

# Digital resource discovery: semantic annotation and matchmaking techniques <sup>\*</sup>

Devis Bianchini<sup>1</sup> Silvana Castano<sup>2</sup> Fulvio D'Antonio<sup>3</sup> Valeria De Antonellis<sup>1</sup>  
Mounira Harzallah<sup>4</sup> Michele Missikoff<sup>3</sup> Stefano Montanelli<sup>2</sup>

<sup>1</sup> Università degli Studi di Brescia - Dip. di Elettronica per l'Automazione  
Via Branze, 38 - 25123 Brescia - Italy  
{bianchin|deantone}@ing.unibs.it

<sup>2</sup> Università degli Studi di Milano - Dip. di Informatica e Comunicazione  
Via Comelico, 39 - 20135 Milano - Italy  
{castano|montanelli}@dico.unimi.it

<sup>3</sup> LEKS - Lab for Enterprise Knowledge and Systems  
IASI - CNR, Viale Manzoni 30, 00185 Rome (Italy)  
{missikoff|dantonio}@iasi.cnr.it

<sup>4</sup> Laboratoire d'Informatique de Nantes Atlantique  
Rue de la Houssinière, Nantes Cedex 03 (France)  
mounira.harzallah@univ-nantes.fr

**Abstract.** The Web is a huge container of digital resources, that is, sets of objects that are accessible and manageable by a computer. When dealing with structured queries (rather than with flat list of keywords), model matchmaking is a fundamental task in the digital resource discovery process. This paper presents MAGDA (MApping Generic Discovery Architecture), a reference architecture designed to implement different match-making algorithms aimed at defining mappings between digital resource models. The basic conceptual structures of MAGDA are described: a labeled oriented multigraph as reference model, model mappings, generic matchmakers and matchmaking techniques. Finally, MAGDA is used to support the definition and use of three different discovery solutions.

## 1 Introduction

The Internet gives access to the most massively distributed set of data in history. We will use the term *digital resource* to indicate the set of objects that are accessible and manageable by a computer. From this viewpoint the Web is a huge container of digital resources. Typical examples of digital resources are electronic documents, images and Web Services (e.g., “today’s weather report for Los Angeles”). A key challenge for the future is to face the problem of locating relevant resources on the web; such problem is alleviated somewhat by search services (e.g. Google, Yahoo), still far from being efficient and effective. In general, most methods for searching and retrieving resources employ a keyword-based strategy. Given the keywords, such a system searches its index and, upon

---

<sup>\*</sup> This paper has been partially funded by NoE INTEROP, IST Project n. 508011 - 6th EU Framework Programme.

finding hits, shows a list of URLs of Web resources with their respective titles, possibly ranked by their relevance scores (computed according to a relevance metrics).

When dealing with more structured queries, more sophisticated means for *resource discovery* are needed; in such cases model matchmaking is a fundamental task in the digital resource discovery process. For example, in Web Service discovery, a search engine has to determine possible matches between a “service request” description and the available “service offers” accessible through the network. On the basis of the matching results, proper mappings are then defined among the descriptions, and ranked service offers are eventually provided as discovery results. Aim of this paper is to present the reference architecture of MAGDA (*M*apping *G*eneric *D*iscovery *A*rchitecture), a generic reference model designed to implement different matchmaking algorithms in view of defining mappings between digital resource models. We describe the basic conceptual structures of MAGDA: labeled oriented multigraphs as abstract syntax for models, model mappings, generic matchmakers and matchmaking techniques.

MAGDA is aimed at developing solutions that primarily support the various aspects of modeling-related tasks, specifically:

- the *matchmaker engineer*, who develops new matchmaking solutions by defining matchmaking rules, application policies etc. instantiating the modules of the reference model;
- the *modeling language engineer*, who allows exogeneous mapping tasks (i.e. mapping among models expressed according to different formalisms), defining new model converters for the language developed; such mappings could be then used for “translation/change of formalism” tasks;
- the *model engineer*, that defines resource descriptions in available modeling languages and uses MAGDA for ensuring the consistency of multiple models by establishing the correspondences, in a semiautomatic way, by means of MAGDA instantiations.

