Journal of Universal Computer Science, vol. 19, no. 13 (2013), 1986-2012
submitted: 2/11/12, accepted: 28/6/13, appeared: 1/7/13 © J.UCS

Semantic Integration of Heterogeneous Data Sources in the MOMIS Data Transformation System

Maurizio Vincini
(University of Modena and Reggio Emilia, Modena, Italy
maurizio.vincini@unimore.it)

Domenico Beneventano
(University of Modena and Reggio Emilia, Modena, Italy
domenico.beneventano@unimore.it)

Sonia Bergamaschi
(University of Modena and Reggio Emilia, Modena, Italy
sonia.bergamaschi@unimore.it)

Abstract: In the last twenty years, many data integration systems following a classical wrapper/mediator architecture and providing a Global Virtual Schema (a.k.a. Global Virtual View - GVV) have been proposed by the research community. The main issues faced by these approaches range from system-level heterogeneities, to structural syntax level heterogeneities at the semantic level. Despite the research effort, all the approaches proposed require a lot of user intervention for customizing and managing the data integration and reconciliation tasks. In some cases, the effort and the complexity of the task is huge, since it requires the development of specific programming codes. Unfortunately, due to the specificity to be addressed, application codes and solutions are not frequently reusable in other domains. For this reason, the Lowell Report 2005 has provided the guideline for the definition of a public benchmark for information integration problem. The proposal, called THALIA (Test Harness for the Assessment of Legacy information Integration Approaches), focuses on how the data integration systems manage syntactic and semantic heterogeneities, which definitely are the greatest technical challenges in the field. We developed a Data Transformation System (DTS) that supports data transformation functions and produces query translation in order to push down to the sources the execution. Our DTS is based on MOMIS, a mediator-based data integration system that our research group is developing and supporting since 1999. In this paper, we show how the DTS is able to solve all the twelve queries of the THALIA benchmark by using a simple combination of declarative translation functions already available in the standard SQL language. We think that this is a remarkable result, mainly for two reasons: firstly to the best of our knowledge there is no system that has provided a complete answer to the benchmark, secondly, our queries does not require any overhead of new code.

Keywords: Semantic Integration, Ontology Matching, Semantic Annotations, XML, Semantic Web
Categories: H.3.1, H.3.2, H.3.3, H.3.7, H.5.1

1 Introduction

In the last twenty years, several data integration systems having a classical wrapper/mediator architecture [Wiederhold, 93] and generating a Global Virtual

Schema (a.k.a. a Global Virtual View - GVV) of heterogeneous data sources have been proposed. Since the GVV provides a reconciled, integrated, and virtual view of heterogeneous sources, modelling the mappings among the data in the sources and their global representation in the GVV is a crucial aspect. Two basic approaches for specifying the mappings in a Data Integration System have been proposed in the literature: Local-As-View (LAV) [Halevy, 01], and Global-As-View (GAV), respectively [Ullman, 97].

The LAV approach is based on the idea that a global schema representing the real world already exists and the content of each local source should be described in terms of the global schema. A LAV approach suffers from some limitations: it is effective only whenever the GVV provided by the data integration system is stable and well-established in the organization. Moreover, the process for rewriting queries against the LAV into sub-queries to be executed by the constituent data sources is typically complex and requires reasoning techniques. On the other hand, as a positive aspect, the extensibility of the system is really simple in LAV systems: adding a new source simply means enriching the already existing mappings with new assertions, without other changes.

In the GAV approach, the structure of the GVV is not predefined and is described in terms of a view over the local sources. A GAV architecture facilitates the system in carrying out query processing, because it tells the system how to use the sources to retrieve data (unfolding). However, extending the system with a new source is a difficult task, since a new source may have an impact on the definition of various classes of the GVV, whose associated views need to be redefined.

Starting from our previous experience in the information integration area, where we developed the MOMIS, a mediator system following a GAV approach [Beneventano, 00, Beneventano, 01, Beneventano, 03], we developed an extension of the system devoted to the support of syntactic and semantic data heterogeneities by means of declarative transformation functions that avoids the overhead due to write ad-hoc hard-coded transformation functions. Based on this idea, we developed the Data Transformation System (DTS) of MOMIS that supports the transformation functions and generates query translation in order to push down to the sources the execution.

Each mediator system that has been proposed in the literature (see [Doan, 05] for a survey) focused on data integration issues, ranging from system-level heterogeneities, to structural syntax level heterogeneities at the semantic level. The approaches now available require the user a lot of effort for the customization of the data integration and reconciliation process. This implies in some cases the need of writing specific pieces of programming code. The specialization of the data integration systems makes the comparison among the approaches difficult.

Some frameworks have been proposed for evaluating and comparing techniques for data matching transformations. The current proposals can be classified into two categories: benchmarks providing datasets, queries and results to be achieved, and frameworks providing metrics for evaluating the complexity of the process, independently of the datasets used. The last category has recently received notable attention, since it does not strictly rely on specific scenarios and thus is supposed to provide results which are more domain independent than other benchmarks. The first kind of category includes frameworks as THALIA (Test Harness for the Assessment

of Legacy information Integration Approaches) [6], which provides researchers with a collection of downloadable data sources representing University course catalogues, a set of twelve benchmark queries, as well as a scoring function for ranking the performance of an integration system. The second category includes proposals as [Mecca, 12], where the authors provide a definition of the quality of a data translation tool on a mapping scenario and a definition of the user-effort needed to achieve such quality. These measures can be adopted for comparing the performances of different systems. Finally, STBenchmark [Alexe, 08], a benchmark composed of three components of (1) a basic suite of mapping scenarios, (2) a mapping scenario generator, and (3) a simple usability model that can be used as a first-cut measure on the ease of use of a mapping system. STBenchmark can be considered as a technique that bridges both the categories, since it includes both scenarios and a usability model, which is intended to provide a first-cut measure on the amount of effort required for a mapping scenario.

In this paper we use THALIA as a benchmark for evaluating data transformation approaches. The reason of our choice is that THALIA follows the guidelines for the definition of a public benchmark for information integration problem provided by the Lowell Report published in 2005 [Abiteboul, 05]. Consequently, it focuses on syntactic and semantic heterogeneities in order to evaluate and compare the greatest technical challenges in the field.

In this paper, we show how the system is able to manage all twelve queries of the THALIA benchmark by using a simple combination of declarative translation functions and without any overhead of new code.

The paper is structured as follows. Section 2 presents the THALIA benchmark and Section 3 the MOMIS integration methodology. Section 4 defines the mapping refinement that in Section 5 is used for the query translation. Section 6 reports the experimental results with THALIA, Section 7 the related works and finally Section 8 the conclusion.

2 The THALIA Benchmark

THALIA is a public available test bed and benchmark for information integration systems [Hammer, 05]. It provides a collection of 40 sources representing University course catalogues from computer science departments around the world. The goal of the benchmark is a systematic classification of the different types of syntactic and semantic heterogeneity.

