Multidimensional service-oriented ontology mapping

Nuno Silva* and João Rocha

GECAD – Knowledge Engineering and Decision Support Research Group, Instituto Superior de Engenharia do Porto,
4200-072 Porto, Portugal
E-mail: Nuno.Silva@dei.isep.ipp.pt E-mail: Jrocha@ipp.pt
*Corresponding author

Abstract: Ontology mapping is the process whereby semantic relations are defined between two ontologies at conceptual level, which in turn are applied at data level transforming source ontology instances into target ontology instances. This paper focuses on the formalisation, representation and automatic specification of semantic relations between two ontologies. The described approach grounds on the notion of service, which represents not only the transformation capabilities of the system, but also the expertise in the manipulation of semantic relations. The main contributions of this paper are the formalisation of the ontology mapping process and the specification of the multidimensional service-oriented architecture.

Keywords: interoperability; mapping; matching; ontology; semantic web; services.

Reference to this paper should be made as follows: Silva, N. and Rocha, J. (2005) 'Multidimensional service-oriented ontology mapping', *Int. J. Web Engineering and Technology*, Vol. 2, No. 1, pp.50–80.

Biographical notes: Nuno Silva received a PhD degree in informatics from the University of Trás-os-Montes e Alto Douro in 2005. He joined the School of Engineering of the Polytechnic Institute of Porto in 1989 as a student and in 1995 as Assistant Professor. Since 2004 he has been Adjunct Professor. Currently he is a member of the GECAD – Knowledge Engineering and Decision Support Group. His main research interests include ontology engineering, semantic web and agent-based systems.

João Rocha received a PhD degree in computer science from the University of Porto in 2000 and a master's degree in computers and electrical engineering from the University of Porto in 1991. He joined the School of Engineering of the Polytechnic Institute of Porto (Portugal) where he has been a Coordinator Professor since 2001. Currently he is member of the GECAD – Knowledge Engineering and Decision Support Group – where he leads the special interest group on knowledge representation and ontologies. His main research interests include knowledge engineering, ontology engineering and the semantic web.

1 Introduction

The semantic web, the next WWW trend, suggests the annotation of web resources with machine-processable metadata, which can provide tools to analyse meaning and semantic relations between documents and their parts. Ontologies, as a means for conceptualising

and structuring knowledge, are seen as the key to the realisation of the semantic web vision. Ontology allows the explicit specification of a domain of discourse, which permits access to and reason about an agent knowledge. Ontologies raise the level of specification of knowledge, incorporating semantics into the data, promoting its exchange in an explicit understandable form. The semantic web and ontologies are therefore fully geared as a valuable framework for distinct applications, namely business applications, such as e-commerce and B2B. However, ontologies do not overcome *per se* any interoperability problems, since it is hardly conceivable that a single ontology may be applied for all kind of domains and applications. Ontology mapping does not intend to unify ontologies and their data, but to transform ontology instances according to the semantic relations (mapping relations) defined at conceptual level. Repositories are therefore kept separated, independent and distinct, maintaining their complete semantics and contents. The solution proposed in this paper adopts a declarative specification of mappings, hiding the procedural complexity of specification and execution, while its proposed open architecture allows the integration of new mapping relations into the system, improving mapping capabilities throughout the overall process.

Section 2 presents the motivations and goals associated with this work. Section 3 formalises the ontology concept as understood in the context of this work. The ontology mapping process is described and a few premises are specified in Section 4, allowing the description and analysis of features from some related works. In Section 5, the semantic bridging ontology is presented and formalised, which supports further reasoning about the semantic bridging representation. Section 6 presents the service-oriented architecture of the MAFRA Toolkit system developed in the scope of this work, and proposes extensions to this approach that can be applied in other phases of the ontology mapping process. In Section 7, the proposed multidimensional service-oriented approach is applied in the automation of the semantic bridging phase. Section 8 presents the applications of the MAFRA Toolkit in several research projects and describes performance experiences in comparison with a similar tool. Finally, Section 9 provides an overview of achievements and the main contributions of the paper followed by a description of current and future efforts.

2 Motivations and goals

2.1 Motivations

Considering ontology as an explicit specification of a conceptualisation (Gruber 1993), it encodes individual (or corporate) perspectives of the universe, leading to further integration conflicts. The distributed nature of the web amplifies this problem and even the suggested exploitation of lexical and domain super ontologies do not overcome integration problems. In fact, living in the same world does not mean that everyone sees and understands it in the same way. Additionally, ontologies are typically developed for private use only, with minimal interconnection purposes. As a consequence, there is no requirement to annotate or link conceptual descriptions to more abstract ontologies.

Ontology mapping is a transversal key technology for many different problems regarding data integration, also referred as data translation. It concerns the transforming of one dataset, described according to a certain model, into another dataset described with respect to another model and to each-one's specificities and requirements. In particular, we distinguish between two categories of problems:

- Off-line data integration. With the advent of enterprise and organisations' integration and merging, separated heterogeneous databases have to be integrated in a fast and reliable manner. In this scenario, source and target information repositories are known *a priori* and a (long) integration phase is possible. Data migration, data warehouse clean and transform (Critchlow et al., 1998), data model evolution (Mitra and Wiederhold, 2001) and data driven portal generation (Maedche et al., 2002b) are variant problems of off-line data integration;
- Online data integration. Web services are emerging as a major paradigm for the semantic web implementation of business-process and promotion of B2B and e-commerce to an unprecedented level. It is necessary to develop mechanisms capable of mediation and alignment between distinct data models. Syntactical, structural and semantic heterogeneity arise in different degrees and variations between different information repositories. The ability to semi-automatically map between different knowledge bases and answer agents' queries from different agents becomes of primordial importance to the success of truly dynamic and autonomous businesses. Negotiation and argumentation between partners becomes therefore a major requirement for ontology mapping systems.

The final motivation is in respect of the supporting technology. In the context of the semantic web, the base ontology representation language is RDFS (W3C 2003). RDFS stands for Resource Description Framework Schema and it is an extension to RDF, allowing the specification of domain vocabulary. Other ontology representation languages have been proposed in respect to the semantic web, most of them extend RDFS and inherit its primitive representation mechanisms. However, none provides minimal constructs for mapping between ontologies. Elements like *rdfs:sameAs* are clearly insufficient. Description-logics theory included in OIL, DAML and OWL provide useful but insufficient constructs.

2.2 Driving vectors

An infinite number of possible mapping relations between two ontologies may exist. Consequently, the ontology mapping process is not a straightforward or a mathematical problem, thus the need for goals and quality indicators. Six driving vectors have been identified:

- applicability is concerned with the type of mapping relations that are supported by the system
- semantic expressivity is concerned with the capacity of the system to explicitly express semantic relations
- automation is concerned with the support that the ontology mapping system is able to provide to the human being
- modularisation is concerned with the system characteristics to be built upon the combination of small, simple modules into more complex ones

- reutilisation of components is concerned with the exploitation and application of knowledge created from previous ontology mapping experiences and its recycling when obsolete
- declarativity is concerned with the capabilities of the system to supply conditions so that the domain expert can focus on the semantics (what to) instead of the syntax (how-to) – maximising declarativity will improve quality and productivity, at the same time minimising software development and customisation mistakes and therefore costs.

Despite their abstract nature, the specification of those vectors is very important in the context of this work as it permits maximising the usefulness of research efforts.

Additionally, it is important to notice that this work grounds on the semantic web technology and must primarily provide technology for the semantic web. In particular, we consider:

- ontologies as a domain description modelled in RDFS or any language that can be grounded to RDFS
- the distributed and ever-evolving nature of the semantic web, namely respecting distributed ontologies and its unpredictable evolution;
- the ambiguous nature of conceptualisations, namely due to the private (or corporate specification of ontologies)
- the incomplete nature of the semantic web, namely concerning the description of models, constraints and data.

The following sections will consider these constraints and drivers in the analysis, specification and development of the ontology mapping system.

3 Ontology

Because many definitions exist for ontology and knowledge base it is important to univocally define both terms.