In the paper we describe the use of MAGDA to support the definition and use of three different discovery solutions.

The paper is organized as follows. Basic conceptual structures of MAGDA are defined in Section 2. The MAGDA reference architecture is illustrated in Section 3. Section 4 presents three discovery solutions and their representation and possible use in MAGDA. Section 5 briefly reviews related work. In Section 6 concluding remarks are discussed.

## 2 The reference scenario

The complete scenario of digital resource discovery is depicted in Figure 1. The basic objects are the *MatchMaker*, a module able to draw correspondences between different models, and the *Mapping Consumer*, a module that receives as inputs a set of mappings and will use such information for specific purposes such as computing a global similarity ranking of the mapped models, using mappings to generate model or instance transformations or for query rewriting.

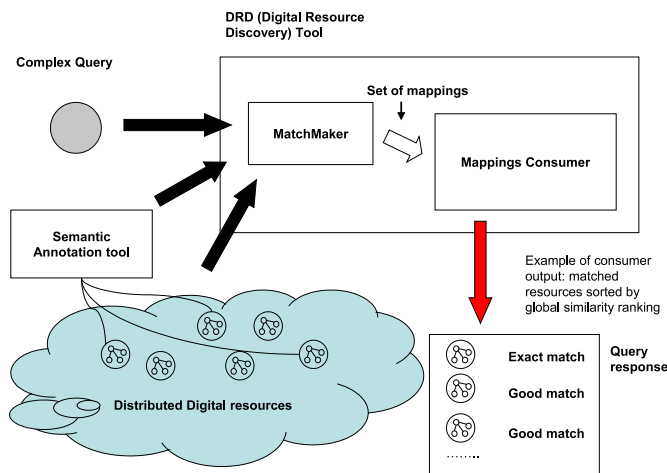


Fig. 1. Digital Resource Discovery scenario

We believe that the problems of discovering mappings and of “consuming” them should be addressed separately. In this document, we will address the former problem by showing an architecture, namely the MAGDA (Mapping Generic Discovery Architecture) framework, acting as an abstract reference model for designing matchmaking algorithms and for classifying the existing ones. We will show three existing ontology-related mapping discovery techniques and their relationships with the MAGDA framework.

## 2.1 Preliminary definitions

We assume that queries and digital resources will be represented by models; the problem is that several modeling languages exist; they differ both in syntax and in expressive power. Most modeling languages are thought to be both machine-processable and understandable by human users; models are usually presented to a human user by means of a diagrammatic representation; in general, this is possible given the intrinsic “graphic” nature of these languages (referring to the well-known mathematical objects called *graphs*); in fact, most languages can be represented using nodes and edges connecting them; nodes and edges can have a label associated with them. Given these assumptions, we have chosen to adopt an abstract graph formalism to represent models and to uniform different syntaxes for model representation according to this formalism.

**Definition 1.** A **Labeled Oriented Multigraph (LOMGraph)** is a 6-tuple  $G = \langle V, E, s, t, l_v, l_e \rangle$ , where  $V, E$  are two finite sets,  $s, t : E \rightarrow V$  are two mappings that indicate the source and the target of an edge,  $l_v : V \rightarrow \Sigma_V$ ,  $l_e : E \rightarrow \Sigma_E$

are two mappings from  $V$  and  $E$  towards the two finite sets of labels  $\Sigma_V$  e  $\Sigma_E$ , respectively.

We call **MOD** the set of multigraphs that are the representation of some models (expressed in any formalism or language) and **M2M** the set of functions  $f$  such that  $f: \mathbf{MOD} \rightarrow \mathbf{MOD}$ ; for example, let **UMLModels**  $\subseteq$  **MOD** be the set of multigraphs that represent UML models, we assume the existence of a program that converts an UML model into the graph model. We delegate to such function all the issues related to the translation from the original representation of the model, expressed according to some XML-based languages (e.g., XMI for UML diagrams, RDF/XML for Semantic Web Ontologies) or another digital format.

