BT belongs_to⟩	[0]
⟨dept_name SYN description⟩	[1]
⟨section_name SYN course_name⟩	[1]
⟨faculty SYN faculty_name⟩	[1]

4. *Inferring new relationships*: starting from valid explicit relationships obtained through steps 1÷3, 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 4 Relationships inferred from explicit ones in our example are shown in Fig. 3, while a graphical representation of Common Thesaurus for S_1, S_2, and S_3 is reported in Fig. 4, where thick arrows represent BT/NT relationships and thin arrows represent RT relationships. Furthermore, dashed arrows represent inferred relationships while solid ones represent explicitly given relationships.

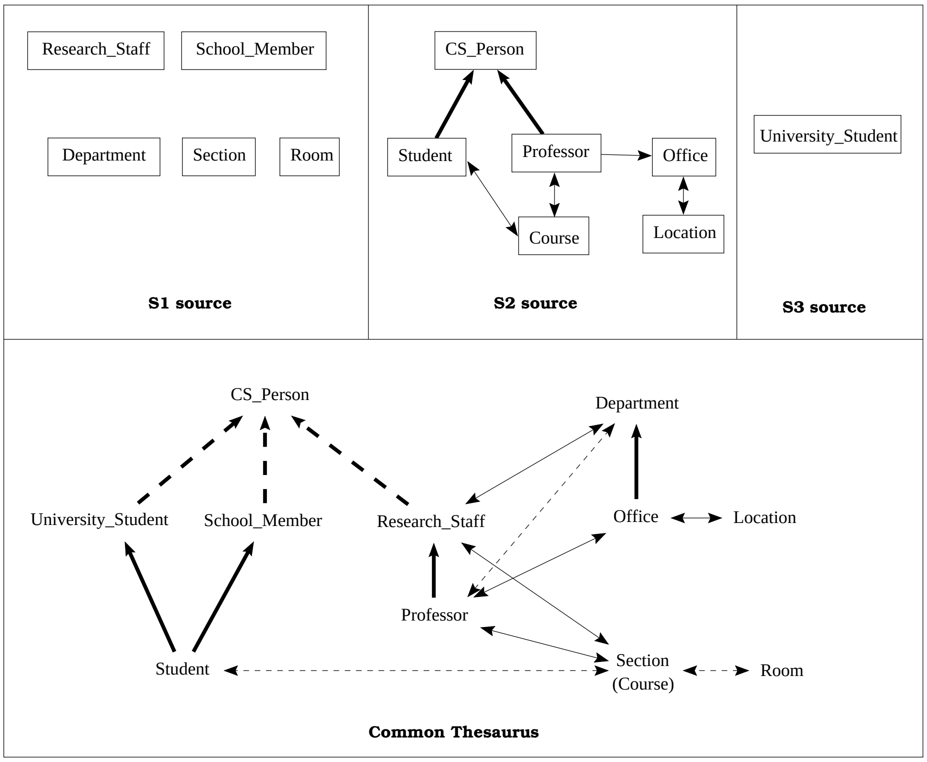

Figure 4: Common Thesaurus for S_1, S_2, and S_3.

A strength $\sigma_{\Re}$ is assigned to each type of terminological relationship $\Re$ in the Common Thesaurus, with $\sigma_{\text{SYN}} \geq \sigma_{\text{BT}} \geq \sigma_{\text{RT}}$. 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. From experimentation, we selected $\sigma_{\text{SYN}} = 1$, $\sigma_{\text{BT}} = \sigma_{\text{NT}} = 0.8$ and $\sigma_{\text{RT}} = 0.5$.

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

In this section, we present an approach to evaluate the level of affinity (i.e., the level of semantic relationship) between classes to find classes that describe the same piece of information in different schemas. It is based on the terminological relationships stored in the Common Thesaurus and on the comparison of the names and attributes of the classes. Other approaches proposed in the literature rely on terminological analysis of schema elements and on the definition of correspondence between attributes to identify and resolve semantic inconsistencies between several schemas for unification. For example, in [6] a Summary Schema Model (SSM) has been proposed to support the identification of semantically similar entities. SSM provides a unified description of the sources at a higher abstraction level in terms of the concepts of an existing domain taxonomy. Two classes are semantically related if they can be mapped to the same concept of the taxonomy. Other approaches consider also attributes and domains of schema classes. For example, in [12] two kinds of automated support to determine attribute correspondences are described, based the identification of name and domain candidate relationships to be validated by the user.