By means of twelve queries, THALIA classifies the possible issues generating by the data heterogeneities into three categories:

- Attribute Heterogeneities (Query 1, 2, 3, 4 and 5),
- Missing Data (Query 6, 7, 8),
- Structural Heterogeneities (Query 9, 10, 11, 12).

2.1 Attribute Heterogeneities

This level of heterogeneity concern inconsistencies that exist between two single attributes in different schema. The simplest scenario is provided by the *synonyms of*

attributes, where the same information is stored in attributes with different names. In **Query 1** 'instructor' could be think as a synonyms of 'lecturer'.

Simple mapping heterogeneity refers to attributes that differ by a mathematical transformation. The issue proposed by **Query 2** is to match values of attributes containing a time value in 12 and 24 hours.

Union types: in many cases attributes in different schemas use different data types to store the same information: in **Query 3** the target schema uses a string attribute to denote the course name while the challenge schema uses only one string description for the name and the link of the course.

Complex mapping: this scenario refers to cases where related attributes differ by a complex transformation of their values: for example **Query 4** contains a number of 'credit hour' in the target schema and a string description of the expected work in the challenge schema.

A typical real case involves two sources where the same information are stored in different language, giving rise to a language expression heterogeneity. **Query 5** proposes a target schema where the course name is expressed in English and the challenge schema in German.

2.2 Missing data

This kind of heterogeneity is due to missing information (concerning values or structure elements) in one of the data source.

Nulls treatment and Semantic incompatibility: the goal of this scenario is to distinguish the case where an attribute does not exist in a source from the one where the attribute has a null value in a particular record. In **Query 6** the book for a course is not present in the challenge schema and only for some record in the target schema. **Query 8** proposes a target schema with a student classification that is not present in the challenge schema.

Virtual column: this heterogeneity refers to the situation where some information is explicit in a source and only implicitly available in the others. **Query 7** gives an example where the course prerequisites are present together with the course description in the challenge schema.

2.3 Structural Heterogeneities

This kind of heterogeneity is due to a different description of the domain of interest in the schemas of the data sources.

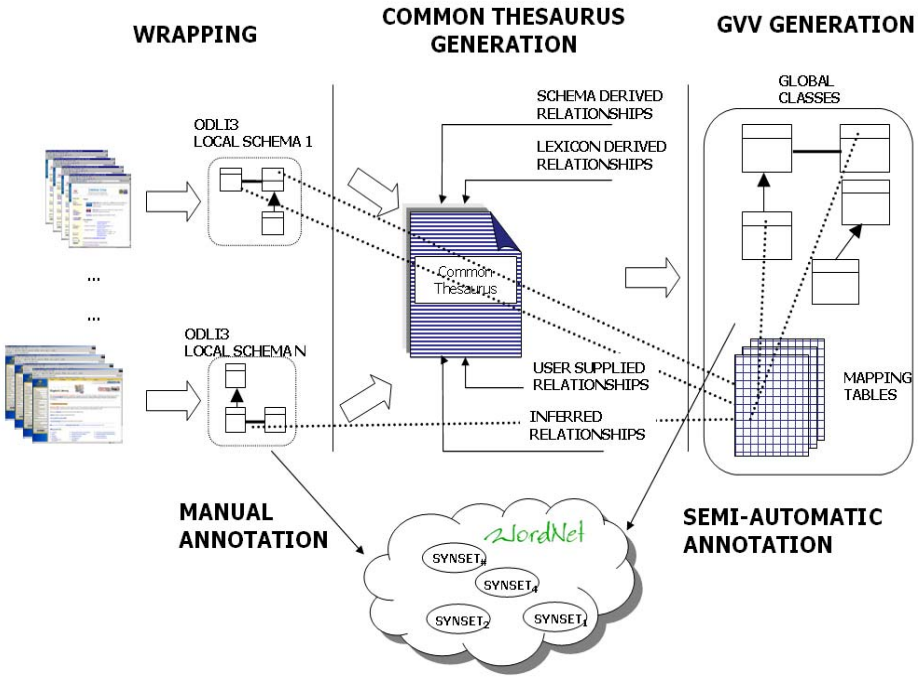

Figure 1: Ontology Generation Process

Structural heterogeneity of an attribute: the same attribute may be located in different positions in different schemas. **Query 9** proposes a target schema with the attribute *room* located in the *course* relation and a challenge schema where the room information is an element of *section* that is a part of course.

Handling sets: this case occurs when a single attribute contains a string value that describes a set of values in a schema, while the same information is split in single attributes in an other schema. **Query 10** contains a target schema with a single lecturer attribute of course and a challenge schema where a professor is defined for each section of the course.

Attribute name does not define semantics: a typical case where the attribute name does not refer to the semantic of the contained information. **Query 11** contains a challenge schema where the lecturers are spread in three different attributes whose names are the teaching periods.

Attribute composition: this scenario includes a complex data that can be represented either as a single string or a set of attributes. **Query 12** contains a challenge schema where the title, day and time of a course are represented with a single attribute rather than in three different attributes.

3 MOMIS Integration Methodology

The Mediator Environment for Multiple Information Sources (MOMIS) is a framework for extracting information and integrating heterogeneous, semi-structured information sources such as Web data sources (www.dbgroup.unimo.it/momis/).

Unlike other data-integration systems that follow the local-as-view (LAV) approach, which is based on the idea that each source's content should be represented by predefined global schema, MOMIS implements a semiautomatic methodology that follows the global-as-view (GAV) approach: the obtained global schema is expressed in terms of the data sources. More precisely, to each element of the global schema, a view over the data sources is associated, so that its meaning is expressed as the data residing at the sources. MOMIS uses ODLI3, which is based on the Object Definition Language (ODL) to describe both the input (the sources) and the result of the synthesis process (global virtual view).

MOMIS generates a global schema that provides an integrated GVV composed of a set of global classes that represent the information contained in the underlying sources and the mappings that establish the connections among the global attributes of the global classes and the source schemas. Since a GVV conceptualizes a domain, it might be thought of as an ontology for the integrated sources.

In the following we describe the MOMIS Ontology generation process by means of three different THALIA's queries (query 4, 7 and 12), each one referring to a different heterogeneities category.

3.1 Ontology generation

The Ontology Generation process can be outlined as follows (see Figure 1):

1. *Extraction of Local Source Schemas*: Wrappers acquire schemas of the involved local sources and convert them into ODLI3, a language used internally to MOMIS for representing the data sources. Schema descriptions of traditional structured sources (e.g. relational database and object-oriented database) can be directly translated into ODLI3. The extraction of schemas from semi-structured sources needs the development of specific techniques as usual for similar applications [Abiteboul, 00]. To perform information extraction from XML Schema files, as other systems [Du, 04], we developed a wrapper that automatically translates the XSD schema into relational structures and imports data into a DBMS. All schemas and data provided by THALIA benchmark (10 databases and 701 records) can be automatically wrapped by our tool and translated into a DBMS.