Ontology is a tuple in the form of:

$$O \coloneqq (C, is\_a, P, \sigma)$$

where:

- C is the set whose elements are called concepts, or classes, defined by the domain expert to describe the domain
- *is_a* is the partial order on C representing the hierarchical relation between concepts:

$$is\_a \subseteq C \times C$$

Commonly referred as 'subclass of' relation, *is_a* is binary, reflexive, transitive, and anti-symmetric relation:

- P is the set whose elements are called properties, defined by the domain expert to describe the domain
- σ is a function which assigns to every property their domain and range concepts.

Because the RDFS framework adopt a triple representation of information (Hsieh, 1990), σ is always a binary function in the form of:

$$\sigma : P \to \left(2^C \setminus \{\varnothing\}\right) \times \left(2^{(C \cup \{Literal\})} \setminus \{\varnothing\}\right)$$

The following functions are available:

- domain: $P \to 2^C \setminus \{\varnothing\}$ gives the set of domain concepts of the property
- $P \to 2^{C \cup \{Literal\}} \setminus \{\varnothing\}$ gives the set of range concepts of the property.

When the range of a property is a domain-defined concept it is said to be a Relation. When the range of a property is a primitive type (Literal) it is said to be an Attribute.

Despite it is not explicitly referred in the ontology definition, ontology entities are the elements of the domain of discourse, and are therefore the union of concepts, properties and Literal:

$$E = C \cup P \cup \{Literal\}$$

Knowledge Base (KB) is an instantiated ontology, also known as populated ontology (Kalfoglou and Schorlemmer, 2003). Knowledge base is formally defined as the tuple:

$$KB := (O, I, instC, instP)$$

where:

- O is an ontology
- I is a set of elements called instances or objects, which corresponds to the instantiation of $C \cup \{Literal\}$

  instC is the function that associates ontology concepts with the set of respective instances:

$$instC : C \to 2^I$$

- *instP* is the relation that associates pairs of instances through ontology properties, corresponding to a relationship between the two instances:

$$instP : P \to 2^{I \times I}$$

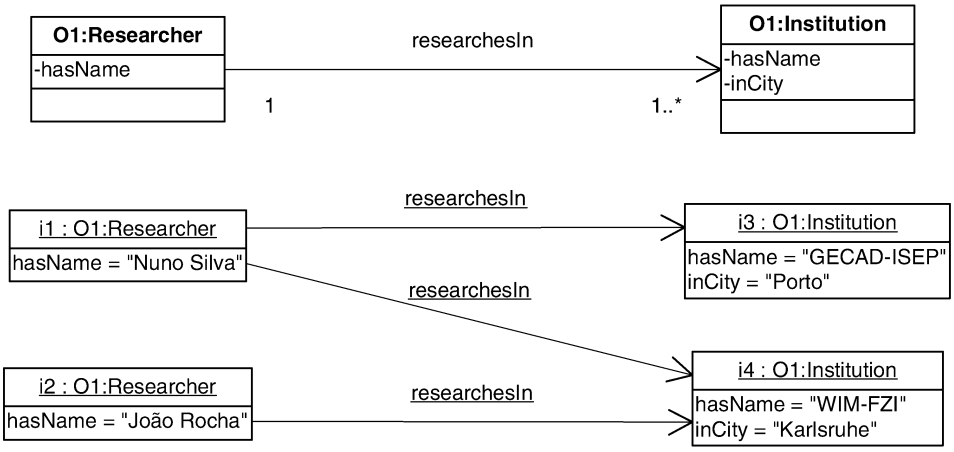
Accordingly, an ontology instance is any data element represented according to (and coherent with) the domain ontology. As an example, the following definition corresponds

to the specification of both ontology and knowledge base, which is represented using UML notation in Figure 1:

$$
\begin{aligned}
KB_1 &:= (O_1, I_1, instC_1, instP_1)\\
O_1 &:= (C_1, is\_a_1, P_1, \sigma_1)\\
C_1 &:= \{Researcher, Institution\}\\
is\_a_1 &:= \{\ \}\\
P_1 &:= \{hasName, researchesIn, inCity\}\\
\sigma_1 &:= \begin{Bmatrix} hasName(Researcher, Literal), hasName(Institution, Literal),\\ researchesIn(Researcher, Institution), inCity(Institution, Literal) \end{Bmatrix}
\end{aligned}
$$

$$
\begin{aligned}
I_1 &:= \{i_1, i_2, i_3, i_4, "Nuno\ Silva", "João\ Rocha", "GECAD - ISEP", "WIM - FZI", "Porto", "Karlsruhe"\}\\
instC_1 &:= \begin{Bmatrix} Researcher(i_1), Researcher(i_2), Institution(i_3), Institution(i_4),\\ Literal("Nuno\ Silva"), Literal("João\ Rocha"),\\ Literal("FZI - WIM"), Literal("GECAD - ISEP"), Literal("Porto"), Literal("Karlsruhe") \end{Bmatrix}\\
instP_1 &:= \begin{Bmatrix} hasName(i_1, "Nuno\ Silva"), hasName(i_2, "João\ Rocha"), hasName(i_3, "GECAD - ISEP"),\\ hasName(i_4, "WIM - FZI"), researchesIn(i_1, i_3), researchesIn(i_1, i_4), researchesIn(i_2, i_3),\\ inCity(i_3, "Porto"), inCity(i_4, "Karlsruhe") \end{Bmatrix}.
\end{aligned}
$$

Figure 1 UML representation of a simple ontology and knowledge base



4 Ontology mapping problem

4.1 Ontology mapping

Informally, ontology mapping is the process whereby semantic relations are defined at ontological level between source ontology entities and target ontology entities; and then further applied at instance level, transforming source ontology instances into target

ontology instances. Ontology mapping is therefore a two-phase process. The first phase, named (semantic bridging) specification phase, occurs at the meta-level, in the sense that the object of process are the domains at instance-level. This phase is formally defined as a relation between source and target ontology entities:

$$M \subseteq 2^{E^s} \times 2^{E^t}$$

M is an ontology mapping document, or simply ontology mapping, which contains the necessary and sufficient information required in the second phase. The ontology mapping document will be further described in Section 5. The second phase, named (semantic bridging) execution phase, occurs at instance level. This phase is formally defined as the relation between source knowledge base instances and target knowledge base instances, according to the ontology mapping document defined previously:

$$T(M) \subseteq 2^{I^s} \times 2^{I^t}$$

Whereas important conflicts arise from syntactic and data model heterogeneity, this work specially focuses on the identification and analysis of semantic heterogeneities arising in the context of semantic bridging.

4.2 Semantic heterogeneity

Semantic heterogeneity has been studied for a long time (Fodor et al., 2002; Hammer and Medjahed, 1993; Sheth and Larson, 1990; Stuckenschmidt and Wache, 2000; Visser et al., 1997) but no agreement exists on what semantic heterogeneity univocally is, neither is it possible to enumerate all its facets. Unlike previous referred works, the analysis of semantic relations in this work aims to identify and characterise their components. This approach allows better specification of requirements and possibilities of the envisaged system. Five dimensions have been identified (Maedche et al., 2002a).

- entity type dimension, which reflects the type of ontological entities being related
- transformation dimension, which respects the function to transform instances
- cardinality dimension reflects the number of ontology's entities being related
- constraint dimension, which respects the constraints holding during the execution phase
- structural dimension, which reflects the relations between semantic relations.

These dimensions are systematised in Table 1, corresponding to the envisaged support concerning the applicability vector referred to in Section 2.2.

Table 1 Summary of the multi-dimensional characterisation of semantic relations

Dimensions	*Characteristics*
Entity type	
	Concept to concept
	Concept to property
	Property to concept
	Property to property
	Entity to instance
Transformation	
Function	Basic functions required
	Combination of functions
	Integration of new functions
Directionality	Unidirectional
	Manually bidirectional
	Automatically bidirectional
Cardinality	
	0:1
	0:n
	1:1
	1:n
	n:0
	n:1
	m:n
Constraint	
	Not constrained
	Ontological entities-based
	Non-ontological entities-based
Structural support required	
	Object-oriented
	Property-centric
	Flow execution control

According to the envisaged support and considering the maximisation of the driving vectors previously identified, the following sections will describe and analyse some related work.

4.2.1. Protége approach

The work described by Gennari and colleagues (1994) is probably the first attempt to systematise and describe ontology mapping relations through an ontology. This ontology serves not only as a description of the ontology mapping domain, but also as an ontology mapping representation language, when instantiated in specific mapping scenarios. The types of relations defined in this ontology are:

- renaming relations, are able to copy the values of the source instances into the target instances
- filtering relations, apply filters (constraints) and/or transformations (functions) to create the target instances from source instances
- class relations, are able to fill up target instances according to information captured from source classes (entities) instead of their instances.