In our approach, the level of affinity of two classes c_{ji} and c_{hk} belonging to sources S_i and S_k respectively, is measured by a coefficient (called *Global Affinity coefficient*) obtained as the combination of a *Name Affinity coefficient* and a *Structural Affinity coefficient*, respectively.

Name Affinity coefficient

The Name Affinity coefficient measures classes' affinity with respect to their names. Two names have affinity if they are connected through a path in the Thesaurus. Their level of affinity depends on the length of the path, on the type of relationships involved in this path, and on their strengths. This coefficient is defined as follows.

Definition 1 (Name Affinity coefficient) The Name Affinity coefficient of two classes c_{ji} and c_{hk}, denoted by $NA(c_{ji}, c_{hk})$, is the measure of the affinity of their names $n_{c_{ji}}$ and $n_{c_{hk}}$ computed as follows.

$$NA(c_{ji}, c_{hk}) = \begin{cases} 1 & \text{if } n_{c_{ji}} = n_{c_{hk}} \\ \sigma_{\{ji\}1_{\Re}} \cdot \sigma_{12_{\Re}} \cdot \ldots \cdot \sigma_{(m-1)\{hk\}_{\Re}} & \text{if } n_{c_{ji}} \rightarrow^m n_{c_{hk}} \text{ AND} \\ & \sigma_{\{ji\}1_{\Re}} \cdot \sigma_{12_{\Re}} \cdot \ldots \cdot \sigma_{(m-1)\{hk\}_{\Re}} \geq \alpha \\ 0 & \text{otherwise} \end{cases}$$

where notation $n_{c_{ji}} \rightarrow^m n_{c_{hk}}$ denotes the path of length m ($m \geq 1$) between $n_{c_{ji}}$ and $n_{c_{hk}}$ in the Common Thesaurus with the highest affinity, and α is a threshold used to select name having high affinity.

For any pairs of names, $NA() \in [0, 1]$. The affinity of two names is 0 if a path does not exist between them in the Thesaurus, and it is 1 if the names coincide or are synonyms. In remaining cases, the longer the path defined between two names, the lower the affinity of these names in the Thesaurus. For a given path length, the higher the strength of the involved relationships, the greater the affinity of the considered names.

Example 5 Consider the relationships illustrated in Figure 3. A path exists between `Research_Staff` and `University_Student` in the Thesaurus. In fact, we have `Research_Staff` $\rightarrow^{\text{NT}}$ `CS_Person` $\rightarrow^{\text{BT}}$ `University_Student`. Therefore, $NA(\texttt{Research\_Staff}, \texttt{University\_Student}) = 0.8 \cdot 0.8 = 0.64$

In the remainder of the paper symbol $\sim$ will be used to denote affinity between names.

Structural Affinity coefficient

The Structural Affinity coefficient measures classes' affinity with respect to their attributes. They are defined using the Dice's function, which returns an affinity value in the range $[0, 1]$ proportional to the number of attributes that have affinity in the considered classes, refined by a "control factor" F_c.
The F_c factor 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, in Section 3): its value will be the ratio of positive checks to the 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 two classes. The value 0 indicates the absence of attributes with affinity in the considered classes, while the value 1 indicates that all attributes of both classes have affinity and positive checks.

Definition 2 (Structural Affinity coefficient) The Structural Affinity coefficient of two classes c_{ji} and c_{hk}, denoted by $SA(c_{ji}, c_{hk})$, is the measure of the affinity of their attributes computed as follows:

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

$$F_c = \frac{|\{x \in C| \ flag(x)=1 \ \}|}{|C|}$$

$$\mathrm{C} = \{(a_t, a_q) \mid a_t \in A(c_{ji}), a_q \in A(c_{hk}), \langle a_t \text{ SYN } a_q \rangle \text{ or } \langle a_t \text{ BT } a_q \rangle \text{ or } \langle a_t \text{ NT } a_q \rangle\}$$

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

Example 6 Consider the source schemas in Fig. 2. Consider now class `Research_Staff` in S_1 and class `University_Student` in S_3. We have that $SA(\texttt{Research\_Staff}, \texttt{University\_Student}) = \frac{2 \cdot 1}{5+4} \cdot \frac{1}{1} = 0.2\bar{2}$ due to the common attribute `name`.