2. *Local Source Annotation*: Terms denoting schema elements in data sources are semantically annotated according to a common lexical reference to provide a shared meaning to each of them. We chose the WordNet database (wordnet.princeton.edu) as lexical reference. The system automatically detects, for each term in the sources, the (most commonly) used meaning present in WordNet. An algorithm for automatic annotation prepares the terms by removing stop-words and applying stemming functions to enhance the accuracy of the result [Bergamaschi, 07]. Then the Ontology Designer is

supported by a graphical interface in manually revising the meaning(s) associated to each annotated term. Considering the THALIA schemasdata sources, the recall rate of the terms automatically annotated in the sources is 80%, with a precision of 82%.

3. *Common Thesaurus Generation*: MOMIS builds a Common Thesaurus that describes intra and inter-schema knowledge in the form of: synonyms (SYN), broader terms/narrower terms (BT/NT), meronymy/holonymy (RT) relationships. The Common Thesaurus is incrementally built by starting from schema-derived relationships, i.e. automatic extraction of intra-schema relationships from each schema separately. Then, the relationships existing in the WordNet database between the annotated meanings are exploited to generate relationships between the respective elements that are called lexicon-derived relationships. The Ontology Designer may add new relationships to capture specific domain knowledge. Finally, a Description Logics reasoner, ODB-Tools [Bergamaschi, 97], which performs equivalence and subsumption computation) computes the transitive closure of Common Thesaurus relationship and infers new relationships.

4. *GVV generation*: Starting from the Common Thesaurus and the local sources schemas, MOMIS generates a GVV consisting of a set of global classes, plus mappings to connect the global attributes of each global class and the local sources' attributes. Going into details, the GVV generation is a process where ODLI3 classes describing the same or semantically related concepts in different sources are identified and clustered in the same global class by means of the ARTEMIS tool. ARTEMIS determines the degree of matching of two classes, based on their names and their structure, and produces an affinity tree. Clusters for integration are interactively selected from the affinity tree using a non-predefined threshold based mechanism. The Ontology Designer may interactively refine and complete the proposed integration results; in particular, the mappings which have been automatically created by the system can be fine-tuned by means of the translation functions, as discussed in next section. For each GVV involving a THALIA's query, MOMIS automatically detects the correct basic relations and attributes mapping. For example, the GVV created for solving Query 1 automatically contains the match between Instructor and Lecturer attribute, i.e. the THALIA foreseen challenge. For Query 12's GVV, the system recognizes the mapping between CourseTitle and Title of the two schemas, while the challenge, i.e. information contained in a unique attribute rather than separate attributes, is achieved by means of the translation functions.

5. *GVV annotation*: The GVV is automatically annotated, i.e. each of its elements is associated to the broadest meanings extracted from the annotated sources. The annotation of a GVV is a significant result, since these metadata can be useful to make the meaning of the created GVV understandable to external users and applications. As an example, we report the following global class meaning (Query 12):

Global class / attribute	Local Classes	Meaning (from WordNet)
Course	Cmu.Course Brown.Course	Course#1
Instructor	Cmu.Lecturer Brown.Instructor	Instructor#1
Title	Cmu.CourseTitle Brown.Title	Title#1

4 Mapping refinement

During the GVV generation process, the system automatically generates a Mapping Table (MT) for each global class C of the GVV, whose columns represent the local classes L(C) belonging to C and whose rows represent the global attributes of C. An element MT [GA][LC] represents the set of local attributes of LC which are mapped onto the global attribute GA.

After this automatic step, the Designer can refines the MT by adding:

- Data Transformation Functions applied to local attributes
- Join Conditions between pairs of local classes belonging to C
- Resolution Functions for global attributes to solve data conflicts of local attribute values.

4.1 Data Transformation Function

The Designer can define, or refine, for each element MT[GA][L], a Data Transformation Function, denoted by MTF[GA][L], which represents the mapping of local attributes of L into the global attribute GA. MTF[GA][L] is a function that has to be executable/supported at the local source by means of the local source wrapper. In fact we want, intuitively, to push as much as possible the function execution to the sources, as proposed in [Chang, 99] for the constraint mapping.

The system prototype accepts the following SQL-92 like functions:

CHAR_LENGTH: returns the length of a string

POSITION: searches a pattern in a string

SUBSTRING: returns a part of a string

CAST: converts a value from a type to another

CASE … WHEN … THEN: transforms a record on the basis of a specific data value

In addition, also the classic RIGHT and LEFT string functions, i.e. the first (or the last) n characters of a string, are available in the system. All these functions are executed at the wrapper level by the translation of the particular SQL-dialect for the relational DBMSs and are built-in for other wrappers (like XML).

Finally, we define a specific function for datetime type conversion:

TIME12-24: transforms a string to a time value expressed in 12 or 24 hours format.

In the following, we show how the aforementioned functions have been exploited for solving some of the THALIA queries.

Example Query 4: a complex transformation function is required to convert the string that describes the expected scope of the course in the "ethz" table into a credit unit. The conversion formula is provided by the ETH's Computer Science:

#KE = #V + #U + 1

To solve this query, the designer should specify in the mapping table attribute a combination of the transformations functions:

```
MTF[Unit][ethz.Unterricht] =
          CAST(SUBSTRING(Umfang, POSITION('V' IN Umfang) - 1, 1) AS int)
        + CAST(SUBSTRING(Umfang, POSITION('U' IN Umfang) - 1,1) AS int)
            + 1
```

Example Query 7: the transformation exploits the information that is attached to the description to infer if a course has prerequisite. In particular, the function searches and return the text that follows the 'Prerequisite' term.

```
MTF[prerequisite][asu.Course] =
          CASE POSITION('%Prerequisite%' IN Description)
          WHEN 0 THEN 'None'
          ELSE RIGHT(Description, CHAR_LENGTH(Description) -
                        POSITION('%Prerequisite%' IN Description) + 1)
          END
```

Example Query 12: the challenge is to extract the correct title, day and time values from the title column in the catalog of Brown University that contains all the information. By using a combination of SUBSTRING and POSITION functions, it is possible to solve the query in a declarative way.

```
MTF[Title][brown.Course] =
```

```
                SUBSTRING(Title FROM POSITION('/"' IN Title) + 3 FOR
                    POSITION ('hr.' IN SUBSTRING(Title FROM
                                POSITION('/"' IN Title) + 3 FOR 100)) - 1)

MTF[Day][brown.Course] =
                SUBSTRING(Title FROM POSITION('hr.' IN Title) + 4 FOR
                    POSITION(' ' IN SUBSTRING(Title FROM
                                POSITION('hr.' IN Title) + 4 FOR 10)))
MTF[Time][brown.Course] =
                SUBSTRING(Title IN POSITION(' ' FROM SUBSTRING(Title
                        FROM POSITION('hr.' IN Title) + 4 FOR 10)) +
                            POSITION('hr.' IN Title) + 4 FOR 15)
```

The transformation of a local class L obtained by applying all the Data Transformation Functions MTF[GA][L] is denoted with T(L).