While the renaming relation type is far insufficient, filtering relations are able to cope with complex heterogeneity problems. However, as referred to in Gennari et al. (1994), specification of filtering relations might not be easy. Class relations correspond, according to the previously defined terminology, to the 'entity to instance' semantic relation.

One important modelling concern of this ontology is the simplicity of the types of semantic relations. Authors claim that if complex semantic relations are necessary, then the user or domain expert is suggested to programmatically (procedurally) implement the mapping.

Later, Park and colleagues (1998) expanded this approach in respect to the feedback received in the application of the first mapping ontology. A valuable set of desiderata and mapping dimensions is presented, giving rise to the following types of semantic bridges (referred as mapping):

- instance mapping, which corresponds to the 'concept to concept' type of semantic relations: despite the target instance creation, instance mappings are also used to embrace the mappings responsible for the slot mappings
- slot mapping corresponds to the 'property to property' types of semantic relations, and are therefore responsible for the creation of target properties from source entities.

Each set of one Instance mapping and slot mapping results in one new instance of a mapping, referred as either renaming; direct; lexical; or transformation mapping, depending on the type of slot mappings included.

No published improvements had been released since 1996 until late in 2003, when Crubézy and colleagues (2003) reviewed the work. Still, not many details were presented though, apparently, the major outcome of this update is the current implementation that exploits the semantic web technologies (namely the RDFS-based ontology representation language) and its correlation with the Unified Problem-solving Method development Language (Fensel et al., 1999) in scope of semantic web. Yet, this is a very valuable reference work in this research domain.

4.2.2 The approach of Stuckenschmidt and colleagues

Stuckenschmidt and colleagues (Stuckenschmidt and Visser, 2000; Stuckenschmidt and Wache, 2000) researched on the combination of context transformation and integration rules on the integration of Geographic Information Systems. Context transformation corresponds to the transformation of data respecting the specificities of a context, into data respecting the semantics of another context. Contexts are conceptualisations of the domain of knowledge, which corresponds to the notion of ontology as introduced in Section 3. The process includes two phases: contextualisation and integration. Once contextualised through contextualisation rules (i.e. once the source data is semantically compatible with target context) it is structurally integrated through integration rules.

Two other types of rules represent both context transformation rules and integration rules: combination and replacement rules. In fact, the approach distinguishes between the representation level (combination and replacement rules) and the application level (context transformation and integration rules). While this separation might sound beneficial, it turns out to complicate the perception of the approach. The described approach provides no support for object-oriented modelling structure. Nesting and streaming context transformation rules provide support to some extent to the property-centric modelling feature of the ontology representation language, but the process and the relations between rules becomes rather complicated and difficult to automate.

Stuckenschmidt and colleagues (Stuckenschmidt and Visser, 2000; Stuckenschmidt and Wache, 2000) assumed some limitations of this approach and proposed another strategy based on the automatic reclassification of concepts through the standard logic-based reasoners, such as the FaCT reasoner (Horrocks, 1998). The proposed approach is based on the specification of necessary and sufficient conditions for class (concept) membership, *à la* description logics. On one hand, the necessary conditions allow the derivation of non-explicitly specified properties of source concept instances, while sufficient conditions allow the inference of target class membership based on properties of source instances. Conditions are represented in PROLOG-like rules, which are then directly forwarded into the FaCT reasoner.

The reclassification partially supports the object-oriented modelling of ontologies in the mapping representation, but because both approaches (rule-based and reclassification) have not been integrated, the overall solution, in general, and of the representation language, in particular, is rather limited.

4.2.3 RDFT approach

RDFT (Omelayenko, 2002) has been developed in the same period as the approach described in this paper, and has been, therefore, strongly considered for a representation language of semantic relations. Yet, its expressiveness and transformation capabilities are clearly insufficient. In fact, RDFT has been developed with the B2B catalogue integration application domain in mind, neglecting some important features such as 'property to concept' semantic relations. According to Omelayenko (Omelayenko 2002), such relations give raise to misunderstandings and should therefore be treated by procedural-programming based solutions. However, one of the most important limitations of RDFT is its capability to describe only rather simple regular-expression transformation of attributes.

Despite these important limitations, RDFT has some positive features. One very important feature is the capability to represent not only relations at the conceptual level but also at the syntactic level, thus supporting the so called normalisation sub-phase to the ontology mapping process (Silva and Rocha, 2003a). In particular, RDFT provides constructs to translate from DTD's and XSD's into RDFS models and vice-versa. These capabilities are more functional than conceptual though, since in Omelayenko and Fensel (2001) suggest a two-layer approach, not only one as the capabilities of RDFT would suggest.

4.2.4 OntoMerge approach

In the scope of OntoMerge (Dou et al., 2002, 2003), authors argue that ontology mapping is better understood if thought in terms of ontology merging. Ontology merging consists in defining a new ontology (the merged ontology) by the union of both source and target

ontologies entities and bridging axioms (semantic bridges in current terminology) representing the semantic relations between source and target ontologies entities. The merged ontology is a new ontology which entities are new representations of the original ontologies. As a consequence, the merged ontology is it self a fully fledged ontology and can be part of other merging process.

The Web-PDDL language, a Lisp-like, strongly typed, first order logic language is used to represent ontologies and bridging axioms. Despite some extra construct that has been added to the language, Web-PDDL is automatically processable by the in-house developed OntoEngine reasoner, responsible for the 'ontology translation'.

The simplest bridging axiom corresponds in current terminology to the 'concept to concept' semantic bridge. For every pair of semantically related ontology concepts, a new concept is defined in the merged ontology and two bridging axioms are specified, bridging each ontology concept with the new one.

Authors argue that the presented approach permits automatic translation in both directions, but this feature is possible only in a few very special cases. Moreover, it has been noticed that some very simple but ordinary transformations are not supported in OntoMerge due to the use of the generic reasoner. For example, it would be very hard (if not impossible) to semantically relate a relation to an attribute holding the number of source relations of an instance (corresponds to the SQL Count command).

Yet, probably the most important limitation of the OntoMerge is the completeness of the execution, since, like most theorem-provers, OntoEngine does not guarantee completeness. To overcome this problem, a very simple (even naive) heuristic rule is used, based on the sizes of the left and right bridging axioms.

$$W_1 \times \text{size of conclusion} - W_2 \times \text{size of premise.}$$

Concerning that a new ontology arises from the process, as described by the authors, the advantages depend on several factors, such as the size of the overlap of components ontologies and how these sizes change as they are merged. Considering the scale, heterogeneity and dynamicity of the web, these characteristics are unpredictable and therefore these solutions tend to be disregarded in favour of other less centralised and immediate approaches.

4.2.5 Outlook

Other research works and respective approaches have been analysed, such as MOMIS (Beneventano et al., 2001; Bergamaschi et al., 1999) in the database integration field and Clio (Miller et al., 2000) in application of views for information integration. However, their associated representation languages do not correspond to the identified requirements.

OntoMerge is definitely a very good approach in ontology mapping. It is based on private operational components (OntoEngine) constituting a unique solution. A similar situation is observed with MOMIS (the integration systems), ODB-tools (the DL and inference system) and the ODL_I^3 (representation language). RDFT is far the less capable of the approaches, but its simplicity and semantic web awareness had been a good source of inspiration and comparison. The work by Stuckenschmidt and colleagues lacks the combination of rule and reclassification approaches.

Table 2 compares the described approaches with the driving vectors identified in Section 2.2. Notice that not only characteristics of the representation language are considered in this table, but the overall approach capabilities and limitations.