Global Affinity coefficient

Definition 3 (Global Affinity coefficient) The Global Affinity coefficient of two classes c_{ji} and c_{hk}, denoted by $GA(c_{ji}, c_{hk})$, is the measure of their affinity computed as the weighted sum of the Name and Structural Affinity coefficients as follows:

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

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.

Weights in $GA(c_{ji}, c_{hk})$ 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 affinity equally relevant 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 semantically related concepts and their Structural Affinity is not further evaluated. Consequently, also the $GA()$ coefficient is equal to zero.

Example 7 Let us return to the Name and Structural Affinity coefficients of Example 5, and Example 6, respectively. The Global Affinity coefficient of `Research_Staff` and `University_Student` is computed as follows:

$$GA(\texttt{Research\_Staff}, \texttt{University\_Student}) = 0.5 \cdot 0.64 + 0.5 \cdot 0.2\bar{2} = 0.43\bar{1}$$

5 Cluster generation

To identify groups of classes having affinity in N source schemas, we employ hierarchical clustering techniques. These techniques classify classes into groups at different levels of affinity to form a tree [13].

The hierarchical clustering procedure works as follows. First of all, the affinity coefficients for all possible pairs of classes to be analyzed are computed, that is, $K \cdot (K-1)/2$, where K is the total number of classes to be analyzed. These coefficients are kept in a matrix M of rank K. An entry $M[j,h]$ of the matrix represents the affinity coefficient $GA(c_{ji}, c_{hk})$ between classes c_{ji} and c_{hk}. Clustering is iterative and starts by placing each class in a cluster by itself. Then, at each iteration, the two clusters having the greatest affinity coefficient in M are merged. While the algorithm proceeds, 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 affinity values between the newly defined cluster and each remaining cluster are also computed, by keeping the maximum $GA()$ value among the ones of merge clusters with each remaining cluster. As the result of clustering an affinity tree is obtained.

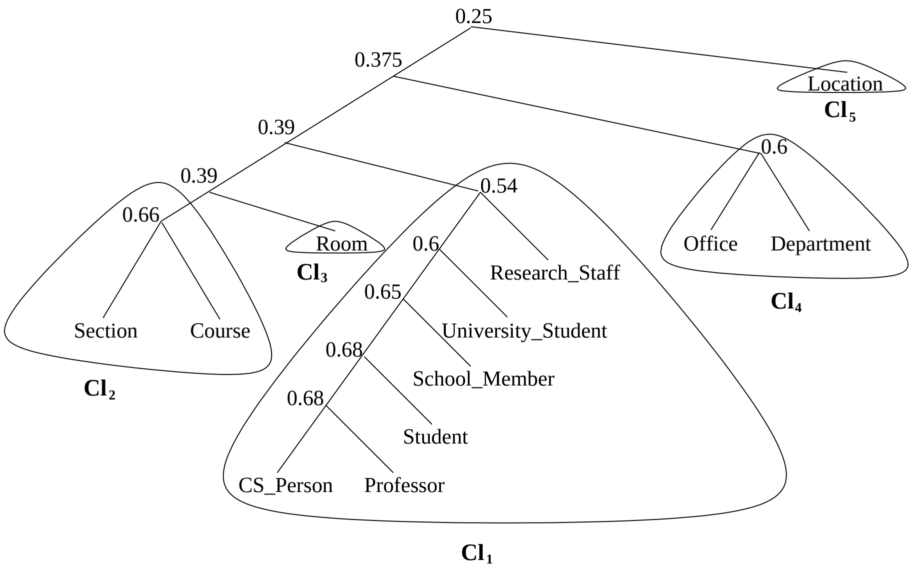

Figure 5: Affinity tree of S_1, S_2, and S_3

Figure 5 shows the affinity tree resulting from applying the clustering procedure to our set of classes.

