Virtual Integration of Existing Web Databases for the Genotypic Selection of Cereal Cultivars*

Sonia Bergamaschi and Antonio Sala

Dipartimento di Ingegneria dell'Informazione
Universitá di Modena e Reggio Emilia
bergamaschi.sonia@unimore.it, sala.antonio@unimore.it

Abstract. The paper presents the development of a virtual database for the genotypic selection of cereal cultivars starting from phenotypic traits.

The database is realized by integrating two existing web databases, Gramene [1] and Graingenes [2], and a pre-existing data source developed by the Agrarian Faculty of the University of Modena and Reggio Emilia. The integration process gives rise to a virtual integrated view of the underlying sources. This integration is obtained using the MOMIS system (Mediator envirOnment for Multiple Information Sources), a framework developed by the Database Group of the University of Modena and Reggio Emilia (www.dbgroup.unimo.it). MOMIS performs information extraction and integration from both structured and semistructured data sources. Information integration is performed in a semi-automatic way, by exploiting the knowledge in a Common Thesaurus (defined by the framework) and the descriptions of source schemas with a combination of clustering and Description Logics techniques. Momis allows querying information in a transparent mode for the user regardless of the specific languages of the sources. The result obtained by applying MOMIS to Gramene and Graingenes web databases is a queriable virtual view that integrates the two sources and allow performing genotypic selection of cultivars of barley, wheat and rice based on phenotypic traits, regardless of the specific languages of the web databases. The project is conducted in collaboration with the Agrarian Faculty of the University of Modena and Reggio Emilia and funded by the Regional Government of Emilia Romagna.

1 Introduction

In the last few years the progress in the field of the molecular biology gave rise to an exponential growth of data available to researchers. The great problem they are now facing is how to have access to this great amount of data in order to exploit them for their research activity. Many resources are available on Web

[*] This work is supported by the Italian Ministry of Research and University through the PRIN2004 "WISDOM" project.

[1] http://www.gramene.org/

[2] http://wheat.pw.usda.gov/

R. Meersman, Z. Tari et al. (Eds.): OTM 2006, LNCS 4275, pp. 909–926, 2006.

databases, but usually these informations reside in different, heterogeneous and sometimes numerous sources. Another problem is that these databases usually present different interfaces and structure of the information and are thus difficult to be queried by biology researchers. For these reasons a maybe simple information search can take long time and eventually fails because of the number of different data sources to be accessed.

What is needed to solve these problems is the definition of methods for:

1. Extracting and fusing the information coming from different (and heterogeneous) information sources (e.g. web sites and web databases)
2. Presenting the information according to a unique interface.

The paper presents the MOMIS system (Mediator envirOnment for Multiple Information Sources) [2], a mediator framework to perform information extraction and integration from heterogeneous distributed data sources and query management facilities to transparently support query posed to the integrated data sources. The framework consists of a language and two main components:

- The ODL_{I^3} language is an object-oriented language, with an underlying Description Logic; it is derived from the standard ODL-ODMG [9].
- The Ontology Builder: sources integration is performed in a semi-automatic way, by exploiting the knowledge in a Common Thesaurus (defined by the framework) and ODL_{I^3} descriptions of source schemas with a combination of clustering techniques and Description Logics. This integration process gives rise to a virtual integrated view of the underlying sources (the Global Schema, GVV) for which mapping rules and integrity constraints are specified to handle heterogeneity.
- The MOMIS Query Manager is the coordinated set of functions which take an incoming query, decompose the query according to the mapping of the GVV onto the local data sources relevant for the query, send the subqueries to these data sources, collect their answers, perform any residual filtering as necessary, and finally deliver the answer to the requesting user.

The MOMIS system is based on a conventional wrapper/mediator architecture, and provides methods and open tools for data management in Internet-based information systems (Fig.1). The MOMIS development begun as a joint collaboration between the University of Modena and Reggio Emilia and University of Milano and Brescia, within the INTERDATA national research project. The research activities continued within the SEWASIE European research project (IST-2001-34825) (www.sewasie.org).

MOMIS can analyze the contents of the source web databases and build a Global View. This Global View (GVV) is "Virtual" i.e. it is not materialized: data reside on the "local" sources and the View is an entry point for data retrieving. Consequently, only the changes in the information source structures have a side effect on a built GVV. Data changes do not have any influence on it. Moreover the existence of semantic tags in the sources, or the semantic annotations with respect to a lexical ontology (e.g. WordNet) are exploited to build

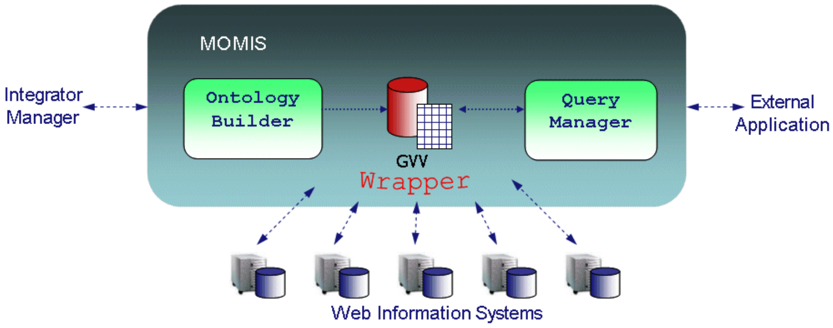


Fig. 1. The Momis Architecture

the GVV which can be seen as a domain ontology of the involved sources. The GVV, can be exported in RDFS and OWL thus guaranteeing interoperability with other external applications/ontologies or external users.

In this paper we present the use of the MOMIS system to perform intelligent data integration of existing databases to create a Virtual View for the genotypic selection of cereal cultivars based on their phenotype. This GVV has been realized as a part of the CEREALAB project conducted by the Agrarian faculty of the University of Modena and Reggio Emilia in collaboration with the Database Group of the University of Modena and Reggio Emilia and funded by the Regional Government of Emilia Romagna. The aim of the CEREALAB project is to make available to the cereal breeders of the Emilia Romagna region the knowledge learnt in the research activity by the Universities, in particular to provide them with a tool to perform genotypic selection of cereal cultivars from phenotypic traits. Thus, the idea was to create the CEREALAB database as an integration of two existing web databases, Gramene (concerning maize and rice) and Graingenes (concerning wheat and barley), with another data source storing the information achieved by the research group of the project. In this way the CEREALAB database performs both the tasks of (1)providing a valid support for the research activity, suppling information from the existing databases, and (2)being a knowledge base to store the data obtained by researchers.