Table 2 Support of described projects to defined requirements

Requirements	*Protégé*	*Stuckenschmidt et al.*	*RDFT*	*OntoMerge*	*Envisaged support*
Applicability	+	−	−	+	+
Semantic expressivity	+	+	−	+	+
Automation	None	None	None	None	+
Modularisation	+	−	−	+	+
Reutilisation	−	+	−	+	+
Declarativity	+	+	+	+	+
Semantic web-awareness	After 2003	−	+	+	+

One of the most important remarks arising from this table is the fact that none of the approaches provide automation of the semantic relations specification. It is tempting to argue that no connection exists between the automation of the mapping process and the representation of semantic relations. However, one might argue that the more features the language has, the less automatic support it provides in comparison to the system possibilities. Empirically another reflection arises naturally: why not focus on the relations allowed and processable by the system. In fact, representation languages should be able to represent all but the semantic relations the underlying system can process. Manual mapping experiences support this hypothesis by confirming that users tend to relate entities according to a limited set of 'well-known' types of transformation functions (e.g. copy, concatenation, split or table translation), instead of a large and unlimited set of possibilities. According to such a limited set, the user tries to associate source and target ontologies, entities with one of the transformation functions.

Therefore, instead of analysing the representation language according to its larger capabilities, the representation language should be analysed by its limitations and constraints. What is more, the representation language should be perfectly declared and represent all but the system transformation capabilities. Addressing the problem under this perspective, the representation language determines the semantic relation possibilities, and as a consequence the searching space is reduced.

5 Semantic bridging ontology

SBO is the MAFRA companion respecting description and representation of ontology mappings. SBO has been previously presented in work by Maedche et al. (2002a) using UML notation. Using an abstract, mathematical notation, this section formally presents SBO, explicitly defining its semantics and coping with misunderstandings and ambiguities. Additionally, grounding it to mathematical notation promotes syntax independence.

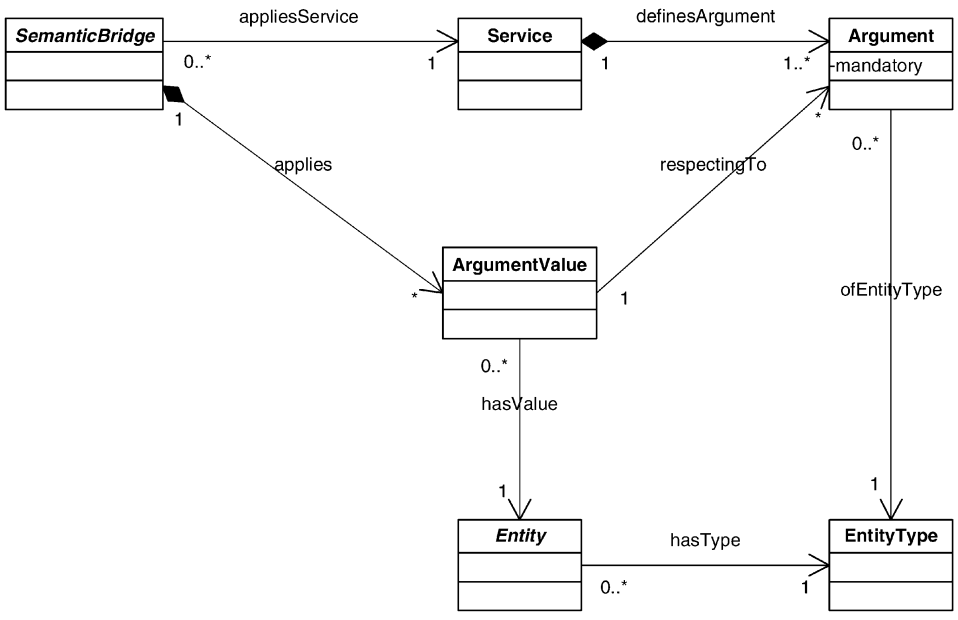
SBO is an ontology of the ontology mapping domain of knowledge. It specifies, classifies and describes the types of ontology mapping relations, interrelates them and provides other modelling constructs necessary to express ontology mapping documents. Since ontologies conceptual entities are the objects to be mapped, SBO is in fact a meta-ontology.

In essence, SBO is composed of two fundamental distinct but complementary concepts:

- SemanticBridge class, which represents the semantic relation between a set of source ontology entities and a set of target ontology entities – it encompasses all information necessary to its interpreted at execution time
- Service class, which represents the transformation capabilities present in the ontology mapping system.

Because the SemanticBridge concept does not contain transformation capabilities in itself, and is not able to define the process required to transform source ontologies entities into target ontology entities, it is necessary to associate to each SemanticBridge one specific Service responsible for those roles. The semantics and signature of each Service are defined and described according to the source and target arguments and respective characteristics (e.g. the EntityType). The ArgumentValue class correlates the ontologies entities (i.e. Entity instances) with the Service's Arguments (Figure 2).

Figure 2 UML representation of the SemanticBridge and Service conceptual relation



Notice that Entity class does not concern to ontologies entities only, but also entities composed by the ruled association of ontologies entities. In turn, EntityType class corresponds to both the ontology entities and the other composed entities (refer to Section 5.5). The clear separation of competences has three main advantages:

- it allows the separate evolution of the semantic bridge upon the entities and transformation dimensions
- it allows the enhancement of services in both operation and arguments, with minimal or no consequences to the SemanticBridges specification
- it allows the incorporation of new functionalities into Services with no consequences to the SemanticBridges specifications – this possibility will be further exploited in other phases of the ontology mapping process, as described in Section 6.

The SBO part described so far answers, to a certain extent, the requirements raised by the Transformation and Cardinality dimensions of semantic relations (Table 1). While it is not able to support all the other requirements, it provides a supportive starting point to further and finer characterisation of other SBO entities.

5.1 Semantic bridge

The SemanticBridge concept conveys all necessary information from the source semantic bridging phase to the execution phase. According to the semantic relations characterisation (Section 4.2), the SemanticBridge concept is formally defined as:

$$B := \left(E^s, E^t, Q, S, \phi^s, \phi^t, \delta^s, \delta^t, K\right)$$

where:

- E^s and E^t are respectively the source and target set of Entities, where Entities are the instances of *EntityTypes* as referred above
- Q is a set of constants elements. Every constant element is either a single constant (Literal) or an Array of Literals (refer to Section5.5)
- S represents the Service applied in the SemanticBridge
- φ^s and φ^t are the functions that associate, respectively, the source and target entities to the transformation Service arguments (each entity can be associated with more then one argument):

$$\phi^s : E^s \rightarrow 2^{Args^s} \setminus \{\varnothing\}$$
$$\phi^t : E^t \rightarrow 2^{Args^t} \setminus \{\varnothing\}$$

  This function corresponds to the ArgumentValue class's instances of Figure 2. Because Service is responsible for the definition and characterisation of its own arguments, and because a SemanticBridge is related to only one Service, the cardinality of the SemanticBridge is in fact stated by the associated Service

- δ^s and δ^t are the relations that associate constants with, respectively, source and target arguments of the transformation Service:

$$\delta^s : Q \rightarrow 2^{Args^s} \setminus \{\varnothing\}$$
$$\delta^t : Q \rightarrow 2^{Args^t} \setminus \{\varnothing\}$$

- K is a set of ConditionExpressions, which constrain the execution of the SemanticBridge according to the knowledge base instances.

The formal definition of ontology, presented in Section 3, and the analysis of the entity type dimension of semantic relations (Section 4.2), clearly distinguish between concept and property entities. Following the same approach, the SemanticBridge class is further specialised into ConceptBridge and PropertyBridge classes.

5.2 Concept bridge

ConceptBridge class represents the semantic relation between source and target ontology concepts. Semantically, this bridge means that each instance of the source concept gives rise to a new instance of the target concept. The ConceptBridge specialises the SemanticBridge concept in two ways:

- The cardinality of a ConceptBridge is always 1:1, which means that exactly one source concept is semantically related to exactly one target concept. In order to semantically relate the same source concept with many different target concepts, multiple ConceptBridges are defined (Silva and Rocha, 2003b);
- The transformation process executed in transforming concept instances into target concept instances is always the same: CopyInstance. As a consequence, the transformation Service specification in ConceptBridges can be made implicit.