4.2 Join condition

Merging data from different sources requires different instantiations of the same real world object to be identified; this process is in general called as object identification [Naumann, 02]. Object identification is a very active research area with significant contributions both from the artificial intelligence [Tejada, 01] and database communities [Ananthakrishna, 02, Chaudhuri, 03]. We assume, for sake of simplicity, that the ontology designer is able to define join conditions among local classes.

The Join Conditions among pairs of local classes belonging to the same global class are introduced in order to identify instances of the same object and fuse them. Given two local classes L1 and L2 belonging to C, a Join Condition between L1 and L2, denoted with JC(L1,L2), is a Boolean expression of atomic constraints (L1.Ai Op L2.Aj) where Ai (Aj) are global attributes with a not null mapping in L1 (L2) and Op is a relational operator.

As an example, for the Query 4, the designer should define the following join condition:

```
JC(L1, L2) : L1.CourseName = L2. Titel
            WHERE L1= cmu.Course AND L2= ethz.Unterricht
```

4.3 Resolution Function

The fusion of data coming from different sources and taking into account the problem of inconsistent information among sources is a challenging research topic [Naumann, 02, Di Giacomo, 04, Bertossi, 03, Greco, 03, Lin, 98].

In MOMIS, the approach proposed in [Naumann, 02] has been adopted: a Resolution Function for solving data conflicts may be defined for each global attribute mapping onto local attributes coming from more than one local source.

A global attribute with no data conflicts (i.e. the instances of the same real object in different local classes having the same value for this common attribute), is called Homogeneous Attribute. Of course, for homogeneous attributes, resolution functions

are not necessary (a global attribute mapped onto only one source is a particular case of an homogeneous attribute).

As an example, for solving Query 1 we should define a precedence function for Course Title:

gatech.Course.Title has a higher precedence than cmu.Course.CourseTitle.

4.4 Mapping query

MOMIS follows a GAV approach, thus for each global class C, a *mapping query* QC over the schemas of a set of local classes L(C) must be defined. By exploiting the enriched MT, the system automatically generates the mapping query QC associated to C, by extending the Full Disjunction (FD) operator [Galindo-Legaria, 94], been recognized as providing a natural semantics for data merging queries [Rajaraman, 96]. In our context: given a global class C composed of L1,L2, ..., Ln, we consider FD(T(L1),T (L2),...,T (Ln)), computed on the basis of the Join Conditions.
With two classes, FD corresponds to the full (outer) join:

FD(T (L1),T (L2)) = T (L1) full join T (L2) on (JC(L1,L2)).

For the complete definition and computation of FD see [Bergamaschi, 11]. Finally, QC is obtained by applying Resolution Functions to the attributes resulting from FD: for a global attribute GA we apply the related Resolution Function to T (L1).GA, T (L2).GA, . . . , T (Lk).GA.

5 Query Translation in MOMIS DTS

To answer a query expressed on the GVV (global query), the query must be rewritten as an equivalent set of queries expressed on the local schemas (local queries); this query translation is performed by considering the mapping between the GVV and the local schemas. In a GAV approach, the query translation is performed by means of query unfolding, i.e., by expanding a global query on a global class C of the GVV according to the definition of the mapping query QC as defined in section 4.4.

In this section we present the methodology of MOMIS Data Translation System for query unfolding.

5.1 Query Unfolding

The query unfolding process is performed for a Global Query Q over a global class C of the GVV

Q = SELECT <Q_SELECT-list> FROM C WHERE <Q_condition>

where <Q_condition> is a Boolean expression of positive atomic constraints:
(GA1 op value) or (GA1 op GA2), with GA1 and GA2 attributes of C.

The query unfolding process is made up of the following three steps:

Step 1) Generation of Local Queries:
LQ = SELECT <SELECT-list>
FROM L
WHERE <condition>

where L is a local class related to C.

The <SELECT-list> is computed by considering the union of:

- the global attributes in <Q_SELECT-list> with a not null mapping in L,
- the global attributes used to express the join conditions for L,
- the global attributes in <Q_condition> with a not null mapping in L.

The set of global attributes is transformed in the corresponding set of local attributes on the basis of the Mapping Table.

The <*condition*> is computed by performing an *atomic constraint mapping*: each atomic constraint of <*condition*> is rewritten into one constraint that is supported by the local source. The atomic constraint mapping is performed on the basis of the *Data Conversion Functions* and *Resolution Functions* defined in the Mapping Table. For example, if the numerical global attribute GA is mapped onto L1 and L2, and we define AVG as resolution function, the constraint (*GA = value*) cannot be pushed at the local sources, because AVG has to be calculated at a global level.

In this case, the constraint is mapped as true in both the local sources. On the other hand, if GA is an homogeneous attribute the constraint can be pushed at the local sources. For example, an atomic constraint (GA op value) is mapped onto the local class L as follows:

(MTF [GA][L] op value)	**if** MT [GA][L] is not null and the op operator is supported into L
true	**otherwise**

An atomic constraint (GA1 op GA2) is mapped in a similar way.

Step 2) Generation of FD(LQ1,LQ2, … ,LQn) which computes the Full Disjunction of the LQs
Step 3) Generation of the final query (application of Resolution Functions):

- for Homogeneous Attributes we can take one of the values;
- for non-Homogeneous Attributes (e.g. Address) we apply the associated Resolution Function (in this case the precedence function).

5.2 Multilingual query condition

In an information integration scenario, frequently the data are expressed in different languages, for example a source contains data in English and another in German o Italian.

The MOMIS DTS provides the possibility of formulation a query condition in a specific language, for example in English, and the query rewriting process tries to translate each local source query in a suitable condition.

This operation is performed by a TRANSLATION function, that translates the words from the query language into the specific languages used in the local sources. In our application, the translation is obtained by exploiting the open dictionary published by the Gutenberg Project (www.gutenberg.org) and determines, for each word, the translation in another language, that is a metadata associated to each source during the integration phase.

More precisely, given a global attribute GA, a multilingual constraint is "GA op TRANSLATE(term,Language)". The condition of a Global query is then a Boolean expression of atomic and multilingual constraints.

Given two languages, Language1 and Language2, and a term of Language1, we consider a function TRANSLATION(term,Language1,Language2) whose result is a set of terms of Language2: {term_1, …, term_n }, with n ≥ 1.To perform the constraint mapping (see Step 1 above) of a multilingual constraint

AG LIKE TRANSLATE(term,Language)

w.r.t. a local class L we consider a preliminary step where the multilingual constraint is transformed into a disjunction of atomic constraints:

GA LIKE term_1 OR …. OR GA LIKE term_n

where term_i, 1 ≤ I ≤ n, is a term obtained by TRANSLATION(term,Language,Language2), and Language2 is the language of the local class L. Then the disjunction of atomic constraints is mapped into the local class L as discussed before.