**Definition 2.** Given a model  $M \in \mathbf{MOD}$ ,  $EL_M = V \cup E$  is the set of the basic elements of the model (that is, nodes and edges). Given a graph  $\mathcal{G}$ , we define the set of all subgraphs of  $\mathcal{G}$  as  $Sub(\mathcal{G})$ .

In the *model mapping* process the two models are given in advance and they define the space of all the functions that put in correspondence elements in the source model with elements in the target one.

A mapping is *Element-to-Element* if it puts in correspondence one element of a model with exactly one of the other model; there are, by the way, more complex cases in which we map single elements in the source model with subgraphs in the target one (*Element-to-Subgraph* mapping) or, even, a subgraph in the source model with a subgraph in the target one (*Subgraph-to-Subgraph* mapping).

**Definition 3.** Given two models  $A$  and  $B$  ( $A, B \in \mathbf{MOD}$ ), a mapping is a relation  $m \subseteq Sub(A) \times Sub(B)$ .

This definition is very general and captures a great variety of classes of mappings: functional mappings, Element-to-Element mappings, etc. If we consider mappings as sets of edges that can be drawn to put in correspondence elements of different models, it is very common to associate with such edges a label expressing some other properties of the mapping itself.

**Definition 4.** Given two models  $A$  and  $B$  ( $A, B \in \mathbf{MOD}$ ) and a marking language  $L_M$ , a marked mapping is a relation  $m \subseteq Sub(A) \times Sub(B) \times L_M$ .

A typical example could be the association to an edge of a number in the interval  $[0,1]$  to state the degree of similarity between the elements. For example, if we use as marking language  $L_M = [0, 1]$ , we can generate (*Employee, Person, 0.7*) or (*Employee, Student, 0.3*).

Note that it is important to distinguish the concept of *model mapping* (just presented) from the one of *mapping discovery*, that is, the process of constructing the mappings, usually in a semantically meaningful way (that is, trying to relate semantically equivalent or similar entities in the two models).

### 3 The MAGDA Framework description

MAGDA is an architecture that allows for the implementation and classification of *matchmakers*, that is, objects able to discover mappings between pairs of models, by enforcing the separation of the different roles in the matchmaking process and the re-factoring of the basic components of the matchmaker itself. MAGDA can be thought as a reference model for the development of matchmaking tools, allowing for rapid prototyping by reusing matchmaking components. Figure 2 shows the overall MAGDA architecture.

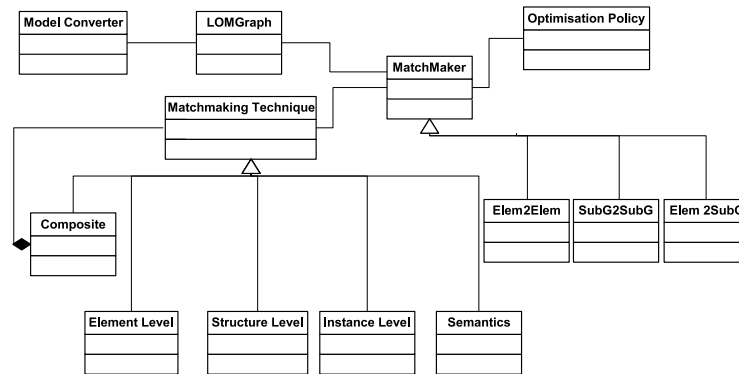


Fig. 2. The overall MAGDA architecture.

A matchmaking tool can be seen as constituted by at least the following elements:

- a set of **model converters**, to translate different languages into the abstract LOMGraph representation; any linear form for graph can be used (e.g., GXL or RDF);
- a set of **matchmaking techniques**, used to compare two subgraphs of the models to be mapped; the result of the application of a matchmaking technique to a pair of subgraphs is an element of a marked mapping;
- a set of **optimization policies**, used to limit the number of applications of the matchmaking technique that has to be used in order to enhance the performances of the overall process.