The outline of the paper is the following: the first section describes the use of MOMIS to create the CEREALAB GVV: in particular Sect.2 describes the sources integration approach, Sect.3 sketches out the querying process presenting some examples. Finally Sect.4 compares MOMIS with other system and gives conclusions.

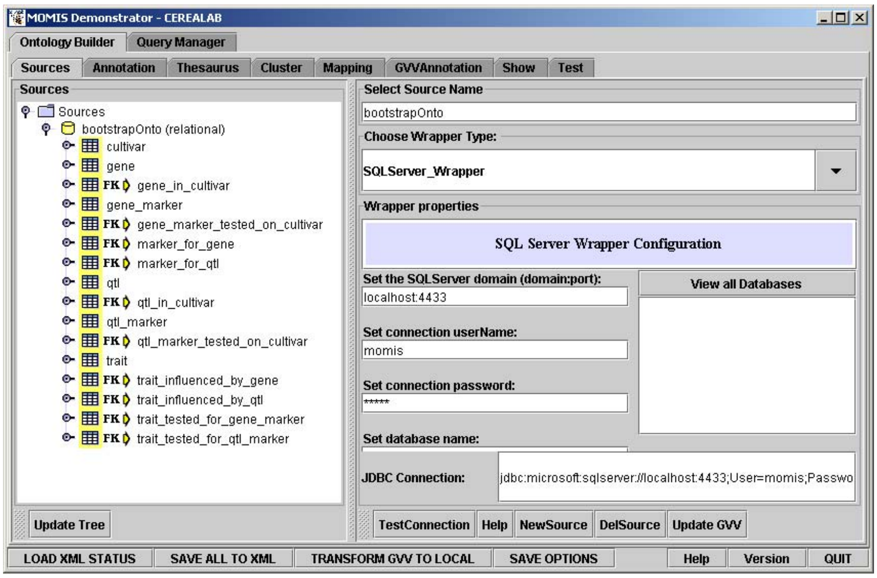
1.1 The CEREALAB Domain

The main entities of the CEREALAB domain are:

- Cultivar, which identify an assemblage of plants that has been selected for a particular attribute or combination of attributes and is clearly distinct, uniform and stable in its characteristics.

- Trait, an inherited feature of a plant.
- Gene, the unit of heredity in living organisms, which controls the physical development of the organism. An allele is any one of a number of viable DNA codings of the same gene occupying a given locus (position) on a chromosome.
- QTL, quantitative trait locus, a region of DNA that is associated with a particular trait. Though not necessarily genes themselves, QTLs are stretches of DNA that are closely linked to the genes that underlie the trait in question.
- Marker, a known DNA sequence (e. g. a gene or part of gene) that can be identified by a simple assay, associated with a certain phenotype. A genetic marker may be a short DNA sequence, such as a sequence surrounding a single base-pair change, or long one, like microsatellites.

Bearing in mind these main entities, we developed a kernel GVV (i.e. a bootstrap ontology) of the CEREALAB database to be used as a reference for the integration process of the available data sources. This ontology was created with MOMIS as a relational data source (see Fig.2), re-engineering the set of files used by the CEREALAB research group.



Select Source Name: bootstrapOnto

Choose Wrapper Type: SQLServer_Wrapper

Set the SQLServer domain (domain:port): localhost:4433

Set connection userName: momis

Set connection password: *****

JDBC Connection: jdbc:microsoft:sqlserver://localhost:4433;User=momis;Passwo

Fig. 2. The bootstrap Ontology for the CEREALAB database

2 The MOMIS Integration Methodology

In this section, we describe the information integration process for building the GVV. The process, shown in Fig.3, gives rise to Global Virtual View of several specific data sources. The GVV-generation process in our case has been modified compared with the usual MOMIS approach [5] as we have a pre-existing bootstrap ontology which has to be enriched with other data source:

1. Insertion of a pre-existing ontology as local source. This data source stores the information achieved research group of the CEREALAB project.
2. Local source schemata extraction. Wrapper generates schemas for the involved sources and translates them into the common language ODL_{I^3} [2]
3. Local source annotation with WordNet. The integration designer chooses a meaning for each element of a local source schema, according to the WordNet lexical ontology (`http://www.cogsci.princeton.edu/~wn`). This phase may be executed semi-automatically: a tool supports the integration designer in the choice by proposing a WordNet concept for each source element.
4. Common thesaurus generation. Starting from the annotated local schema, MOMIS constructs a set of relationships describing inter and intraschema knowledge about classes and attributes of the source schemata.
5. GVV generation. The MOMIS methodology, applied to the common thesaurus and the local schemata descriptions, generates a global schema and sets of mappings with local schemata
6. Mapping refinement. The system automatically generates a Mapping Table for each global class of the GVV which can be extended by the designer.

The above methodology is described in the following sections. The Ontology Builder Tool supports the integration designer in all the GVV generation process phases to realize our virtual view for the genotypic selection of cereal cultivars.

2.1 The ODL_{I^3} Language

As a common data model for integrating a given set of local information sources, MOMIS uses an object-oriented language called ODL_{I^3}. ODL_{I^3} extends ODL with the following relationships expressing intra- and inter-schema knowledge for the source schemas: SYN (synonym of), BT (broader terms), NT (narrower terms) and RT (related terms). By means of ODL_{I^3}, only one language is exploited to describe both the sources (the input of the synthesis process) and the GVV (the result of the process). The translation of ODL_{I^3} descriptions into one of the Semantic Web standards such as RDF, DAML+OIL, OWL is a straightforward process. In fact, from a general perspective an ODL_{I^3} concept corresponds to a Class of the Semantic Web standard, and ODL_{I^3} relationships are translated into properties. Figure 3 shows the global schema generation, where local schemas are annotated according to the lexical ontology WordNet, the Common Thesaurus generation, and finally the GVV global classes. In particular, these ones are connected by means of mapping tables to the local schemas and are (semi-automatically) annotated according to WordNet. The designer can refine the mappings supported by the Ontology Builder.