Example Query 5: the challenge is related to the expression of the attribute values in different languages. For example, in the target schema the course name is expressed in English and in the challenge schema in German language.

The global query applied to MOMIS DTS is the following:

```
SELECT Name
FROM Course
WHERE Name LIKE TRANSLATE('%Database%', 'en')
```

For the English target schema (umd), no translation is required, so the rewritten local query is:

```
SELECT CourseName
FROM Course
WHERE CourseName LIKE '%Database%'
```

While, in the challenge schema (ethz), the language is German, thus the local query becomes:

```
SELECT Titel
FROM Unterricht
WHERE Titel LIKE '%Datenbank%'
OR Titel like '%Datei%'
OR Titel like '%Datenbasis%'
```

5.3 An example of Query Translation

To show a complete example of Query Translation, we consider the following query

Example Query 4: the benchmark query 'List all database courses that carry more than 10 credit hours' has been written in our system as the following global query:

```
SELECT Title, Units
FROM  Course
WHERE Title LIKE TRANSLATE('%Database%', 'en') and Units > 10
```

The (portion of) the Mapping Table of the class Course involved in the query is the following:

Course	ethz.Unterricht	Cmu.Course
Title	Titel	CourseTitle
Units	MTF[Unit][ethz.Unterricht]	Units

This global query is automatically rewritten, by means of the translation function of the mapping table, for the target schema (cmu) and for the challenge schema (ethz)

```
Local Query 1 - Source "cmu" (LQ1)
SELECT Course.CourseTitle, Course.Units
FROM Course
WHERE (CourseTitle) like ('%Database%')
AND Units > 10
```

```
Local Query 2 - Source "ethz" (LQ2)
SELECT Unterricht.Titel,
(CAST(SUBSTRING(Umfang, CHARINDEX('V', Umfang) - 1,
1) AS int) + CAST(SUBSTRING(Umfang, CHARINDEX('U',
Umfang) - 1, 1) AS int) + 1) AS Umfang
FROM Unterricht
WHERE ((Titel) like ('%Datenbank%') or (Titel) like ('%Datei%')
or (Titel) like ('%Datenbasis%')) AND
```

```
(CAST(SUBSTRING(Umfang, CHARINDEX('V', Umfang) - 1,
1) AS int) + CAST(SUBSTRING(Umfang, CHARINDEX('U',
Umfang) - 1, 1) AS int) + 1) > 10

Join Global Query (JGQ1)
SELECT LQ1.CourseTitle AS Title_1, LQ2.Titel AS Title_2,
LQ1.Units as Units_1, LQ2.Umfang AS Units_2
FROM LQ1 full outer join LQ2 on LQ1.CourseTitle = LQ2.Titel

Final Global Query
SELECT resolution(Title_1, Title_2) AS Title, resolution(Units_1, Units_2) as Units
FROM JGQ1
```

The records obtained after the query execution are shows in a grid, where, for each attribute of a single record, the user can visualize the local source that has provided the data.

6 THALIA Benchmark: Experimental Results

In this section we describe the result of the MOMIS DTS applied to the THALIA benchmark. MOMIS DTS is fully written in Java and distributed under GPL licence at www.datariver.it; MOMIS DTS is also available via web interface.

The THALIA benchmark provides twelve queries in XML format, while MOMIS DTS provides an SQL-like syntax as a query language. As a preliminary step we have transformed the benchmark queries into SPJ queries by a straightforward and ease operation, with no effect to the benchmark result. In addition, our XML-Schema wrapper generated a relational view of the database schemas and automatically loaded the xml data file into a (generic) RDBMS, thus each data set provided by THALIA is loaded by the wrapper in a specific database.

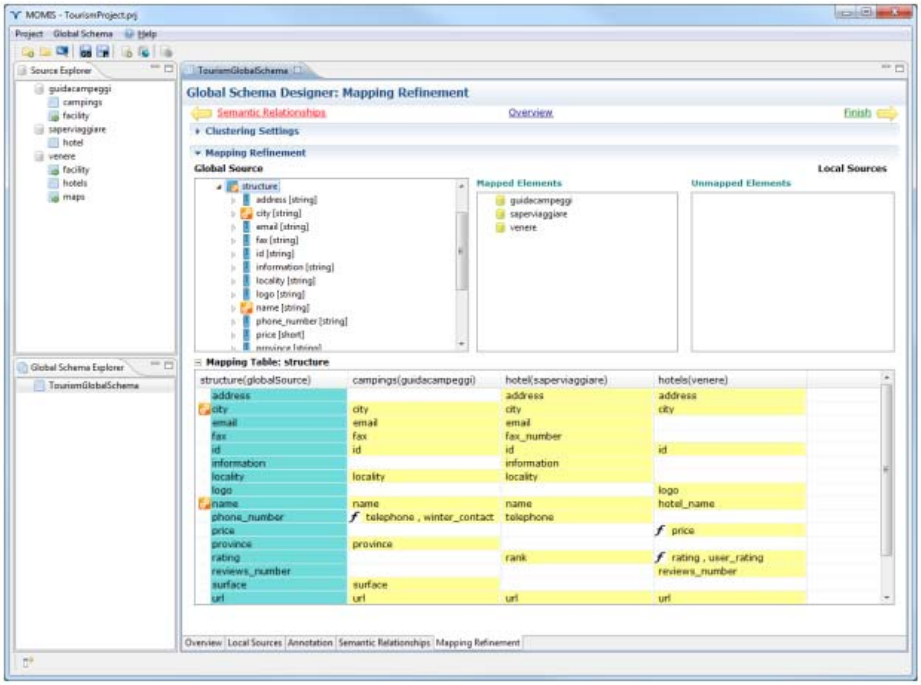

Figure 2: MOMIS Schema Mapping example

6.1 Integration phase

During this phase, for each pair of target and challenge schema related to a single query, the designer semi-automatically created a specific GVV, i.e. twelve GVVs have been deployed for the benchmark.

Since the reference schemas are very simple, the GVV creation has been very simple and completely automatic. The designer was only in charge to the refinement of the mappings computed. As an example we describe the GVV building process related to the Query 1.

The reference schemas are of Georgia Tech University and of Carnegie Mellon University.

Georgia Tech University schema contains only a Course class with several attributes such as Title, Section, Instructor, Room, Description, …

Also Carnegie Mellon University contains only Course class with attributes like CourseTitle, Room, Lecturer, Time, Unit, …

During the mapping refinement phase, for each query, a specific composition of translation functions has been inserted to overcome the challenge. Appendix A reports the mapping refinement for each query, and the following table summarizes the translation functions used for each query challenge.