6 Mediator schema generation

In this section we present the process which leads from the cluster generation phase to the definition of the mediator global schema, that is the mediator view of data stored in local sources. Starting from the output of the cluster generation, we define, for each cluster, a single class, i.e. *global_class*$_i$, that represents the unified view of all the classes of the cluster, and a mapping, that relates the *global_class*$_i$ to the classes of the same cluster (which may belong to different sources). The generation of the *global_class*$_i$ is obtained in two phases. In the first phase we associate to the $global\_class_i$ the union of the attributes of the classes belonging to the same cluster, after a simple substitution process which consists in replacing:

- the Synonyms Terms (preserving only one term for each list of synonym attributes);
- all the Narrower Terms (or the set of Narrower Terms) with their corresponding Broader Term.

With reference to Cluster_1 we obtain the following cluster class:

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

The second phase consists in building, for every cluster, a mapping table which relates the attributes of the $global\_class_i$ to the attributes of the classes in the associated cluster. Some more transformations are needed, besides the simple union of the attributes, in order to create the mapping table. The integration designer may in general add the following information:

University_Person	**name**	**rank**			**faculty**
Research_Staff	name	'professor'			null
School_Member	name	'student'			faculty
CS_Person	first_name *and* last_name	null			'Computer_Science'
Professor	first_name *and* last_name	rank			'Computer_Science'
Student	first_name *and* last_name	rank			'Computer_Science'
University_Student	name	'student'			faculty_name

Figure 6: University_Person mapping table

1. the $global\_class_i$ name;
2. the type of the BT or NT relationship. If two or more specialized terms have to be replaced with a single broader term, the type of the composition must be specified, selecting from this options:
   - *and* composition: a Broader Term is composed by the union of the Narrowers (e.g.: name = first_name and last_name);
   - *or* composition: the Broader Term corresponds to the Narrower Terms one at a time;
3. default values;
4. new attributes to the cluster.

With reference to point 3, we propose a simple solution, derived from the ODL_rule extension supported by ODB-Tools [3]. Using this definition language, the system integrator may add a set of *if then* rules.

With reference to the source S_2, we know that the student's faculty value is "Computer Science", even if this information is not stored in any attribute. This information may be inserted by the integration designer as an ODL_rule:

```
R1:  forall X in Student then X.faculty = 'Computer_Science'
```

In the same way, the following rules can be defined:

```
R2:  forall X in School_Member then X.rank = 'student'
R3:  forall X in Research_Staff then X.rank = 'professor'
R4:  forall X in University_Student then X.rank = 'student'
```

Furthermore, a set of statements to refine the description of *global_class*$_i$ and to map it to the source schemas must be provided (see points 1 and 2):

```
define  Cl1 as University_Person
        begin
        refine name = first_name and last_name;
        end
```

As the result of the two steps, the following *global_class*$_1$ `University_Person` is obtained:

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

and a partial view of its mapping table is shown in Figure 6. The mapping table has a tuple for each class of the cluster and the following possible values:

- *null*: attribute not present;
- *' ... '*: default value;
- *identifier* of the corresponding attribute in the source;
- *and/or composition expression* between source attributes;

The mapping table can be exploited by the Query Manager module to retrieve information from the local sources. Let us suppose the user wants to retrieve the names of the student belonging to the "Physics" faculty. The query, based on the mediator schema which is the unique view of data for the user, is the following:

```
select name
from University_Person
where rank = 'student' and faculty = 'Physics'
```

The Query Manager (realized with ODB-Tools) will automatically infer, from `ocdl` classes description and optimization rules (in particular rule R1), that no interesting data for the query are stored in the `Computer_Science` tables. Furthermore, exploiting rule R3, the class `Research_Staff` will not be queried, sending only two subqueries to the `University` source and to the `Tax_Position` file system:

```
select first_name, last_name
from School_Member
where faculty = 'Physics'
```

```
select name
from University_Student
where faculty = 'Physics'
```

7 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 Description Logics component (ODB-Tools engine) and on an affinity-based clustering component (ARTEMIS tool) 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.