Matchmakers can be also classified according to the kind of output they produce:

- *Element-to-Element* matchmakers, where mappings are constituted by pairs of basic elements of source and target models (i.e., a single node/edge of the source model is put in correspondence with a single node/edge of the target one);

- *Element-to-Subgraph* (or, equivalently, *Subgraph-to-Element*), where mappings are constituted by pairs of the type  $(e_s, sg_t)$ , where  $e_s$  is a node/edge of the source model and  $sg_t$  is a subgraph of the target one;
- *Subgraph-to-Subgraph*, it is the most general kind of matchmaker; complex structures (subgraphs) are put in correspondence and mappings are constituted by pairs of the type  $(sg_s, sg_t)$ , where  $sg_s$  and  $sg_t$  are subgraphs of the source and the target model, respectively.

In the following, we provide further details about the other framework components.

### Model converters

A model converter receives as input a model in a given formalism represented by means of a linearization format (e.g., an XML-based language) and reduces it to a LOMGraph representation. This conversion should be performed by avoiding complex transformations (e.g., restructuring the initial model), but using the following rules:

- a node/edge in the original model becomes a node/edge in the LOMGraph;
- additional information associated with the elements should be kept by labeling the elements of the LOMGraph (e.g., labeling an edge with the “ISA” string to represent an ISA relationship).

### Matchmaking techniques

In the MAGDA framework a matchmaking technique is an algorithm that takes as input a pair of subgraphs and returns as output an element of a marked mapping. We classified matchmaking techniques into 4 main classes: *Element level*, *Structure level*, *Instance level*, *Semantics level*. For a description of such classes of algorithms see [10].

### Optimization policies

An *optimization policy* is used by a matchmaker to reduce the number of comparisons to be performed during the matchmaking process. The space of mappings can be very large; in fact, given two models  $A$  and  $B$ , the space of mappings  $MSPACE_{AB}$  is constituted by the set of all relations  $m \subseteq Sub(A) \times Sub(B)$ , and thus  $|MSPACE_{AB}| = |2^{Sub(A)}| \times |2^{Sub(B)}|$ . Therefore, using a “brute force” approach, i.e., performing all possible comparisons between a structure in the source model with a structure in the target one, will rapidly lead to an highly inefficient matchmaking procedure. Different tools and algorithms adopt various approaches to solve this problem; such approaches are concrete instantiations of the optimization policy component in the MAGDA framework. An optimization policy is a function  $p$  that takes as inputs two models  $A$  and  $B$  and returns as output a “pruned” version of the  $MSPACE_{AB}$  (i.e., a subset of it). For example, we consider an optimization policy to evaluate only *Element-to-Element* comparisons: given two models  $A$  and  $B$ , the policy prunes all the

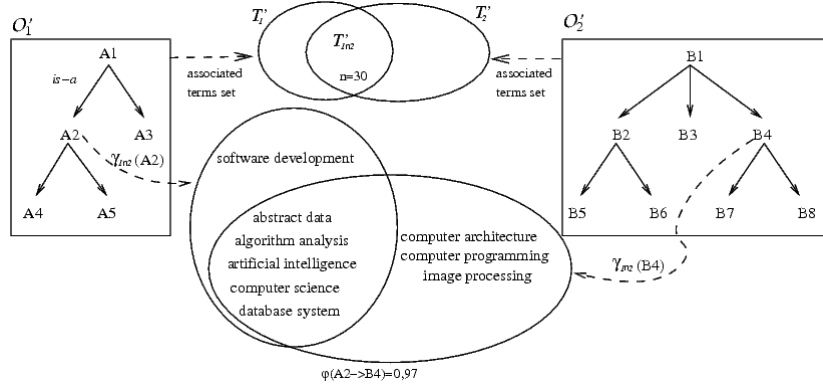
*Element-to-Subgraph* and *Subgraph-to-Subgraph* pairs and produces as output the set  $\{(x, y) : x \in EL_A, y \in EL_B\}$ , where  $EL_A$  (respectively,  $EL_B$ ) is the set of all basic elements of  $A$  (respectively,  $B$ ).

#### 4 Using the MAGDA framework to classify existing matchmaking approaches

The basic components of the MAGDA framework (i.e., model converter, matchmaking technique, optimization policy and the matchmaker) can be considered as a criterion for the classification of existing matchmaking approaches. We consider three different matchmaking tools, namely FC-MATCH [3, 4], H-MATCH [6, 7] and IAM [8, 11], and we discuss their position with respect to the MAGDA framework.