Attribute Heterogeneities	
Query 1	**Only Attributes mapping**
Query 2	**TIME12-24, SUBSTRING**
Query 3	**SUBSTRING, POSITION**
Query 4	**CAST, SUBSTRING, POSITION**
Query 5	**TRANSLATE**
Missing data	
Query 6	**Attributes mapping, NULL treatment**
Query 7	**CASE WHEN … THEN, CHAR_LENGTH, RIGHT, POSITION**
Query 8	**Attributes mapping, NULL treatment**
Structural Heterogeneities	
Query 9	**SUBSTRING, POSITION**
Query 10	**SUBSTRING, POSITION**
Query 11	**CASE WHEN … THEN, CHAR_LENGTH**
Query 12	**SUBSTRING, POSITION**

6.2 Query phase

During this phase, the user has to write the queries over the integrated GVVs and the system, after the unfolding and the rewriting operations returns the records of the local sources (retrieved both from the target and the challenge schema) that satisfy the request.

MOMIS DTS provides a command line tool for the query and a grid interface for the data answer.

Example Query 4: the benchmark query 'List all database courses that carry more than 10 credit hours' has been written in our system as the following global query:

```
SELECT Title, Units
FROM Course
```

```
WHERE Title LIKE TRANSLATE('%Database%', 'en') and Units > 10
```

This global query is automatically rewritten, by means of the translation function of the mapping table, for the target schema (cmu) and for the challenge schema (ethz)

Local Query 1 - Source "cmu" (LQ1)

```
SELECT Course.CourseTitle, Course.Units
FROM Course
WHERE (CourseTitle) like ('%Database%')
AND Units > 10
```

Local Query 2 - Source "ethz" (LQ2)

```
SELECT Unterricht.Titel,
(CAST(SUBSTRING(Umfang, CHARINDEX('V', Umfang) - 1,
1) AS int) + CAST(SUBSTRING(Umfang, CHARINDEX('U',
Umfang) - 1, 1) AS int) + 1) AS Umfang
FROM Unterricht
WHERE ((Titel) like ('%Datenbank%') or (Titel) like ('%Datei%')
or (Titel) like ('%Datenbasis%')) AND
(CAST(SUBSTRING(Umfang, CHARINDEX('V', Umfang) - 1,1) AS int)
+ CAST(SUBSTRING(Umfang, CHARINDEX('U', Umfang) - 1, 1)
    AS int) + 1) > 10
```

Join Global Query (JGQ1)

```
SELECT LQ1.CourseTitle AS Title_1, LQ2.Titel AS Title_2,
        LQ1.Units as Units_1, LQ2.Umfang AS Units_2
FROM LQ1 full outer join LQ2 on LQ1.CourseTitle = LQ2.Titel
```

Final Global Query

```
SELECT resolution(Title_1, Title_2) AS Title,
        resolution(Units_1, Units_2) as Units
FROM JGQ1
```

The records obtained after the query execution are shown in a grid, where, for each attribute of a single record, the user can visualize the local source that has provided the data.

6.3 Experimental comparison

Three different integration systems Cohera, Integration Wizard (IWIZ) [Hammer, 05] and a 'keyword join' system [Yu, 06] have reported THALIA benchmark results. In the following table we compare the results, and we provide a rough specification of the extra effort required for query answer.

Query	Cohera	Integration Wizard	'Keyword join'
QUERY 1	YES	YES, SMALL	YES
QUERY 2	YES, SMALL	YES, SMALL	NO
QUERY 3	YES, MODERATE	YES, MODERATE	NO
QUERY 4	NO	NO	YES, difficult
QUERY 5	NO	NO	YES, difficult
QUERY 6	YES	YES, MODERATE	YES
QUERY 7	YES, MODERATE	YES, MODERATE	NO
QUERY 8	NO	NO	NO
QUERY 9	YES	YES, SMALL	YES, need semantic metadata
QUERY 10	YES	YES, SMALL	YES, need semantic metadata
QUERY 11	YES, MODERATE	YES, MODERATE	YES
QUERY 12	YES, MODERATE	YES, MODERATE	YES

SMALL: small amount of code
MODERATE: moderate amount of code

Summarizing, Cohera and IWIZ can solve 9 queries, some of these by adding a significant amount of code, while the system presented in [Yu, 06] could deal with 5 queries easily, and another 2 queries with small amount of metadata, without any custom code.

In MOMIS DTS, by means of declarative functions, it is very easy to deal with all 12 queries: we estimate that an information technology worker could define the requested customization (reported in Appendix A) within 4-6 hours.

In additions, we evaluated how three main commercial data integration tools provided by IBM, ORACLE and MICROSOFT are able to solve the THALIA Benchmark: the results of our experiments are shown in the following table.

Query	IBM Information Integrator	Oracle Data Integrator	Microsoft Integration Services
QUERY 1	YES, SMALL	YES, SMALL	YES, SMALL
QUERY 2	YES, difficult	YES, SMALL	YES, SMALL
QUERY 3	YES, SMALL	YES, SMALL	YES, SMALL
QUERY 4	YES, difficult	YES, SMALL	YES, SMALL
QUERY 5	YES, SMALL	YES, SMALL	YES, SMALL
QUERY 6	YES, MODERATE	YES, SMALL	YES, SMALL
QUERY 7	YES, SMALL	YES, SMALL	YES, SMALL
QUERY 8	YES, MODERATE	YES, SMALL	YES, SMALL
QUERY 9	NO	YES, SMALL	YES, SMALL
QUERY 10	NO	YES, SMALL	YES, SMALL
QUERY 11	YES, difficult	YES, MODERATE	YES, MODERATE
QUERY 12	YES, difficult	YES, MODERATE	YES, MODERATE

Basically, Microsoft Integration Services and ORACLE Data Integrator solve all queries with a small amount of adjunctive code. Both the systems are completely manual ETL tools, that extract data from the source and provide a set of transformation functions in order to load a target source suitable for the user query (in our case the THALIA queries): these architectures permit to solve all the queries, but force a scheduled ETL process for data refreshing. On the contrary IBM Information Integration proposes a virtual approach (similar to a federate database architecture), but queries 9 and 10 are not executable and the other require to write a lot of additionally java code.

7 Related Work

In the area of heterogeneous information integration, many projects based on mediator architectures have been developed. The mediator-based TSIMMIS project [Li, 98] follows a "structural" approach and uses a self-describing model (OEM) to represent heterogeneous data sources and the MSL (Mediator Specification

Language) rule to enforce source integration. In TSIMMIS, by means of MSL, arbitrary views (in particular, recursive views) can be defined at the mediator layer. The MOMIS system made a different choice: starting from the semi-automatic generated mappings between global and local attributes stored in the mapping tables, views (global classes) are defined by means of a predefined operator, i.e. the full disjunction, which has been recognized as providing a natural semantics for data merging queries. In particular, in the view definition resolution functions are defined to take into account data conflicts.