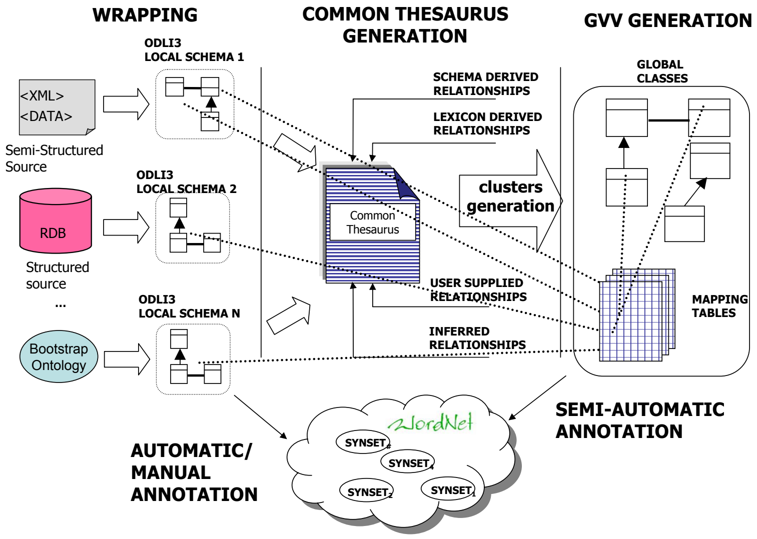


Fig. 3. Integration Process Overview

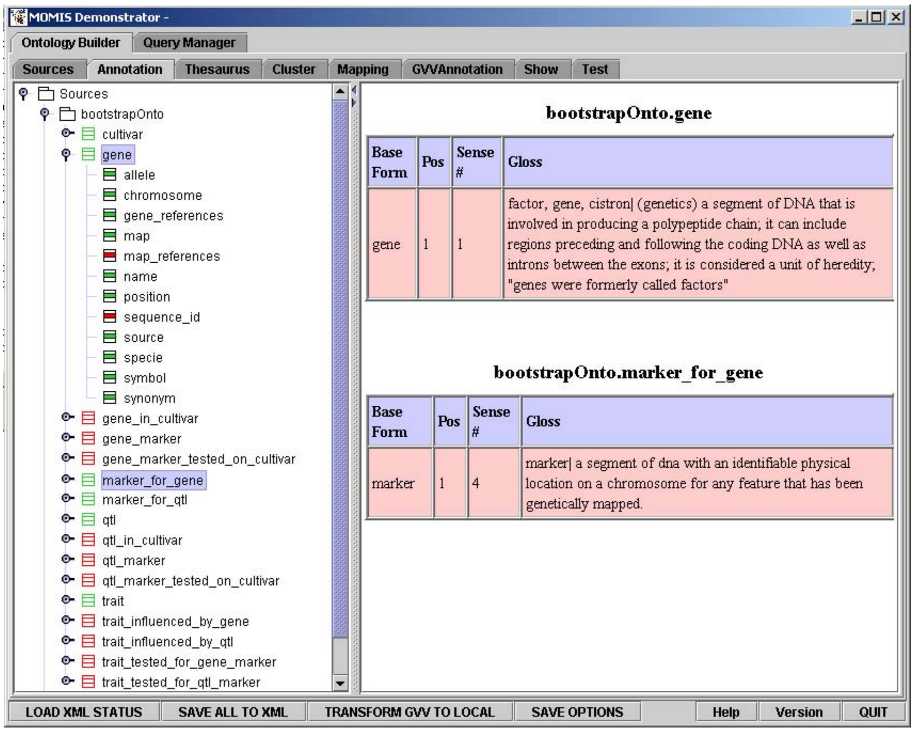
2.2 Insertion of Pre-existing Ontology / Wrapping: Extracting Data Structure for Sources

The first phase of the integration process is the choice of the data sources and their translation into ODL_{I^3} format. In our case a pre-defined ontology, that we call bootstrap ontology, existed and could be enriched by other data sources. Gramene and Graingenes have been chosen as further local sources, as they are the most significant for the domain. The translation process is performed by the MOMIS wrappers, which logically converts the source data structure into the ODL_{I^3} model. The wrapper architecture and interfaces are crucial, because wrappers are the focal point for managing the diversity of data sources. For conventional structured information sources (e.g. relational databases), schema description is always available and can be directly translated. In our case wrapping was easy as it is possible to download the two underlying relational databases of Gramene and Graingenes, creating two local sources. In this way we were able to use the MOMIS SqlServer Wrapper to manage both the sources.

After this step we have three local sources: the bootstrap ontology (CEREALAB), and the Gramene and Graingenes sources.

2.3 Semi-automatic Annotation of a Local Source with WordNet

The goal of the annotation phase is to assign a name and a set of meanings belonging to the WordNet [14] lexical system to each local class and attribute of the local schemata. For each element of a local schema the system automatically suggests a word form corresponding to the given term (if it exists): thus the designer may confirm or change the word form or meaning of each element.



bootstrapOnto.gene

Base Form	Pos	Sense #	Gloss
gene	1	1	factor, gene, cistron| (genetics) a segment of DNA that is involved in producing a polypeptide chain; it can include regions preceding and following the coding DNA as well as introns between the exons; it is considered a unit of heredity; "genes were formerly called factors"

bootstrapOnto.marker_for_gene

Base Form	Pos	Sense #	Gloss
marker	1	4	marker| a segment of dna with an identifiable physical location on a chromosome for any feature that has been genetically mapped.

Fig. 4. The manual/automatic annotation for marker-for-gene/gene

As in our case the annotation is related to a very specific domain, WordNet lacks many of the terms involved. MOMIS provides the user with a WordNet Editor [3] to extend WordNet by adding new terms and synsets to the native elements of WordNet. Guided by the CEREALAB research group, we widely extended WordNet to face this new domain according to an existing technical glossary provided by the Gramene website. As an example, Fig.4 shows the automatic annotation for the "gene" attribute and the manual annotation, i.e. the definition we provided, for the "marker-for-gene" attribute. The annotation phase of the bootstrap ontology took quite a long time, but it resulted easier for the other two sources thanks to the extension provided to WordNet (as many of the terms involved in these two sources appear also in the bootstrap ontology) and to the capability of the WordNet Editor to cache the annoted terms and synsets. The advantage is that this extension step has to be performed just the first time a domain is handled.

2.4 Common Thesaurus Generation

Starting from the annotated local schemata, MOMIS constructs a Common Thesaurus describing intra and inter-schema knowledge in the form of SYN(synonyms), BT/NT(broader terms/narrower terms), and RT(meronymy/holonymy) relationships.

The Common Thesaurus is constructed through an incremental process in which the following relationships are added:

- schema-derived relationships: relationships holding at intra-schema level are automatically extracted by analyzing each schema separately. For example, MOMIS extracts intraschema RT relationships from foreign keys in relational source schemas. When a foreign key is also a primary key, in both the original and referenced relation, MOMIS extracts BT and NT relationships, which are derived from inheritance relationships in object-oriented schemas.
- lexicon-derived relationships: we exploit the annotation phase in order to translate relationships holding at the lexical level into relationships to be added to the Common Thesaurus.
- designer-supplied relationships: new relationships can be supplied directly by the designer, to capture specific domain knowledge.
- inferred relationships: Description Logics (DL) techniques of ODB-Tools [4],[6] are exploited to infer new relationships, by means of subsumption computation applied to a "virtual schema" obtained by interpreting BT/NT as subclass relationships and RT as domain attributes.