Accordingly, a ConceptBridge is formally defined as:

$$B^C := (c^s, c^t, Q, \delta^s, \delta^t, K)$$

where $c^s \in C^s \subseteq E^t$, $E^s \in C^t \subseteq c^t$ *and* Q, δ^s, δ^t *and* K *are defined as for the SemanticBridge.*

5.3 Property bridge

PropertyBridge class represents the semantic relation between sets of source and target ontologies properties. Semantically, this bridge means that the set of source properties instances are transformed into a set of target properties instances. Unlike ConceptBridge, the transformations occurring between source and target properties vary enormously. As a consequence, the associated Service has to be explicitly stated in the PropertyBridge. The PropertyBridge is formally defined as the tuple:

$$B^P := (L^s, L^t, Q, S, \phi^s, \phi^t, \delta^s, \delta^t, K)$$

where: $L^s \subseteq E^t$ is the set of source ontology Paths (refer to Section 5.5), $L^t \subseteq E^s$ is the set of target ontology Paths and Q, S, φ^s, φ^t, δ^s, δ^t and K are defined as for SemanticBridge.

Notice that the associated Service is implicitly responsible for several characteristics of the bridge, in particular:

- The type of the Entities related through the bridge. A concatenation Service requires string attributes from both source and target ontologies, while a Service that would create relations between target instances requires the target ontology's entities to be relations (at least).
- The cardinality of the PropertyBridge. For example, the concatenation Service states a n:1 cardinality while the split Service states 1:n cardinality.

Unlike ConceptBridge, that directly applies ontologies concepts as their arguments, PropertyBridges apply ontologies properties in the context of a certain ontology domain (a concept) and range (a concept or an attribute).

5.4 Other constructs

Some extra constructs are available in SBO as referred to in previous sections:

- Paths, which allow address in bridging phase and query the knowledge base in execution phase, whose result is a particular view over the knowledge base data. Path is a non-empty list of valid relations between concepts.
- ConditionExpressions provide the mechanism to specify semantic constraints upon SemanticBridges in the bridging phase, instantiated with ontology entities instances in the execution phase and evaluated accordingly. ConditionExpression is a Boolean expression of comparison expressions, which in turn compares instances values with instances values or constants (Literals) through a variety of comparison operators
- Arrays allow the generalisation of Services according to the cardinality of both the service and the arguments. An array is an ordered set of mapping entities of the same type, but unlike sets in which order does not matter, or in a list in which elements are typically accessed successively, in array the order matters and elements are accessed in an unpredictable fashion, as might be required by specific Services. Concepts, Paths, ConditionExpressions and Literals can be grouped into arrays and applied in the SemanticBridge as a single argument to the Service.

Furthermore, SBO describes taxonomic relations and other non-hierarchical interrelations between semantic bridges, constraining their meaning and application and, consequently, which improve their semantics. This is the subject of the next section.

5.5 Ontology mapping document

Interrelations and constraints between SemanticBridges are of fundamental importance to the rest of the paper. These interrelations are defined in the context of the ontology mapping document, which is formally defined as:

$$M := (O^s, O^t, T, B^C, B^P, A^{B^C}, A^{B^P}, \perp^{B^C}, \perp^{B^P}, \Diamond, \prec)$$

where:

- O^s is the source ontology
- O^t is the target ontology
- T is the set of available transformation Services

- B^C is the set of ConceptBridges holding between O^s and O^t concepts
- B^P is the set of PropertyBridges holding between O^s and O^t defined Paths
- A^{B^C} is the set of containers, named AlternativeBridges-of-ConceptBridges that group together mutually disjoint ConceptBridges
- A^{B^P} is the set of containers, named AlternativeBridges-of-PropertyBridges, that group together mutually disjoint PropertyBridges
- $\perp^{B^C}: A^{B^C} \rightarrow 2^{B^C}$ is the function that associates ConceptBridges with AlternativeBridges-of-ConceptBridges
- $\perp^{B^P}: A^{B^P} \rightarrow 2^{B^P}$ is the function that associates PropertyBridges with AlternativeBridges-of-PropertyBridges
- $\Diamond: B^C \rightarrow 2^{B^P \cup A} B^P$ is the function that relates PropertyBridges with ConceptBridges
- $< \subseteq B^C \times B^C$ is a reflexive, acyclic, anti-symmetric and transitive relation between ConceptBridges, which corresponds to the hierarchical relation between ConceptBridges, referred to as 'sub bridge of'.

The $\Diamond$ function influences the properties allowed in PropertyBridges. In fact, properties that are not accessible from the concepts semantically related in the ConceptBridge cannot be semantically related in such a context (ConceptBridge), i.e. if PropertyBridges are $\Diamond$-related with a certain ConceptBridge, it is mandatory that the Paths defined in PropertyBridges all have the ConceptBridge's concepts as root concept. This constraint is formally represented as the following conjunction:

$$cb \coloneqq \left(c^s, c^t, Q_{cb}, \delta^s_{cb}, \delta^t_{cb}, K_{cb}\right) \in B^C$$
$$pb \coloneqq \left(L^s, L^t, Q, S, \phi^s, \phi^t, \delta^s, \delta^t, A_{pb}, K_{pb}\right) \in B^P$$
$$\forall L^s \in L^s \quad \exists s^s = entry(L^s, 1) \quad \Diamond(cb, pb) \Rightarrow subject(s^s, c^s)$$
$$\wedge$$
$$\forall L^t \in L^t \quad \exists s^t = entry(L^t, 1) \quad \Diamond(cb, pb) \Rightarrow subject(s^t, c^t)$$

Additionally, the $\Diamond$ function further constraint target Paths to those whose length is 1:

$$\forall L \in L^t \quad length(L) = 1$$

Despite not representing a conceptual requirement but instead an implementation decision, four important advantages on the application of this constraint have been identified:

- Execution order of SemanticBridges is irrelevant. If a multi-step target Path is specified, it would be necessary to previously execute the PropertyBridges that create the relation between instances.
- As a consequence, it prevents recursive dependencies between SemanticBridges.
- Promotes modularisation, systematisation and error detection, respecting ConceptBridges. If the application of a multi-step target Path is believed necessary, then it is necessary to $\Diamond$-relate the PropertyBridge with a ConceptBridge, such that:

$$\forall L \in L^t \quad \text{tConcept}(cb) = \text{subject}(L, \text{length}(L))$$

and change Paths, such that:

$$\forall L \in L^t \quad L := [s \mid s := \text{entry}(L, \text{length}(L))]$$

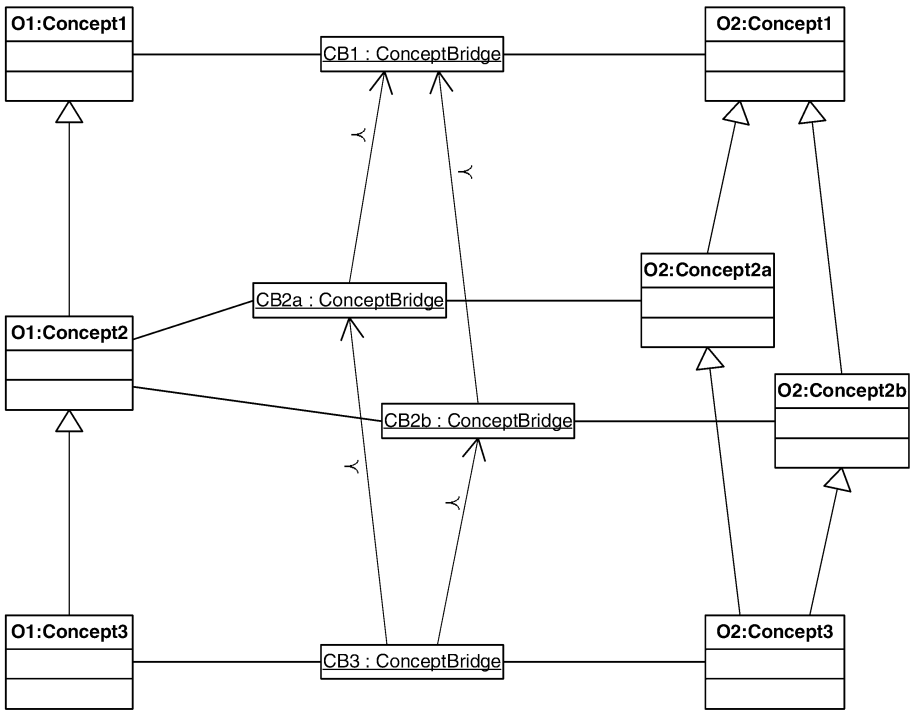
- Evolution procedures are simplified. Since fewer dependencies exist between SemanticBridges, fewer number of propagation changes arise between SemanticBridges.