The SIMS [Knoblock, 00] proposes the creation of a global schema definition by exploiting the use of Description Logics (i.e., the LOOM language) for the description of information sources. The use of a global schema allows both GARLIC and SIMS projects to support every possible user queries on the schema instead of a predefined subset of them.

The Information Manifold system [Levy, 98] provides a source independent and query independent mediator. The input schema of Information Manifold is a set of descriptions of the sources and the integrated schema is mainly defined manually by the designer, while in our approach it is tool-supported.

The goal of Clio [Miller, 01] is to develop a tool for semiautomatically creating mappings between two data representations (i.e., with user input). First of all, in the Clio framework the focus is on the schema mapping problem in which a source is mapped onto a different, but fixed, "target" schema, while the focus of our proposal is the semi-automatic generation of a "target" schema, i.e. the Global Virtual View, starting from the sources. Moreover, the semi-automatic tool for creating schema mapping, developed in Clio, employs a mapping-by-example paradigm that relies on the use of value mappings describing how a value of a target attribute can be created from a set of values of source attributes.

Our proposal for creating schema mappings can be considered orthogonal with respect to this paradigm. In fact, the main techniques of mapping construction rely on the meanings of the class and attribute names selected by the designer in the annotation phase and by considering the semantic relationships between meanings coming from the common lexical ontology. On the other hand, MOMIS and CLIO share a common mapping semantics among a (target) global schema and a set of source schemas expressed by the full-disjunction operator. Infomaster [Genesereth, 97] provides integrated access to multiple distributed heterogeneous information sources giving the illusion of a centralized, homogeneous information system. The main difference of this project w.r.t. our approach is the lack of a tool aid-support for the designer in the integration process.

8 Conclusion

In this paper we present the Data Transformation System (DTS) of MOMIS and we demonstrate the capability by responding to all challenges provided by THALIA benchmark. The future work is devoted to enhance the TRANSLATION function by exploiting bing translator (http://www.bing.com/translator/), by means of the API provided by Jonathan Feinberg (http://babel.mrfeinberg.com).

References

[Abiteboul, 00] Abiteboul S., Buneman P., Suciu D. Data on the Web: From relations to semistructured data and XML. Data Management Systems. Morgan Kaufmann (2000).

[Abiteboul, 05] Abiteboul S., Agrawal R., Bernstein P. A., Carey M. J., Ceri S., Croft W. B., DeWitt D. J., Franklin M. J., Garcia-Molina H., Gawlick D., Gray J., Haas L. M., Halevy A. Y., Hellerstein J. M., Ioannidis Y. E., Kersten M. L., Pazzani M. J., Lesk M., Maier D., Naughton J. F., Schek H., Sellis T. K., Silberschatz A., Stonebraker M., Snodgrass R. T., Ullman J. D., Weikum G., Widom J., Zdonik S. B. The Lowell database research self-assessment. Commun. ACM 48(5): 111-118 (2005).

[Alexe, 08] Alexe B., Tan W. C., Velegrakis Y.: STBenchmark: towards a benchmark for mapping systems. PVLDB 1(1): 230-244 (2008)

[Ananthakrishna, 02] Ananthakrishna, R., Chaudhuri, S., Ganti, V. Eliminating fuzzy duplicates in data warehouses. In VLDB Conference, (pp. 586–597) 2002.

[Beneventano, 00] Beneventano, D., Bergamaschi, S., Corni, A., Guidetti, R., Malvezzi, G., Vincini, M. Information integration: The MOMIS project demonstration. Proceedings of the 26th International Conference on Very Large Data Bases, VLDB'00 , pp. 1-4, (2000).

[Beneventano, 01] Beneventano D., Bergamaschi S., Guerra F., Vincini M. The momis approach to information integration. ICEIS 2001 - Proceedings of the 3rd International Conference on Enterprise Information Systems, (2001).

[Beneventano, 03] Beneventano D., Bergamaschi S., Guerra F., Vincini M. Synthesizing an Integrated Ontology. IEEE Internet Computing 7(5): 42-51 (2003).

[Bergamaschi, 97] Bergamaschi, S., Beneventano, D., Sartori, C, Vincini, M. ODB-QOptimizer: A tool for semantic query optimization in OODB. Proceedings - International Conference on Data Engineering, pp. 578, 1997.

[Bergamaschi, 07] Bergamaschi, S., Bouquet, P., Giacomuzzi, D., Guerra, F., Po, L., Vincini, M. MELIS: An incremental method for the Lexical annotation of domain ontologies. International Journal on Semantic Web and Information Systems 3 (3) , pp. 57-80, 2007.

[Bergamaschi, 11] Bergamaschi, S., Guerra, F., Orsini, M., Sartori, C., Vincini, M. A semantic approach to ETL technologies. In Data and Knowledge Engineering 70 (8) , pp. 717-731, 2011.

[Bertossi, 03] Bertossi, L. E., Chomicki, J. Query answering in inconsistent databases. In J. Chomicki, R. van der Meyden, & G. Saake (Eds.), Logics for Emerging Applications of Databases (pp. 43–83) 2003. Springer.

[Chang, 99] Chang K., Garcia-Molina H. Mind Your Vocabulary: Query Mapping Across Heterogeneous Information Sources. SIGMOD Conference 1999: 335-346.

[Chaudhuri, 03] Chaudhuri, S., Ganjam, K., Ganti, V., Motwani, R. Robust and efficient fuzzy match for online data cleaning. In ACM SIGMOD Conference (pp. 313–324) 2003.

[Di Giacomo, 04] Di Giacomo, G. D., Lembo, D., Lenzerini, M., Rosati, R. Tackling inconsistencies in data integration through source preferences. In ACM IQIS Workshop (pp. 27–34) 2004.

[Doan, 05] Doan A., Halevy A. Y. Semantic Integration Research in the Database Community: A Brief Survey. AI Magazine 26(1): 83-94 (2005).

[Du, 04] Du F., Amer-Yahia S., Freire J. ShreX: Managing XML Documents in Relational Databases. VLDB 2004: 1297- 1300, 2004.

[Galindo-Legaria, 94] Galindo-Legaria, C. A. Outerjoins as disjunctions. In ACMSIGMOD Conference (pp. 348–358), 1994.

[Genesereth, 97] Genesereth M. R., Keller A. M., Duschka O., Infomaster: An Information Integration System, in proceedings of 1997 ACM SIGMOD Conference, May 1997.

[Greco, 03] Greco, G., Greco, S., Zumpano, E. A logical framework for querying and repairing inconsistent databases. IEEE Trans. Knowl. Data Eng., 15 (6), 1389–1408 2003.

[Halevy, 01] Halevy, A. Y. Answering queries using views: A survey VLDB Journal, 10(4), 270–294, 2001.

[Hammer, 05] Hammer J., Stonebraker M., Topsakal O. THALIA: Test Harness for the Assessment of Legacy Information Integration Approaches. ICDE 2005: 485-486.