FC-MATCH: *an ontology-based matchmaking approach for Web service discovery*. According to the Service Oriented Architecture [12], a new paradigm has emerged for complex process execution by means of service selection and composition. Due to the ever-growing number of available services, automation of service discovery has become a crucial task. In order to enforce service discovery functionalities, the FC-MATCH ontology-based matchmaking algorithm has been developed. The FC-MATCH approach aims at discovering the kind and the degree of match between the description of a requested service  $\mathcal{R}$  and each description  $\mathcal{S}$  of a set of available services. To this end, FC-MATCH translates the WSDL service descriptions (operations, input/output parameters and service categories) into OWL-DL expressions and applies appropriate matchmaking techniques in order to identify correspondences between service descriptions by means of: (i) a *domain ontology*, used to assess subsumption and equivalence relationships between service description elements; (ii) a *thesaurus*, that contains terminological relationships (synonymy, hypernymy and so on) between names of domain ontology concepts; (iii) a *service ontology*, used to organize advertised services on different levels of abstraction by means of semantical relationships. Two strategies are combined to obtain a composite, element level, semantic matchmaking technique with respect to the MAGDA framework: i) a *logic-based matching*, where the domain ontology and the thesaurus are used to determine the kind of match between the request and the advertised services, classified in **exact** ( $\mathcal{R} \equiv \mathcal{S}$ ), **plug-in** ( $\mathcal{R} \sqsubseteq \mathcal{S}$ ), **subsume** ( $\mathcal{R} \sqsupseteq \mathcal{S}$ ), **intersection** ( $\neg(\mathcal{R} \sqcap \mathcal{S} \sqsubseteq \perp)$ ) and **mismatch** ( $\mathcal{R} \sqcap \mathcal{S} \sqsubseteq \perp$ ); ii) a *similarity-based matching*, where the thesaurus is used to quantify the degree of similarity between two service descriptions by means of properly defined coefficients [4]. The similarity value is also used to rank the final set of searching results. The semantic relationships in the service ontology are exploited to speed up the comparison phase. According to the MAGDA reference framework, during the matchmaking task, input/output parameters of the overall services are compared to evaluate how much the two services are based on the same information (*Element-to-element* comparisons) and operations with their corresponding input/output parameters are compared to evaluate how much the two services perform the same functionalities (*Subgraph-to-Subgraph* comparisons). An optimization policy is defined to

restrict the matching space only between homogeneous elements (e.g., inputs, outputs, operation names).



**Fig. 3.** Discovery of implications between concepts

**H-MATCH: ontology matchmaking in open networked contexts.** The H-MATCH algorithm and related techniques have been developed for ontology matchmaking in open networked contexts in order to provide a fully-automated matchmaking process that can be used for the alignment of two independent ontologies. H-MATCH takes two ontologies as input and returns the mappings that identify corresponding concepts in the two ontologies, namely the concepts with the same or the closest intended meaning. For the sake of internal representation of ontology specification languages, and in particular for Semantic Web languages like OWL, H-MATCH relies on a reference model, called H-MODEL. H-MODEL provides a LOMGraph-based representation of ontologies in terms of concepts, properties, and semantic relations. H-MATCH defines element-to-element mappings as correspondences between a concept of the first ontology and a (set of) concept of the second one with an annotation that indicates their semantic affinity value. With respect to the MAGDA framework, H-MATCH is an ontology matchmaker enforcing both element level and structure level matchmaking techniques. Given two concepts  $c$  and  $c'$ , H-MATCH calculates a semantic affinity value  $SA_{c,c'}$  as the linear combination of a linguistic affinity value  $LA_{c,c'}$  and a contextual affinity value  $CA_{c,c'}$ . The H-MATCH linguistic affinity function provides a measure of similarity between two ontology concepts  $c$  and  $c'$  computed on the basis of their linguistic features. For the linguistic affinity evaluation, H-MATCH relies on a thesaurus of terms and terminological relationships automatically extracted from the WORDNET lexical system. The H-MATCH contextual affinity function provides a measure of similarity by taking into account the contextual features of the ontology concepts  $c$  and  $c'$ . The context of a concept can include properties, semantic relations with other concepts and property