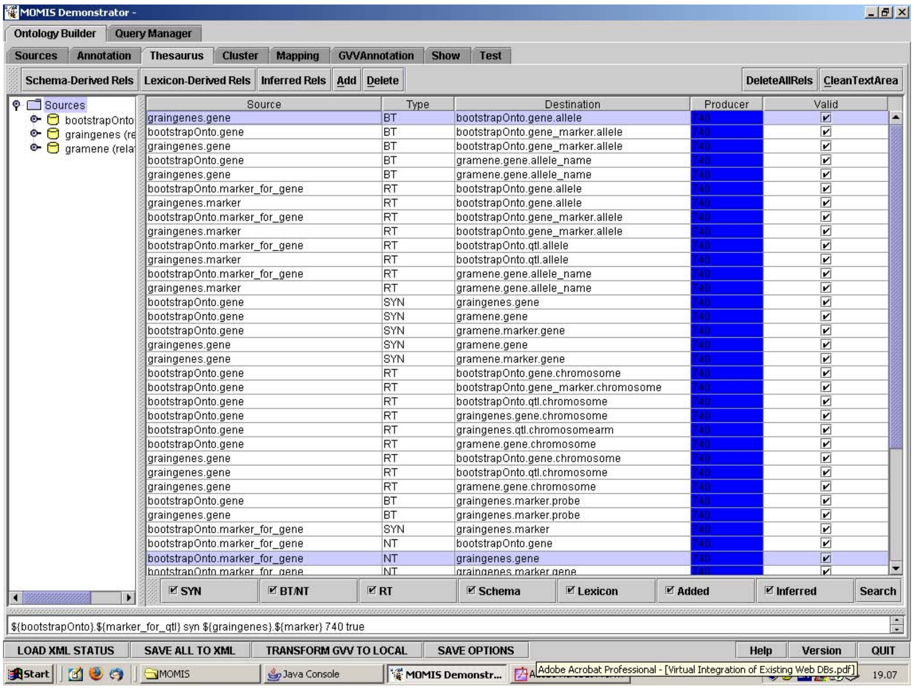
A detailed presentation of the methodology can be found in [2], [7].

Figure 5 shows some relationships automatically extracted by MOMIS for the `gene` classes and attributes. In our case, the relationships holding at lexical level are widely exploited to discover semantic relationships between local classes. For example, MOMIS sematically promotes the RT relationship between `gene` and `allele` (see Fig.4) into a BT relationship (see Fig.5, line 6). For the class `marker-for-gene` there is no evident schema relationship, but sematically it is identified as a NT of `gene` (see Fig.5, last line).

2.5 Global Virtual View (GVV) Generation

The MOMIS methodology allows us to identify similar ODL_{I^3} classes, that is, classes that describe the same or semantically related concepts in different sources, and mappings to connect the global attributes of each global class with the local sources' attributes. To this end, affinity coefficients are evaluated for all possible pairs of ODL_{I^3} classes, based on the relationships in the Common Thesaurus properly strengthened. Affinity coefficients determine the degree of matching of two classes based on their names (Name Affinity coefficient) and their attributes (Structural Affinity coefficient) and are fused into the Global Affinity coefficient, calculated by means of the linear combination of the two coefficients. Global affinity coefficients are then used by a hierarchical clustering algorithm, to include ODL_{I^3} classes in clusters according to their degree of affinity. The designer may interactively refine and complete the proposed integration results; in particular, the mappings which has been automatically created by the system can be fine tuned as discussed in Sect.2.6.

Being so specific the domain we are facing, the annotation phase gave rise to a large number of BT/NT relationships. This, together with a very different structure of the two sources, Gramene and Graingenes, from our bootstrap



Source	Type	Destination	Producer	Valid
graingenes.gene	BT	bootstrapOnto.gene.allele	740	☑
bootstrapOnto.gene	BT	bootstrapOnto.gene_marker.allele	740	☑
graingenes.gene	BT	bootstrapOnto.gene_marker.allele	740	☑
bootstrapOnto.gene	BT	gramene.gene.allele_name	740	☑
graingenes.gene	BT	gramene.gene.allele_name	740	☑
bootstrapOnto.marker_for_gene	RT	bootstrapOnto.gene.allele	740	☑
graingenes.marker	RT	bootstrapOnto.gene.allele	740	☑
bootstrapOnto.marker_for_gene	RT	bootstrapOnto.gene_marker.allele	740	☑
graingenes.marker	RT	bootstrapOnto.gene_marker.allele	740	☑
bootstrapOnto.marker_for_gene	RT	bootstrapOnto.qtl.allele	740	☑
graingenes.marker	RT	bootstrapOnto.qtl.allele	740	☑
bootstrapOnto.marker_for_gene	RT	gramene.gene.allele_name	740	☑
graingenes.marker	RT	gramene.gene.allele_name	740	☑
bootstrapOnto.gene	SYN	graingenes.gene	740	☑
bootstrapOnto.gene	SYN	gramene.gene	740	☑
bootstrapOnto.gene	SYN	gramene.marker.gene	740	☑
graingenes.gene	SYN	gramene.gene	740	☑
graingenes.gene	SYN	gramene.marker.gene	740	☑
bootstrapOnto.gene	RT	bootstrapOnto.gene.chromosome	740	☑
bootstrapOnto.gene	RT	bootstrapOnto.gene_marker.chromosome	740	☑
bootstrapOnto.gene	RT	bootstrapOnto.qtl.chromosome	740	☑
bootstrapOnto.gene	RT	graingenes.gene.chromosome	740	☑
bootstrapOnto.gene	RT	graingenes.qtl.chromosomearm	740	☑
bootstrapOnto.gene	RT	gramene.gene.chromosome	740	☑
graingenes.gene	RT	bootstrapOnto.gene.chromosome	740	☑
graingenes.gene	RT	bootstrapOnto.qtl.chromosome	740	☑
graingenes.gene	RT	gramene.gene.chromosome	740	☑
bootstrapOnto.gene	BT	graingenes.marker.probe	740	☑
graingenes.gene	BT	graingenes.marker.probe	740	☑
bootstrapOnto.marker_for_gene	SYN	graingenes.marker	740	☑
bootstrapOnto.marker_for_gene	NT	bootstrapOnto.gene	740	☑
bootstrapOnto.marker_for_gene	NT	graingenes.gene	740	☑
bootstrapOnto.marker_for_gene	NT	graingenes.marker.gene	740	☑