In order to apply the $\prec$ (subBridgeOf) relation between two ConceptBridges, any ConceptBridge may be defined as a subBridgeOf another ConceptBridge if its source and target concepts are sub concepts (*is_a*) of respectively the source and target concept of the envisaged super-bridge. This constraint is formally defined by:

$$\begin{aligned}
&\forall cb_1, cb_2 \in \mathcal{B}^{\mathcal{C}} \\
&\qquad \prec (cb_2, cb_1) \\
&\Rightarrow \\
&\qquad \text{sConcept}\left(cb_1, c_1^s\right) \wedge \text{tConcept}\left(cb_1, c_1^t\right) \wedge \text{sConcept}\left(cb_2, c_2^s\right) \wedge \text{tConcept}\left(cb_2, c_2^t\right) \wedge \\
&\qquad \left(\left(is_a\left(c_2^s, c_1^s\right) \wedge is_a\left(c_2^t, c_1^t\right)\right) \vee \left(is_a\left(c_2^s, c_1^s\right) \wedge c_1^t == c_2^t\right) \vee \left(c_1^s == c_2^s\right) \wedge is_a\left(c_2^t, c_1^t\right)\right)
\end{aligned}$$

Figure 3 presents an ontology mapping scenario, in which the hierarchical relation between ConceptBridges ($\prec$) is adopted according to the hierarchy of the ontology concepts.

Figure 3 Example of hierarchical relations between ConceptBridges, according to the ontologies concepts hierarchy



The fundamental consequence of the $\prec$ relation is the inheritance by sub-bridges, of the PropertyBridges $\Diamond$-related with super-bridges. As so, such PropertyBridges are automatically and implicitly $\Diamond$-related to its sub-bridges. In execution phase, super bridge's PropertyBridges are executed in scope of sub-bridges as any explicitly $\Diamond$-related PropertyBridge.

6 Service-oriented approach

At the beginning of the previous section, the Service concept was introduced and interrelated with the SemanticBridge. Services are responsible for the transformation of instances, and are formally described as a tuple in the form of:

$$S \coloneqq (Location, Args^s, Args^t, \alpha^s, \alpha^t)$$

where:

- *Location* corresponds to the information needed to locate and access the transformation Service capabilities during every phase of the ontology mapping process
- $Args^s$ and $Args^t$ are respectively the set of source and target Arguments composing the signature of the Service
- α^s is the function that associates an EntityType with each and every source Argument

$$\alpha^s : EntityTypes \rightarrow Args^s$$

- α^s is the function that associates an EntityType with each and every target Argument

$$\alpha^t : EntityTypes \rightarrow Args^t$$

- *EntityTypes* is the set of entity types allowed in Services, which is the union of: ontology concepts; Paths; Literals (constants); Arrays of Paths; and Arrays of Literals.

For example, consider the CopyAttribute Service, capable of copying instances of a source ontology attribute into instances of a target ontology attribute:

$$S_{CopyAttribute} \coloneqq \begin{pmatrix} \text{"pt.ipp.isep.gecad.mafra.services.transformations.CopyAttribute"}, \\ Args_1^s, Args_1^t, \alpha_1^s, \alpha_1^t \end{pmatrix}$$

$$Args_1^s \coloneqq \{sourceAttribute\}$$

$$Args_1^t \coloneqq \{targetAttribute\}$$

$$\alpha_1^s \coloneqq \{attributePath(sourceAttribute)\}$$

$$\alpha_1^t \coloneqq \{attributePath(targetAttribute)\}$$

Table 3 presents some other examples of services and their arguments currently available in MAFRA Toolkit, in a more concise form.

Table 3 Some MAFRA Toolkit transformation functions and their interface

Service/Argument ID	*Type*	*Comment*
CopyInstance: creates target concept instances for each source concept instance that fulfils the conditions. This is the service implicitly associated with ConceptBridges		
Source concept	Class	Source ontology class whose instances will be transformed
Extensional specification	ArrayOfConditions	xtensional definition of source class Einstances
Target concept	Class	Target ontology class to create
CopyRelation: creates a relation between concepts instances based		
Source path	Path	Source ontology path for each path the bridge will be executed
Extensional specification	ArrayOfConditions	Extensional definition of source class instance
Target path	RelationPath	Target ontology path to create
CopyAttribute: copies (no changes) the source property value (string) to the target property		
Source attribute	AttributePath	Source ontology attribute whose instances will be copied
Target attribute	AttributePath	Target ontology attribute to copy to
CountRelations: counts the number of instances of a relation		
Source path	Path	Source ontology path to copy
Extensional specification	ArrayOfConditions	Extensionally defines source class instance
Target attribute	AttributePath	Target ontology attribute to create
Split: splits by separators, the literal in source attribute		
Source attribute	AttributePath	Source ontology attribute whose instances will be splited
Separators	ArrayOfLiterals	Literals or regular expressions to split by
Target attributes	ArrayOfAttributePaths	List of attributes to instantiate with splited values
Concatenation: concatenate several attribute values each one separated by a literal		
Source attributes	ArrayOfAttributePath	List of source attributes whose instances will be concatenated
Separators	ArrayOfLiterals	Literals to concatenate between attribute values
Target attribute	AttributePaths	Target attributes to instantiate with concatenated values

Table 3 Some MAFRA Toolkit transformation functions and their interface (continued)

Service/Argument ID	*Type*	*Comment*
TableTranslation: translates attributes according to a file-based table of relations		
Source attribute	AttributePath	Source ontology attribute whose instances will be translated
File location	FilePath	The location of the file containing the table
Target attribute	AttributePath	Target attributes to instantiate with translated value

Services are entities external to the core of the ontology mapping system. They are dynamically pluggable, self-described entities, primarily responsible for the transformation capabilities of the ontology mapping system, and are therefore the primary constraint in the definition of SemanticBridges. The number and type of entities are defined in respect to the Service associated with the SemanticBridge. As consequence, it is possible to validate the SemanticBridge specification based on the associated Service.

Such an approach is commonly adopted in other domains, such as programming languages compiling and linking processes, in which the interface of a procedure or function is used to (limitedly) determine the program correctness.

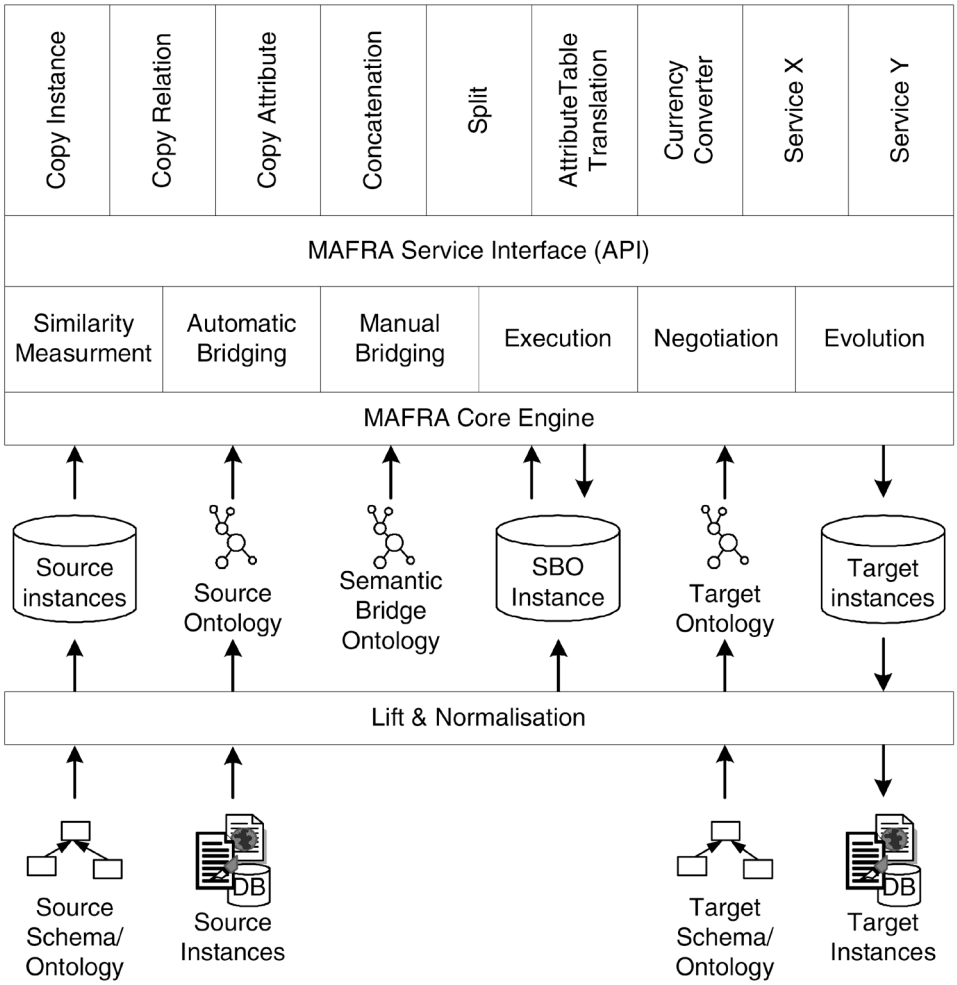
However, two simple observations suggested the improvement of the Service concept:

- it is virtually impossible to provide all possible transformation requirements using a monolithic set of transformation capabilities
- the transformation results of ontology mapping execution largely depends on the transformation Services available and associated with SemanticBridges.