values. The context can be differently composed to provide a composite matchmaking approach and to consider different levels of semantic complexity. Four matching models, namely, *surface*, *shallow*, *deep* and *intensive*, are defined to this end. A ranking strategy is defined to restrict the concepts to be considered during the matchmaking process. The idea is to focus the matchmaking process on a subset of concepts, those which have a high probability to be similar. In H-MATCH, the concepts in the input ontologies are ranked according to an importance-based criterion and only the top-k ranked concepts are considered during the matchmaking procedure.

*IAM: an implicative alignment method.* The Implicative Alignment method (IAM) allows for finding an asymmetric relationship between two concepts, using a probabilistic model of deviation from statistical independence called *implication intensity* [11]. IAM is based on ontologies represented in XML as a set of concepts related by subsumption relationships. Moreover, a set of documents is associated to each concept and each document is indexed by a set of *significant terms*. The basic idea of IAM is that a concept  $\mathcal{A}$  is more specific than a concept  $\mathcal{B}$  if the set of significant terms of  $\mathcal{A}$  is almost included in the set of significant terms of  $\mathcal{B}$ . The goal of IAM is to find an *Element-to-Element* mapping between concepts by observing the sets of their corresponding significant terms. The approach follows two steps: (i) extraction and selection of significant terms for each concept from documents related to it, and (ii) discovery of significant implications between concepts. The first step could be performed using well-known techniques, for example, ACABIT [8], that extracts terms from POS tagged and stemmed texts. In the second step, implicative relationships between concepts are evaluated by using a probabilistic model of deviation from statistical independence (i.e., *implication intensity*). The *implicative tendency*  $\mathcal{C}_1 \rightarrow \mathcal{C}_2$  between two concepts  $\mathcal{C}_1$  and  $\mathcal{C}_2$  is evaluated through the *implication intensity*:  $\varphi(\mathcal{C}_1 \rightarrow \mathcal{C}_2) = Pr(N_{\mathcal{C}_1 \wedge \overline{\mathcal{C}_2}} \leq n_{\mathcal{C}_1 \wedge \overline{\mathcal{C}_2}})$ , where  $n_{\mathcal{C}_1 \wedge \overline{\mathcal{C}_2}}$  is the number of terms associated to  $\mathcal{C}_1$  that are not associated to  $\mathcal{C}_2$  and  $N_{\mathcal{C}_1 \wedge \overline{\mathcal{C}_2}}$  is the expected number (under independence hypothesis) of terms associated to  $\mathcal{C}_1$  that are not associated to  $\mathcal{C}_2$  (Figure 3). To limit the number of discovered implications, only *generative implications* having a value up to a given threshold are taken into account. Generative implication has the most general antecedent and the most specific consequent, that is,  $\mathcal{A} \rightarrow \mathcal{B}$  is a generative implication if  $\forall \mathcal{X}, \mathcal{Y}$  such that  $\mathcal{A} \leq \mathcal{X}$  and  $\mathcal{Y} \leq \mathcal{B}$ ,  $\varphi(\mathcal{X} \rightarrow \mathcal{Y}) \leq \varphi(\mathcal{A} \rightarrow \mathcal{B})$ .

The results of the comparison among the considered matchmaking approaches and their relationships with the MAGDA framework are summarized in Table 4. Note that FC-MATCH, H-MATCH and IAM provide specific model pre-processing techniques to transform data sources to be matched in a LOMGraph-based representation. We want to stress that the adoption of a LOMGraph-based formalism allows heterogeneous data sources with their own specific representation models to be supported by the MAGDA framework.