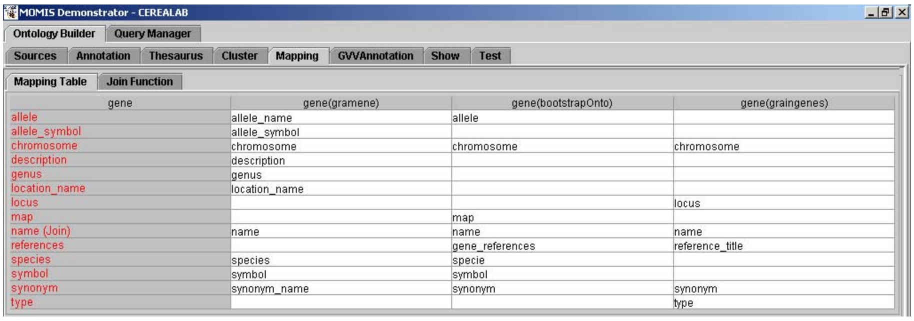
${bootstrapOnto}.${marker_for_qtl} syn ${graingenes}.${marker} 740 true

Fig. 5. The Common Thesaurus for the gene and marker-for-gene classes

Ontology, forced us to give less weight than usual to the BT/NT relationship when calculating the Name Affinity and the Structural Affinity coefficients. In particular, instead of the default weight to calculate the coefficients, SYN=1, BT/NT=0.8, we tuned the weights in order to consider SYN relationships as the most relevant. In this way the system gave rise to a set of significant Global Classes that reflected the entities we defined in Sect.1.1. No particular issues have been encountered during the generation of the GVV as all the data types appearing in the different sources are homogeneous for the same real world objects. These Global Classes have then been verified by the agrarian researchers to validate the Global Virtual View obtained and to indicate possible refinements. The main issues are identifying the Join Conditions and in few cases defining the Resolution Functions, which will be presented in Sect.2.6.

2.6 Mapping Refinement

The system automatically generates a Mapping Table (MT) for each global class C of a 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. Figure 6 shows the MT of the gene Global Class.



gene	gene(gramene)	gene(bootstrapOnto)	gene(graingenes)
allele	allele_name	allele	
allele_symbol	allele_symbol		
chromosome	chromosome	chromosome	chromosome
description	description		
genus	genus		
location_name	location_name		
locus			locus
map		map	
name (Join)	name	name	name
references		gene_references	reference_title
species	species	specie	
symbol	symbol	symbol	
synonym	synonym_name	synonym	synonym
type			type

Fig. 6. The Mapping Table for the gene Global Class

More formally, we define an Integration System IS = (GVV, N, M) as constituted by:

- A GVV, which is a schema expressed in ODL_{I^3}.
- A set N of local sources; each local source has a schema also expressed in ODL_{I^3}.
- A set M of GAV mapping assertions between the GVV and N, where each assertion associates to an element g in GVV a query Q_N over the schemas of a set of local sources in N.

More precisely, for each global class C of the GVV we define:

- a (possibly empty) set of local classes, denoted by $L(C)$, belonging to the local sources in N
- a conjunctive query Q_N over $L(C)$.

Intuitively, the GVV is the intensional representation of the information provided by the Integration System, whereas the mapping assertions specify how such an intensional representation relates to the local sources managed by the Integration System. The query Q_N associated to a global class C is implicitly defined by the designer starting from the MT of C. The designer can extend the MT by adding:

- Data Conversion Functions from local to global attributes
- Join Conditions among pairs of local classes belonging to C
- Resolution Functions for global attributes to solve data conflicts of local attribute values.

On the basis of the resulting MT the system automatically generates a query Q_N associated to C, by extending the Full Disjunction operator [11], which is explained in the following.

Data Conversion Functions. The Ontology Designer can define, for each not null element $MT[GA][L]$, a Data Conversion 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 must be executable/supported by the class L local source. For example, for relational sources, $MTF[GA][L]$ is an SQL value expression. $T(L)$ denotes L transformed by the Data Conversion Functions.

Join Conditions. Merging data from different sources requires different instantiations of the same real world object to be identified; this process is called object identification [16], [18], [1], [10].

To identify instances of the same object and fuse them we introduce Join Conditions among pairs of local classes belonging to the same global class. Given two local classes L_1 and L_2 belonging to C, a Join Condition between L_1 and L_2, denoted with $JC(L_1, L_2)$, is an expression over $L_1.A_i$ and $L_2.A_j$ where A_i (A_j) are global attributes with a not null mapping in L_1 (L_2). As an example, the join condition for the `gene` Global Class is defined as follow:

```
((graingenes.gene.name) = (bootstrapOnto.gene.name)) AND
(((gramene.gene.name) = (bootstrapOnto.gene.name))
OR ((gramene.gene.name) = (graingenes.gene.name)))
```

Resolution Functions. In MOMIS the approach proposed in [16] 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; in this way we can define what value shall appear in the result. Our system provides some standard kinds of resolution functions (Random, Aggregation, Coalescence and others). In our domain we used the followings:

1. *Precedence function*: experimental results obtained by the CEREALAB research group regard mainly italian cultivars. These data could be different from data from the two existing sources, especially referring to phenotypic information since the Gramene and Graingenes are american databases. For this reason a precedence function has been used, to give priority to the CEREALAB informations as they are related to italian cultivars.
2. *All Values*: considering the integration viewpoint, the aim is to preserve all the information coming from the sources. For example, a "reference" attribute is often present for many entities to provide bibliographic references. Sometimes each source can cite different relevant references for the instance in exam. To let the user get all the data provided by the local sources as a result, we used the All Values function provided by MOMIS to return all the references present in the different sources.