[Knoblock, 00] Knoblock C. A. Ambite J.L. Flexible and scalable cost-based query planning in mediators: A transformational approach. Artificial Intelligence, 118(1-2):115-161, 2000.

[Levy, 98] Levy A., The Information Manifold Approach to Data Integration, IEEE Intelligent Systems, 1312-16, 1998.

[Li, 98] Li C., Yerneni R., Vassalos V., Garcia-Molina H., Papakonstantinou Y., Ullman J., Valiveti M. Capability Based Mediation in TSIMMIS. SIGMOD 98, Seattle, June 1998.

[Mecca, 2012] Mecca G., Papotti P., Raunich S., Santoro D. What is the IQ of your data transformation system? CIKM 2012: 872-881

[Lin, 98] Lin, J., Mendelzon, A. O. Merging databases under constraints. Int. J. Cooperative Inf. Syst., 7 (1), 55–76, 1998.

[Miller, 01] Miller R. J., Hernandez M. A., Haas L. M., Yan L., Ho C. T. H., Popa L., Fagin R., The Clio project: managing heterogeneity, ACM SIGMOD Record 30, 1 (March 2001), pp. 78-83.

[Naumann, 02] Naumann, F., Haussler, M. Declarative data merging with conflict resolution. In MIT-IQ Conference (pp. 212–224) 2002.

[Rajaraman, 96] Rajaraman, A., Ullman, J. D. Integrating information by outerjoins and full disjunctions. In ACM-PODS Conference (pp. 238–248) 1996.

[Tejada, 01] Tejada, S., Knoblock, C. A., Minton, S. Learning object identification rules for information integration. Inf. Syst., 26 (8), 607–633, 2001.

[Ullman, 97] Ullman, J. D. Information integration using logical views. In ICDT Conference, (pp. 19–40), 1997.

[Wiederhold, 93] Wiederhold, G. Intelligent integration of information. In ACM SIGMOD Conference (1993) (pp. 434–437).

[Yu, 06] Yu B., Liu L., Ooi B. C., Tan K. L. Keyword Join: Realizing Keyword Search for Information Integration, in MIT press, Computer Science (CS), 2006, http://hdl.handle.net/1721.1/30263

Appendix A

Query 1

Mapping between Instructor attribute of Georgia Tech University and Lecturer attribute of Carnegie Mellon University.

No mapping refinement.

Query 2

Mapping between Time attribute of Carnegie Mellon University and Times attribute of University of Massachusetts.

Mapping refinement:

```
MTF[Time][umb.Course] = TIME12-24(Times, 1, 12) + SUBSTRING(Times, 6, 1) +
                        TIME12-24(Times, 7, 12)
```

Query 3

Mapping between CourseName attribute of University of Maryland and Title attribute of Brown University.

Mapping refinement:

```
MTF[Title][brown.Course] = SUBSTRING(Title FROM POSITION('/"' IN Title) + 3
                           FOR POSITION ('hr.' IN SUBSTRING (Title FROM
                           POSITION ('/"' IN Title) + 3 FOR 100)) - 1)
```

Query 4

Mapping between Units attribute of Carnegie Mellon University and Umfang attribute of ETH Zurich.

Mapping refinement:

```
MTF[Unit][ethz.Unterricht] = CAST(SUBSTRING(Umfang, POSITION('V' IN
                             Umfang) - 1, 1) AS int) + CAST(SUBSTRING(Umfang,
                             POSITION('U' IN Umfang) - 1, 1) AS int) + 1
```

Query 5

Mapping between CourseName attribute of University of Maryland and Title attribute of ETH Zurich.

No mapping refinement.

Query 6

Mapping between title attribute of University of Toronto and no attribute of Carnegie Mellon University.

No mapping refinement.

Query 7

Mapping between prerequisite attribute of University of Michigan and description attribute in Arizona State University.

Mapping refinement:

```
MTF[prerequisite][asu.Course] = CASE POSITION('%Prerequisite%' IN
                                                    Description)
                                WHEN 0 THEN 'None'
                                ELSE RIGHT(Description,
                                      CHAR_LENGTH(Description) -
                                   POSITION('%Prerequisite%' IN Description) + 1)
                                END
```

Query 8

Mapping between ‘Course restricted’ attribute of Georgia Tech University and no attribute of ETH Zurich.

No mapping refinement.

Query 9

Mapping between room attribute of Brown University and time attribute of University of Maryland.

Mapping refinement:

```
MTF[Room][umd.section] = SUBSTRING(Time FROM POSITION('%(%' IN Time)
                                   FOR 30)
```

Query 10

Mapping between lecturer attribute of Carnegie Mellon University and title attribute of University of Maryland.

Mapping refinement:

```
MTF[Title][umd.section] = SUBSTRING(Title FROM
                          POSITION('%.%' IN Title) FOR POSITION('%)%' IN
                          Title) + 2) FOR POSITION('%.%' IN Title) + 1)
```

Query 11

Mapping between lecturer attribute of Carnegie Mellon University and attributes named Fall2003, Winter2004 and Spring2004 of University of California, San Diego.

Mapping refinement:

```
MTF[Lecturer][ucsd.Course] =  CASE WHEN (CHAR_LENGTH (Fall2003) >
                              CHAR_LENGTH (Winter2004) AND
                         CHAR_LENGTH (Fall2003) > C HAR_LENGTH (Spring2004))
                          THEN Fall2003
                          WHEN (CHAR_LENGTH (Winter2004) >
                           CHAR_LENGTH (Fall2003) AND CHAR_LENGTH
                          (Winter2004) > CHAR_LENGTH (Spring2004))
                         THEN Winter2004
                        WHEN (CHAR_LENGTH (Spring2004) >
                        CHAR_LENGTH (Fall2003) AND
```

```
CHAR_LENGTH (Spring2004) >
CHAR_LENGTH (Winter2004)) THEN Spring2004
END
```

Query 12

Mapping between CourseTitle, Day, Time attribute of Carnegie Mellon University and Title attribute of Brown University.

Mapping refinement:

```
MTF[Title][brown.Course] =
    SUBSTRING(Title FROM POSITION('/"' IN Title) + 3 FOR
    POSITION ('hr.' IN SUBSTRING(Title FROM
    POSITION('/"' IN Title) + 3 FOR 100)) - 1)
MTF[Day][brown.Course] =
    SUBSTRING(Title FROM POSITION('hr.' IN Title) + 4 FOR
    POSITION(' ' IN SUBSTRING(Title
    FROM POSITION('hr.' IN Title) + 4  FOR10)))
MTF[Time][brown.Course] =
    SUBSTRING(Title IN POSITION(' ' FROM SUBSTRING(Title
    FROM POSITION('hr.' IN Title) + 4 FOR 10)) +
    POSITION('hr.' IN Title) + 4 FOR 15)
```