## 5 Related Work

In recent years the ever-growing number of ontologies and digital resources whose descriptions are based on ontologies has made the use of ontology-based match-

Approach name	Model pre-processing	Matchmaking technique	Optimization policy	Matchmaker type
FC-MATCH [3, 4]	Translation of WSDL descriptions into OWL-DL expressions	Element level Semantics Composite	Comparison of homogeneous service description elements (Inputs, Outputs, Operations)	Element-to-Element Subgraph-to-Subgraph
H-MATCH [6, 7]	Translation of heterogeneous ontology specifications in a graph-based formalism	Element level Structure level Composite	Concept selection according to importance-based ranking	Element-to-Element
IAM [8, 11]	Translation of heterogeneous data sources in an XML-based formalism	Element level Structure level Instance level Composite	Sub-tree pruning based on generative implications	Element-to-Element

**Table 1.** An example of classification of existing ontology matchmaking approaches according to the MAGDA framework

making algorithms a fundamental issue for resource discovery over the Web. The approaches presented in Section 4 aim at improving current matchmaking techniques in their respective contexts.

The problem of designing a generic matchmaking algorithm has been addressed by several approaches in literature. In [5] a generic matchmaking framework is proposed which is aware of the links between matchmaking algorithms and the kinds of ontologies they have been originally designed for or successfully applied to, allowing a more flexible and automatically triggered use of various matchmaking strategies. The proposed framework uses semantic descriptions of both single matchmaking algorithms and Web ontologies, which are related by means of rules to optimize matchmaking results. The framework is composed of four elements: a matching repository, containing reusable matching components and metadata describing their properties; an ontology repository, that contains ontologies and metadata describing their properties; a rule repository, that links the properties of ontologies and matching components; a matchmaking engine, that is in charge of selecting which algorithms are applicable to a specific set of inputs.

The COMA framework [9] supports different kinds of schemas (such as XML and relational schemas) and provides an extensible library of matchmaking algorithms, a component to combine the results of the search and an extensive functionality to evaluate the matching effectiveness. However the proposed combined methods are unappropriate for complex situations; in such cases users are required to manually customize the matchmaking workflow, so the framework can not be directly employed in open environments, where human interactions must be limited.

The CMC system [13] is focused on schema matchmaking approaches and combines multiple matchmakers according to the credibility prediction. The CMC approach is based on the observation that each matchmaker has different performances in different matchmaking tasks. The CMC system dynamically predicts the accuracy of each matchmaker based on the characteristics of

the current task and evaluates the matchmaker’s credibility. These credibilities are then used as weights in aggregating the matching results of different matchmakers into a combined one. Starting from the base matchmakers as modules, the user could connect them freely. The CMC system also provides a default connection policy.

Model management issues addressed in [1, 2] constitute a basis for service discovery approaches, which start from different existing models. Bernstein et al. [2] developed a generic infrastructure to offer an order-of-magnitude productivity improvement to the builders of model-driven applications, by abstracting the models on which these applications are based and mappings between models and then by performing operations on the abstract representations. Key operations of model management are *match* (to find mappings between two given models), *merge* (to obtain an integration of two models based on mappings) and *compose* (to perform the composition of two mappings).

Atzeni et al. [1] proposed a model management operator called *ModelGen*: given a source data model  $M_1$ , a target data model  $M_2$  and a source schema  $S_1$  expressed according to  $M_1$ , *ModelGen* generates a target schema  $S_2$  according to  $M_2$ . This process is based on the notion of *metamodel*, that is, a set of generic constructs (*metaconstructs*) that can be used to express the constructs of single models, which are instances of the metamodel. The translation of a schema from a model to another one is defined in terms of translations over the metaconstructs. A customizable and extensible tool is proposed to implement *ModelGen*, based on a relational dictionary that stores the metadata of interest (the metamodel, models and schemas). The translation process is a composition of basic transformations, where each transformation is expressed by means of a set of rules written in a Datalog dialect with OID-invention based on Skolem functions.