Full Disjunction. Q_N is defined in such a way that it contains a unique tuple resulting from the merge of all the different tuples representing the same real world object. This problem is related to that of computing the natural outer-join of many relations in a way that preserves all possible connections among facts [17]. Such a computation has been termed as Full Disjunction (FD) by Galindo Legaria [11]. In our context: given a global class C composed of L_1,L_2, ..., L_n, we consider

$$FD(T(L_1), T(L_2), \ldots, T(L_n))$$

computed on the basis of the Join Conditions. With more than 2 local classes, the computation of FD is performed as follows. We assume that: (1) each L contains a key, (2) all the join conditions are on key attributes, and (3) all the join attributes are mapped into the same set of global attribute, say K. Then, it can be proved that: (1) K is a key of C, and (2) FD can be computed by means of the following expression:

$$(T(L_1) \text{ full join } T(L_2) \text{ on } JC(L_1, L_2)) \text{ full join } T(L_3)$$

$$\text{on } (JC(L_1, L_3) \text{ OR } JC(L_2, L_3)) \ldots \text{ full join } T(L_n) \text{ on } (JC(L_1, L_n)$$

$$\text{OR } JC(L_2, L_n) \text{ OR} \ldots \text{OR } JC(L_{n-1}, L_n))$$

Finally, Q_N is obtained by applying Resolution Functions to the attributes resulting from the above expression: for a global attribute GA we apply the related Resolution Function to $T(L_1).GA$, $T(L_2).GA$, $\ldots$, $T(L_k).GA$. As an example, Q_N fr the `gene` Global Class is:

```
bootstrapOnto.gene full outer join graingenes.gene
on (((graingenes.gene.name) = (bootstrapOnto.gene.name)))
full outer join gramene.gene
on (((gramene.gene.name) = (bootstrapOnto.gene.name))
OR ((gramene.gene.name) = (graingenes.gene.name)))
```

3 The MOMIS Query Manager

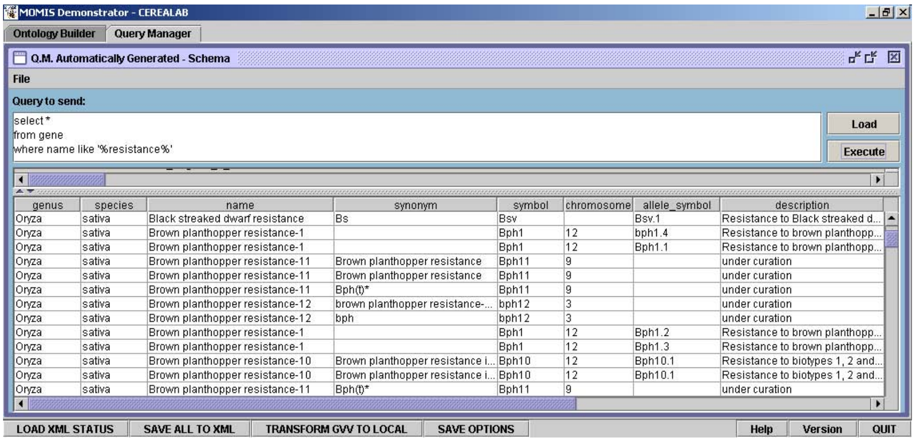
The MOMIS Query Manager is the coordinated set of functions which takes an incoming query (say global query), defines a decomposition of the query according to the mapping of the GVV onto the local data sources, sends the subqueries to these data sources, collects their answers, fuse them (performing any residual filtering as necessary), and finally delivers the answer. Query processing consists of the following steps:

1. Query rewriting: to rewrite a global query as an equivalent set of queries expressed on the local sources (local queries)
2. Local queries execution: the local queries are sent and executed at local sources
3. Fusion and Reconciliation: the local answers are fused into the global answer.

Let us introduce a simple query in order to show the query processing steps:

```
select * from gene where name like '\%resistance\%'
```

The query retrieves all the genes that contain the word "resistance" in their name, i.e. the genes that express a kind of resistance. With Momis, this query allows the user to transparently retrieve information from Gramene and Graingenes with a single query (see Fig.7).



Query to send:

```
select *
from gene
where name like '%resistance%'
```

genus	species	name	synonym	symbol	chromosome	allele_symbol	description
Oryza	sativa	Black streaked dwarf resistance	Bs	Bsv		Bsv.1	Resistance to Black streaked d...
Oryza	sativa	Brown planthopper resistance-1		Bph1	12	bph1.4	Resistance to brown planthopp...
Oryza	sativa	Brown planthopper resistance-1		Bph1	12	Bph1.1	Resistance to brown planthopp...
Oryza	sativa	Brown planthopper resistance-11	Brown planthopper resistance	Bph11	9		under curation
Oryza	sativa	Brown planthopper resistance-11	Brown planthopper resistance	Bph11	9		under curation
Oryza	sativa	Brown planthopper resistance-11	Bph(t)*	Bph11	9		under curation
Oryza	sativa	Brown planthopper resistance-12	brown planthopper resistance-...	bph12	3		under curation
Oryza	sativa	Brown planthopper resistance-12	bph	bph12	3		under curation
Oryza	sativa	Brown planthopper resistance-1		Bph1	12	Bph1.2	Resistance to brown planthopp...
Oryza	sativa	Brown planthopper resistance-1		Bph1	12	Bph1.3	Resistance to brown planthopp...
Oryza	sativa	Brown planthopper resistance-10	Brown planthopper resistance i...	Bph10	12	Bph10.1	Resistance to biotypes 1, 2 and...
Oryza	sativa	Brown planthopper resistance-10	Brown planthopper resistance i...	Bph10	12	Bph10.1	Resistance to biotypes 1, 2 and...
Oryza	sativa	Brown planthopper resistance-11	Bph(t)*	Bph11	9		under curation

Fig. 7. A query on the CEREALAB GVV

3.1 Query Rewriting

MOMIS uses a global-as-view (GAV) approach [12] to model the mapping among the GVV and the local schemata. Then the global query is rewritten by means of unfolding, that is, by expanding each atom of the global query according to its definition in the mapping. A detailed description can be found in [8].

We consider a Global Query Q over a Global Class G:

$$Q = \text{select} < Q_{select-list} > \text{from G where} < Q_{condition} >$$

where $Q_{condition}$ is a Boolean expression of positive atomic constraints:

$$(GA\ op\ value)\text{or}(GA_1\ op\ GA_2)$$

where GA_1 and GA_2 are attributes of the Global Class G.

The query rewriting process is composed of the following steps:

1. Atomic constraint mapping
   In this step, each atomic constraint of a query Q is rewritten into one that can be supported by the local class. The atomic constraint mapping is performed on the basis of mapping functions defined in the Mapping Table.
2. Residual Constraints computation: Intuitively, residual constraints are the constraints of the global query that are not mapped in all local queries.
3. Local select-list computation: The select-list of a local query is a set of attributes, including the global query attributes, the join attributes, the residual constraints attributes, translated into the correspondent set of local attributes on the basis of the mapping table.