Future research work will be devoted to the improvement of the approach in the direction of reducing the effort of the integration designer. Actually, a (sometimes not trivial) manual analysis activity is required to the integration designer, to supply the terminological relationships existing between the different sources not identified by the tool. In the future, we will investigate the use of a minimal top-level ontology: the idea is to force the local schema designers to take into account this common ontology, by linking the local terms to the common ones, to disambiguate them. We expect that this preliminary definition of a *rough* global schema would reduce the integration designer duty, and would help to reduce/avoid problems arising with terminological conflicts. Furthermore, the process that leads to the discovery of terminological relationships may be supported by the presence of a simple lexical system (see for example WordNet [15, 20]): synonyms, hypernyms and hyponyms may be automatically proposed to the designer, by choosing them from predefined sets (for example, we may consider an acronym and its expansion as synonym terms).

References

[1] 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.

[2] 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, Italy, 1997.

[3] D. Beneventano, C. Corni, S. Lodi, M. Vincini, "ODB-Tool: validazione di schemi e ottimizzazione semantica on-line per basi di dati object oriented", in *Proc. of Fifth National Conference of Advanced Database Systems (SEBD97)*, Verona, Italy, 1997.

[4] 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, 1997.

[5] 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.

[6] 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.

[7] 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.*

[8] 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.

[9] S. Castano, V. De Antonellis, "Deriving Global Conceptual Views from Multiple Information Sources", *Proc. of ER'97 Preconference Symposium on Conceptual Modeling: Historical Perspectives and Future Directions*, Los Angeles, November 1997, to appear.

[10] S. Castano, V. De Antonellis, M.G. Fugini, B. Pernici, "Conceptual Schema Analysis: Techniques and Applications", *ACM Transactions on Database Systems*, to appear, 1998.

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

[12] 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

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

[14] 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.*

[15] J. Gilarranz, J. Gonzalo, F. Verdejo, "Using the EuroWordNet Multilingual Semantic Database", in *Proc. of AAAI-96 Spring Symposium Cross-Language Text and Speech Retrieval*, 1996.

[16] N.Guarino, "Semantic Matching: Formal Ontological Distinctions for Information Organization, Extraction, and Integration" *Summer School on Information Extraction*, Frascati, July 1997.

[17] 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.

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

[19] 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.

[20] A.G. Miller, "WordNet: a lexical database for English", *Communications of the ACM*, Vol. 38, No.11, November 1995, pp. 39 - 41.

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

[22] S. Shenoy et al. "The Rufus System: Information Organization for Semistructured Data", in *Proc. of the 19th VLDB Conference*, Dublin, Ireland, 1993

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

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

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

The following is a BNF description for the ODL$_{I^3}$ description language.
We included the syntax fragment which differs 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⟩

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

```
UNIVERSITY source:
interface Research_Staff
 ( source relational University
   extent Research_Staff
   key name
   foreign_key dept_code, section_code )  {
```

```
interface School_Member
 ( source relational University
   extent School_Member
   key name )
 {  attribute string name;
```

```
{   attribute string name;
    attribute string relation;
    attribute string e_mail;
    attribute integer dept_code;
    attribute integer section_code; };
interface Department
  ( source relational University
    extent Department
    key code )
{    attribute string dept_name;
     attribute integer dept_code;
     attribute integer budget; };

interface Room
 (  source relational University
    extent Room
    key room_code )
{    attribute integer room_code;
     attribute integer seats_number;
     attribute string notes; };
```

```
    attribute string faculty;
    attribute integer year; };

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

```
COMPUTER_SCIENCE source:
interface CS_Person
( source object Computer_Science
   extent CS_Persons
   keys first_name, last_name )
{   attribute string first_name;
    attribute string last_name; };
interface Student : CS_Person
 ( source object Computer_Science
   extent Students )
{   attribute integer year;
    attribute set<Course> takes;
    attribute string rank; };
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 Professor : CS_Person
( source object Computer_Science
   extent Professors )
{   attribute string title;
    attribute Office belongs_to;
    attribute string rank; };
interface Office
 ( source object Computer_Science
   extent Offices
   key description )
{   attribute string description;
    attribute Location address; };
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_Student
   key student_code )
{       attribute string name;
        attribute integer student_code;
        attribute string faculty_name;
        attribute integer tax_fee;   };
```