However, existing approaches are still tailored to specific kinds of digital resources to be matched (schemas, ontologies, etc.) and do not clearly distinguish the three roles of modeling language engineer, matchmaker engineer and designer. The MAGDA framework is general enough to be a reference for a set of very different matchmaking approaches, ranging from ontology to service matchmaking techniques. Section 4 shows how the framework can also be instantiated in complex approaches, which themselves constitute value-added strategies in their respective application fields. Moreover, model converters allow to start from very different representation models and pruning policies allow to improve performances of the matchmaking approaches. Finally, the MAGDA proposal offers a reference framework to design new discovery approaches where the roles of modeling language engineer, matchmaker engineer and designer are clearly separated. In this vision, MAGDA aims at constituting a very general reference framework for the classification of existing techniques and for the easy development of new ones.

## 6 Concluding remarks

In this work, we presented the MAGDA framework for ontology-based generic mapping discovery. The MAGDA framework can be adopted to foster reuse

and combination of existing ontology matchmaking approaches, aiming at selecting the most appropriate matchmaking techniques according to the specific case. Furthermore, the proposed architecture can be considered as a conceptual framework to classify existing matchmaking approaches as discussed in Section 4.

Future work will be devoted to further analyze the role of the MAGDA framework with respect to digital resource discovery issues. In particular, we plan to integrate the MAGDA framework within a comprehensive platform for ontology-based digital resource discovery. In this context, ontology matchmaking techniques are adopted to compare ontologies and digital resources whose descriptions are based on ontologies with a given target request and to identify possible matchings. We are working on the definition of appropriate query policies which allow the MAGDA framework to select and to combine the matchmaking techniques that best suit each specific matching case.

## References

1. P. Atzeni, P.A. Bernstein, and P. Cappellari. ModelGen: Model Independent Schema Translation. In *Proc. of the 21st International Conference on Data Engineering (ICDE'05)*, pages 1111–1112, Tokyo, Japan, 2005.
2. P.A. Bernstein and J. Madhavan. Generic Schema Matching with Cupid. In *Proc. of the 27th VLDB Conference*, 2001.
3. D. Bianchini, V. De Antonellis, and M. Melchiori. Capability Matching and Similarity Reasoning in Service Discovery. In *Proc. of the CAiSE Int. Workshop on Enterprise Modeling and Ontologies for Interoperability, EMOI 2005*, Porto, Portugal, 2005.
4. D. Bianchini, V. De Antonellis, B. Pernici, and P. Plebani. Ontology-based Methodology for e-Service discovery. *Journal of Information Systems, Special Issue on Semantic Web and Web Services*, 2004.
5. E.P. Bontas and M. Mochol. A Metadata-Based Generic Matching Framework for Web Ontologies. Technical Report B-05-08, Institut fur Informatik, Freie Universitat Berlin, 2005.
6. S. Castano, A. Ferrara, and S. Montanelli. *Web Semantics and Ontology*, chapter Dynamic Knowledge Discovery in Open, Distributed and Multi-Ontology Systems: Techniques and Applications. Idea Group, 2005.
7. S. Castano, A. Ferrara, S. Montanelli, and G. Racca. Matching Ontologies in Open Networked Systems: Techniques and Applications. *Journal on Data Semantics (JoDS)*, to appear, 5, 2005.
8. B. Daille. Conceptual Structuring through Term Variations. In *Proc. of the ACL 2003 Workshop on Multiword Expressions: Analysis, Acquisition and Treatment*, Sapporo, Japan, 2003.
9. H.H. Do and E. Rahm. COMA - a System for Flexible Combination of Schema Matching Approaches. In *Proc. of the 28th VLDB Conference*, 2002.
10. J. Euzenat et al. State of the Art on Ontology Alignment. Technical Report D2.2.3, Knowledge Web Consortium, 2004.
11. R. Gras. L'implication Statistique, une Nouvelle Méthode Exploratoire de Données. *La pensée sauvage*, 1996.
12. HP - Web Services Concepts. A Technical Overview. [http://www.bluestone.com/downloads/pdf/web\\_services\\_tech\\_overview/w.pdf](http://www.bluestone.com/downloads/pdf/web_services_tech_overview/w.pdf).
13. K. Tu and Y. Yu. CMC: Combining Multiple Schema-Matching Strategies based on Credibility Prediction. In *Proc. Of 10th International Conference on Database Systems for Advanced Applications, to appear*, 2005.