The output of the Query Rewriting process is a set of local queries; each local query Q_L over a local class L is in a form supported by the local source of the class L. For relational sources, a local query Q_L over L will be in the form:

$$Q_L = \text{select} < Q_{L_{select-list}} > \text{from L where} < Q_{Lcondition} >$$

In our example, the local queries Q_L over L are:

```
Source ‘‘bootstrapOnto’’
Query on Local Interface ‘‘bootstrapOnto.gene’’:
SELECT gene.name, gene.map, gene.symbol
FROM gene
WHERE (name) like (’\%resistance\%’)

Source ‘‘gramene’’
Query on Local Interface ‘‘gramene.gene’’:
SELECT gene.name, gene.species
FROM gene
WHERE (name) like (’\%resistance\%’)

Source ‘‘graingenes’’
Query on Local Interface ‘‘graingenes.gene’’:
SELECT gene.type, gene.locus, gene.name, gene.chromosome,
gene.fullname, gene.synonym, gene.reference-title
FROM gene
WHERE (name) like (’\%resistance\%’)
```

3.2 Local Queries Execution / Fusion and Reconciliation

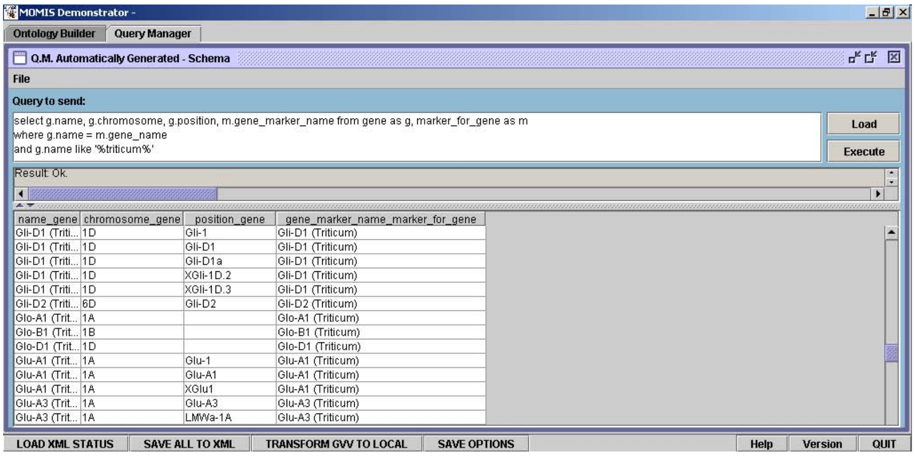
A local query L_Q is sent to the source including the local class L; its answer is transformed by applying the mapping functions related to L: in this way, we perform the conversion of the local class instances into the GVV instances. The result of this conversion is materialized in a temporary table. No data conversion function is necessary for our domain.

Temporary tables are fused and reconciliated into the global answer. In our example, Q_N is:

```
select "Join_Eng_gene_graingenes_gene".type AS type_1,
"Join_Eng_gene_gramene_gene".genus AS genus_1,
"Join_Eng_gene_bootstrapOnto_gene".name AS name_1,
"Join_Eng_gene_graingenes_gene".name AS name_2,
"Join_Eng_gene_gramene_gene".name AS name_3,
"Join_Eng_gene_bootstrapOnto_gene".specie AS species_1,
"Join_Eng_gene_gramene_gene".species AS species_2,
"Join_Eng_gene_bootstrapOnto_gene".map AS map_1,
"Join_Eng_gene_gramene_gene".description AS description_1,
"Join_Eng_gene_graingenes_gene".locus AS locus_1,
"Join_Eng_gene_gramene_gene".location_name AS location_name_1,
"Join_Eng_gene_bootstrapOnto_gene".chromosome AS chromosome_1,
"Join_Eng_gene_graingenes_gene".chromosome AS chromosome_2,
"Join_Eng_gene_gramene_gene".chromosome AS chromosome_3,
"Join_Eng_gene_gramene_gene".allele_symbol AS allele_symbol_1,
"Join_Eng_gene_bootstrapOnto_gene".allele AS allele_1,
"Join_Eng_gene_gramene_gene".allele_name AS allele_2,
"Join_Eng_gene_bootstrapOnto_gene".synonym AS synonym_1,
"Join_Eng_gene_graingenes_gene".synonym AS synonym_2,
"Join_Eng_gene_gramene_gene".synonym_name AS synonym_3,
"Join_Eng_gene_bootstrapOnto_gene".symbol AS symbol_1,
"Join_Eng_gene_gramene_gene".symbol AS symbol_2,
"Join_Eng_gene_bootstrapOnto_gene".gene_references AS references_1,
"Join_Eng_gene_graingenes_gene".reference_title AS references_2
from "Join_Eng_gene_graingenes_gene" full outer join
"Join_Eng_gene_bootstrapOnto_gene"
on ((("Join_Eng_gene_bootstrapOnto_gene".name) =
("Join_Eng_gene_graingenes_gene".name)))
full outer join "Join_Eng_gene_gramene_gene"
on ((("Join_Eng_gene_gramene_gene".name) =
("Join_Eng_gene_graingenes_gene".name))
OR (("Join_Eng_gene_gramene_gene".name) =
("Join_Eng_gene_bootstrapOnto_gene".name)))
```

3.3 Query on Multiple Global Classes

Another example of query can be seen in Fig.8. The picture shows a query on multiple Global Classes. In particular this query retrieves all the genes, chromosome, position of the gene and the marker for that particular gene for the Triticum species. This query is performed on two Global Classes, gene and marker-for-gene.



```
select g.name, g.chromosome, g.position, m.gene_marker_name from gene as g, marker_for_gene as m
where g.name = m.gene_name
and g.name like '%triticum%'
```

Result: Ok.

name_gene	chromosome_gene	position_gene	gene_marker_name_marker_for_gene
Gli-D1 (Triti...	1D	Gli-1	Gli-D1 (Triticum)
Gli-D1 (Triti...	1D	Gli-D1	Gli-D1 (Triticum)
Gli-D1 (Triti...	1D	Gli-D1a	Gli-D1 (Triticum)
Gli-D1 (Triti...	1D	XGli-1D.2	Gli-D1 (Triticum)
Gli-D1 (Triti...	1D	XGli-1D.3	Gli-D1 (Triticum)
Gli-D2 (Triti...	6D	Gli-D2	Gli-D2 (Triticum)
Glo-A1 (Trit...	1A		Glo-A1 (Triticum)
Glo-B1 (Trit...	1B		Glo-B1 (Triticum)
Glo-D1 (Trit...	1D		Glo-D1 (Triticum)
Glu-A1 (Trit...	1A	Glu-1	Glu-A1 (Triticum)
Glu-A1 (Trit...	1A	Glu-A1	Glu-A1 (Triticum)
Glu-A1 (Trit...	1A	XGlu1	Glu-A1 (Triticum)
Glu-A3 (Trit...	1A	Glu-A3	Glu-A3 (Triticum)
Glu-A3 (Trit...	1A	LMWa-1A	Glu-A3 (Triticum)