Furthermore, as referred to in the final remarks about the related work (Section 4.2.5), the transformation capabilities of the system must be clearly and formally defined in order to facilitate and promote the automation of the process.

We suggest, therefore, the extension and improvement of the Service concept so it is able to support not only the transformation process but also other phases in the ontology mapping process (Silva and Rocha, 2003a). Hence, instead of support and encompassing competencies for a unique part of the process, Services are requested to support and encompass competencies for multiple phases, arising therefore as a multidimensional entity, representing expertise and common-sense know-how in the overall ontology mapping process. This approach is reflected in the adoption of a modular, decentralised architecture, where Services are attached to the system functional core modules (i.e. bridging, execution, negotiation, evolution, etc) (Silva and Rocha, 2003a) through the specific API referred as the MAFRA Service Interface (Figure 4). In each phase of the process, different competencies of the Service are used by each core system modules. Except those competencies concerning the characterisation of the Service interface and respective execution, Services are free to provide or withhold competencies. In this case generic functions are applied.

Figure 4 MAFRA service-oriented system architecture



Besides the adoption of this approach in the similarity measuring and semantic bridging phases, as described in next section, we envisage extending the Service competencies to several other phases of the process, especially to the evolution and negotiation of semantic bridges.

7 Automatic bridging

Semantic bridging is a highly subjective time-consuming task, demanding extensive domain expertise. Even on a small scale, any automation or driving support is of great help. Adoption of the multidimensional service-oriented architecture to this problem is based on the premise that each Service knows what and in which circumstances it can be successfully applied. Accordingly, we suggest centring in each Service available in the

system the competencies to decide which entities are semantically related to each other. Similarity measuring, the stage of the ontology mapping process prior to the semantic bridging phase, aims to capture similarities between pairs of source and target ontologies entities. Such a task is very subjective because it deals with two other very subjective, domain and user dependent interpretations of a knowledge domain.

Referring to lexical tools like dictionaries, specific domain thesaurus and WordNet, it is possible to multi-classify similarities between pair of entities. However, these classifications are typically insufficient and error-prone. Structural analyses of ontological entities and statistical and probabilistic analyses of ontologies instances are other insufficient similarity measuring techniques. In fact, according to literature (Bernstein et al. 2001; Doan et al., 2002; Rahm and Bernstein, 2001) and analysing respective results, it is empirically evident that no technique is by itself accurate enough. In that sense, we propose to use an open set of techniques and combine their results according to each Service specificity and specification. Each similarity measuring technique is implemented by an external, pluggable entity connected to the core system by a common specific interface, just as described for Services.

Each of these entities is referred to as a Matcher, and its outcome is a set of so-called Match in the form of:

$$Match := (Source\ ontology\ entity, Target\ ontology\ entity, Matcher, Similarity\ value, Justifications)$$

where:

- *Matcher* is the identification of the Matcher that performed the similarity measure
- *Similarity value* corresponds to the evaluated similarity measure, normally ranging from 0 to 1
- *Justifications* is a set of statements providing argumentation for the evaluated value – currently, justifications are not being used, but their application is envisaged in the negotiation part of the process, where autonomous entities must share information conducting to an agreement about the mapping.

Because multiple matchers are supposedly available in the system, multiple matches may be evaluated for the same pair of ontology entities and the same entity (source or target ontology entity) might be present in several pairs, giving rise to an n:m relation. Each match is not a semantic relation, but it denotes a certain level of a certain type of similarity between the source and the target ontology entity.

For example, consider the ontology mapping scenario of Figure 5 where ontology O1 is to be semantically related to ontology O2.

Consider also the four available matchers:

- word sense disambiguation-based matcher, based on the Resnik method (Resnik, 1999)
- WordNet hyponyms-based matcher that evaluate the lexically equivalent relation of the *is_a* ontological relation
- structural matcher, based on clustering techniques suggested in MOMIS work (Bergamaschi et al., 1999).

Figure 5 Simple ontology mapping scenario using UML notation

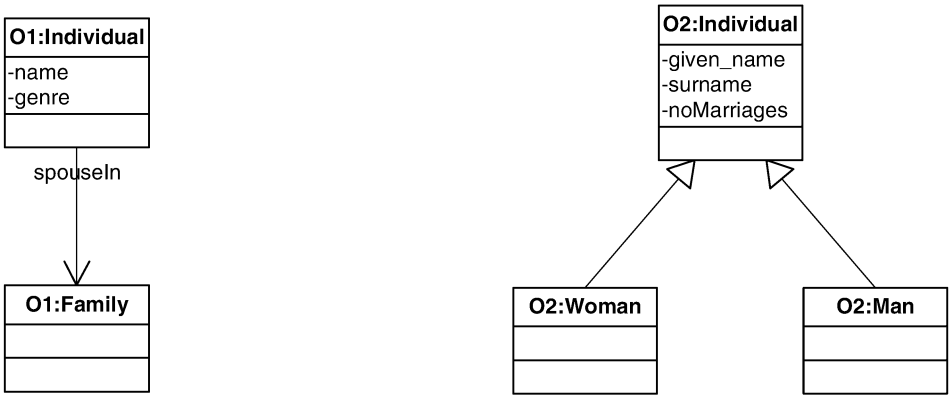


Table 4 represents the matches resulting from the similarity-measuring phase whose value is above a certain predefined global threshold. Notice that at matching level, ontology entities are treated independently of their type, which means concepts might match with properties and vice-versa.

Table 4 Example of matches resulting from the similarity measuring phase

ID	*Source entity*	*Target entity*	*Matcher*	*Similarity value*	*Justifications*
m1	Individual	Individual	Resnik-like	0.95	[]
m2	Individual	Man	Resnik-like	0.86	[]
m3	Individual	Woman	Resnik-like	0.86	[]
m4	name	given_name	Resnik-like	0.82	[]
m5	name	surname	Resnik-like	0.82	[]
m6	spouseIn	noMarriages	Resnik-like	0.66	[]
m7	name	given_name	Hyponyms-based	1	[]
m8	name	surname	Hyponyms-based	1	[]
m9	Individual	Individual	MOMIS-like	0.78	[]
m10	Individual	Man	MOMIS-like	0.78	[]
m11	Individual	Woman	MOMIS-like	0.78	[]

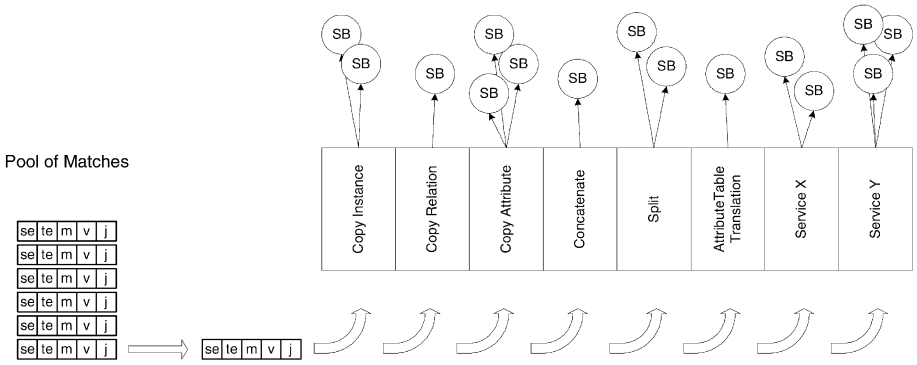
While matches are a 1:1 relation between two ontology entities, SemanticBridges cardinality varies according to the associated Service. As a consequence, matches are to be combined in a way Services are able to accept and process them. Due to the multitude of open sets of Services, a centralised rule-based system would not be feasible. In fact, Services are very different from each other, having different requirements on the type and number of necessary ontology entities.