Fig. 8. A query on multiple Global Classes of the CEREALAB Global Virtual View

First, the query is decomposed into two queries on the single Global Classes:

```
Single Class Query_1 (JoinEngine JE001) :
select g.name , g.chromosome , g.position  from gene as g
where (name like  '%triticum%' )

Single Class Query_2 (JoinEngine JE002) :
select m.gene_name  from marker_for_gene as m
```

Each query is then further decomposed and rewritten as an equivalent set of queries expressed on the local sources.

```
Source "bootstrapOnto"
Query on Local Interface "bootstrapOnto.gene":
SELECT gene.name
FROM gene
WHERE (name) like ('%triticum%')

Source "gramene"
Query on Local Interface "gramene.gene":
SELECT gene.name
FROM gene
WHERE (name) like ('%triticum%')

Source "graingenes"
Query on Local Interface "graingenes.gene":
SELECT gene.locus, gene.name, gene.chromosome
FROM gene WHERE (name) like ('%triticum%')

Source "bootstrapOnto"
Query on Local Interface "bootstrapOnto.marker_for_gene":
SELECT marker_for_gene.gene_name
FROM marker_for_gene
```

Finally, the results of the Local Queries execution are then fused together according to the residual clause:

```
select g.name , g.chromosome , g.position ,
m.gene_name  from gene as g , marker_for_gene as m
where (g.name = m.gene_name )
```

4 Summary and Discussions

We described the design and realization of the CEREALAB database, a support for the research activity about cereal cultivars. This database has been developed as a Virtual View of two existing web databases, Gramene and Graingenes, integrated with another relation source designed to store the information achieved by the research group of the CEREALAB project. The CEREALAB database integrates information from existing databases according to a common ontology providing a unique interface to query different sources.

A possible improvement of the usage of the MOMIS system in this domain would be the use of existing biological ontologies, for example the Open Biological Ontologies (OBO) (`www.obo.org`), as an ontological support to perform the integration process. The primary area related to our work is the area of heterogeneous information integration. Many projects based on mediator architectures have been developed [19], [15], [13] . In this paper we described the application of the MOMIS system for both integrating data sources concerning the molecular biology domain and giving the possibility to querying them. The result of this work is the creation of a virtual database for the genotypic selection of cereal cultivars.

A completely different approach that can be promising in the biology domain is presented in [20], where the dynamic query translation task is addressed. The idea is to develop a light-weight domain based form assistant which can handle alternative sources in the same domain to help the user query across dinamically selected web sources. This approach is completely different from our system as it just suggests a possible translation of the user's query for different web data source, while MOMIS performs information integration and provides a unique interface to query multiple sources.

References

1. Ananthakrishna, R., Chaudhuri, S., & Ganti, V. , "Eliminating fuzzy duplicates in data warehouses", In VLDB Conference, (pp. 586597) (2002).
2. S. Bergamaschi, S. Castano, D. Beneventano, M. Vincini: "Semantic Integration of Heterogeneous Information Sources", Special Issue on Intelligent Information Integration, Data & Knowledge Engineering, Vol. 36, Num. 1, Pages 215-249, Elsevier Science B.V. 2001.
3. R. Benassi, S. Bergamaschi, A. Fergnani, D. Miselli: "Extending a Lexicon Ontology for Intelligent Information Integration", European Conference on Artificial Intelligence (ECAI2004). Valencia, Spain, 22-27 August 2004.
4. D. Beneventano, S. Bergamaschi, C. Sartori, M. Vincini "ODB-QOptimizer: a tool for semantic query optimization in OODB". ICDE'97, UK, April 1997.
5. D. Beneventano, S. Bergamaschi, F. Guerra, M. Vincini: "The MOMIS approach to Information Integration", IEEE and AAAI International Conference on Enterprise Information Systems (ICEIS01), Setbal, Portugal, 7-10 July, 2001.
6. D. Beneventano, S. Bergamaschi, F. Guerra, M. Vincini: "Synthesizing an Integrated Ontology". IEEE Internet Computing 7(5): 42-51 (2003).
7. D. Beneventano, S. Bergamaschi, C. Sartori: "Description Logics for Semantic Query Optimization in Object-Oriented Database Systems", ACM Transaction on Database Systems, Volume 28: 1-50 (2003).
8. D.Beneventano, S.Bergamaschi:"Semantic Search Engines based on Data Integration Systems". In Semantic Web: Theory, Tools and Applicantions (Ed. Jorge Cardoso), Idea Group Publishing, May 2006
9. R. G. G. Cattell, Douglas K. Barry: "The Object Data Standard: ODMG 3.0" Morgan Kaufmann 2000.

10. Chaudhuri, S., Ganjam, K., Ganti, V., & Motwani, R. . "Robust and efficient fuzzy match for online data cleaning". In ACM SIGMOD Conference (pp. 313324) (2003).
11. C. A. Galindo-Legaria, "Outerjoins as Disjunctions". SIGMOD Conference 1994, 348-358.
12. A. Halevy, A. Y. Halevy. "Answering queries using views: A survey". Very Large Database J., 10(4):270-294, 2001.
13. C. Li, R. Yerneni, V. Vassalos, H. Garcia-Molina, Y. Papakonstantinou, J. Ullman, M. Valiveti. "Capability Based Mediation in TSIMMIS", SIGMOD 98, Seattle, June 1998.
14. A.G. Miller. "A lexical database for English". Communications of the ACM, 38(11):39:41,1995.
15. R. J. Miller, M. A. Hernandez, L. M. Haas, L. Yan, C. T. H. Ho, L. Popa, and R. Fagin, "The Clio project: managing heterogeneity", ACM SIGMOD Record 30, 1 (March 2001), pp. 78-83.
16. F. Naumann, M. Haussler: "Declarative Data Merging with Conflict Resolution". International Conference on Information Quality (IQ 2002). 2002, pages 212-224.
17. A. Rajaraman , J. D. Ullman: "Integrating Information by Outerjoins and Full Disjunctions". PODS 1996, pages 238-248.
18. Tejada, S., Knoblock, C. A., & Minton, S. , "Learning object identification rules for information integration", Inf. Syst., 26 (8), 607633 (2001).
19. L. Yan, R. J. Miller, L. M. Haas, and R. Fagin, "Data-driven understanding and refinement of schema mappings", Proc. 2001 ACM SIGMOD Conference (SIGMOD '01), pp. 485-496.
20. Zhen Zhang, Bin He, Kevin Chen-Chuan Chang: Light-weight Domain-based Form Assistant: Querying Web Databases On the Fly. VLDB 2005: 97-108