We aim to develop a (semi-) automatic system in which Services are the first responsible for the decision to combine any set of matches into a SemanticBridge. Later,

the user can redo or refine the resulting mapping. The proposed process consists in pushing every match to every available Service (Figure 6) that must be able to decide if the match:

- should be added to an already existent (provisory) SemanticBridge, in which case entities are attached to the respective arguments
- is relevant for the creation of a new (provisory) SemanticBridge, in which case a new provisory SemanticBridge is created: the Service responsible for this decision is associated with the SemanticBridge and the match entities are attached to the respective arguments
- invalidates an already existent SemanticBridge, in which case the provisory SemanticBridge is deleted.

Figure 6 Figurative representation of the automatic bridging process



Services must therefore determine the requirements that matches must conform with. In particular, Services are requested to specify:

- the number and respective types of arguments, which are mandatorily defined by the Service interface (refer to Section 6 and Table 3)
- the type of matches considered by the Service, which corresponds to define the *Matcher* element in the match
- the similarity value threshold, below which the matches are not pushed to the Service – the threshold can be defined in respect to certain type of match or generically
- the rule or expression responsible for the decisions.

For example, one might define the characteristics shown in Table 5 for the CopyInstance, Split, CopyAttribute and CountRelations Services.

Table 5 Extended specification of services for the automatic bridging process

Service	*Considered matches types*	*Threshold*	*Rule*
CopyInstance	Resnik-like	0.7	Pair of Resnik-like and MOMIS-like matches must exist between source and target concept
	MOMIS-like	0.7	
Split	Resnik-like	0.5	Pairs of Resnik-like and MOMIS-like matches must exist between source and target attributes
	MOMIS-like	1	
CopyAttribute	Resnik-like	0.8	Pair of Resnik-like and MOMIS-like matches must exist between source and target attribute
	MOMIS-like	0.8	
CountRelations	Resnik-like	0.75	Pair of Resnik-like and Type-checker matches must exist between source relation and target attribute
	Type-checker	1	

In order to illustrate the proposed methodology, consider the ontology mapping scenario presented in Figure 5 and the respective matches presented in Table 4. The CopyInstance, CopyAttribute, Split and CountRelation Services are available.

Because every PropertyBridge is $\Diamond$-related to a ConceptBridge, ConceptBridges are defined prior to PropertyBridges, just like in the user-based specification (Silva and Rocha, 2003b). Therefore, all matches are first pushed to the ConceptBridge associated Service (CopyInstance). Due to the CopyInstance interface, only matches relating source and target concepts are pushed. In this example, the pairs (m1, m9), (m2, m10) and (m3, m11) represent necessary and sufficient conditions to create three provisory ConceptBridges (hypothesis). In this particular case, no further match is acceptable or denies the hypothesis, thus ConceptBridges are confirmed:

$$Indiv2Indiv := (O1 : Individual, O2 : Individual, \{\}, \{\}, \{\})$$
$$Indiv2Woman := (O1 : Individual, O2 : Woman, \{\}, \{\}, \{\})$$
$$Indiv2Man := (O1 : Individual, O2 : Man, \{\}, \{\}, \{\})$$
$$B_1^C := \{Indiv2Indiv, Indiv2Man, Indiv2Woman\}.$$

Because the constraint presented in Section 5.5 holds between *Indiv2Indiv* and both *Indiv2Man* and *Indiv2Woman* the *subBridgeOf* ($\prec$) relation is specified between these bridges as follow:

$$\prec_1 := \{(Indiv2Man, Indiv2Indiv), (Indiv2Woman, Indiv2Indiv)\}.$$

It is now the moment to push matches to the rest of Services. The Split Service requires: a set of matches, whose cardinality is greater than 1; the source and target entities are attributes; the source entity is common to all matches; and matches fulfil the requirements specified in Table 4. The set of matches composed by m4, m5, m7 and m8 conform to

the previous rules and because no match contradicts the hypothesis, a PropertyBridge with the associated Split Service is created and confirmed. Respecting the constraints from section 5.5, and exploiting the facilities provided by the $\prec$ relation, PropertyBridge is defined in scope of *Indiv2Indiv* only:

$$name2names := \left(L_1^s, L_1^t, \{\ \}, Split, \phi_1^s, \phi_1^t, \{\ \}, \{\ \}, \{\ \}, \{\ \}\right)$$
$$L_1^s := \left\{L_{1.1}^s\right\}$$
$$L_{1.1}^s := O1 : Individual / name / Literal$$
$$L_1^t := \left\{O2 : Individual / given\_name / Literal, O2 : Individual / surname / Literal\right\}$$
$$\phi_1^s := \left\{(L_{1.1}^s, \{sourceAttribute\})\right\}$$
$$\phi_1^t := \left\{(L_1^t, \{t\arg etAttributes\})\right\}$$
$$\Diamond\left(Indiv2Indiv, \{name2names\}\right)$$
$$B_1^P := \left\{name2names\right\}.$$

The CopyInstance Service requirements are: a set of matches, whose cardinality is 1; the source and target entities are attributes; and matches fulfilling the requirements specified in Table 4. The set composed by the pair (m4, m7) conform to these requirements and consequently a provisory PropertyBridge is created. The set composed by the pair (m5, m8) conform to these requirements and once again, a new provisory PropertyBridge is created. Therefore, previous pairs are mutually contradictory, which invalidates the provisory bridges. In fact, despite the fact that such situations might occur, it is unusual that the same source attribute is semantically equivalent (CopyAttribute) to two distinct target attributes. Hence, the automatic bridging rules reject the provisory PropertyBridges.

Respecting the CountRelation Service, the requirements are: set of matches, whose cardinality is 1; the source entity is a relation and the target entity is an attribute; and matches fulfil the requirements specified in Table 4. Notice that the CountRelation Service requires a type-checker matcher. This matcher is responsible for checking if the output of Service corresponds to the type of target entity. In this particular case, either the matcher does not exist or the matches evaluated to a similarity value below the global threshold. Accordingly, no PropertyBridge is suggested for this Service, even if one is expected between spouseIn and noMarriages.

8 Experiences

The MAFRA Toolkit has been adopted as the development, representation and transformation engine in the Harmonise project (Harmonise, 2003). This project intends to overcome the interoperability problems occurring between major tourism operators in Europe. Problems arise due to the use of distinct information representation languages like XML and RDF, and different business and information specifications, like those provided by SIGRT (SIGRT, 2003), by TourinFrance (TourinFrance, 2003) and by FTB (Finnish Tourist Board, 2003). Harmonise uses an 'Interoperability Minimum Harmonisation Ontology' as *lingua franca* between agents. The use of the MAFRA Toolkit in this type

of centralised system is not contradictory though. In fact, either adopting centralised or peer-to-peer information integration, a one to one mapping specification between repositories is necessary, in which MAFRA has proven to be useful and effective.

MAFRA is responsible for the acquisition, representation and execution of the ontology mapping between each agent specific ontology and IMHO. IMHO describes the tourism domain in about 64 concepts, 120 attributes and 213 interrelations between concepts. IMHO and partner ontologies are very different. For example, the FTB information schema specifies one concept with 48 attributes and SIGRT defines about 50 concepts with typically less then ten properties. Even if many and different semantic and syntactic mismatches occur in mapping these ontologies, no conceptual limitations have been detected in the MAFRA Toolkit, and only a few refinements of the prototypal mapping services have been required.

Currently, the MAFRA Toolkit is being applied in the Artemis and Satine European funded projects. Artemis (Artemis, 2004) aims to integrate medical information conforming distinct standards, which is a very important test case to MAFRA approaches due to the required accuracy of medical information systems. Satine (2004) aims to promote automation in the interconnection of small and medium enterprises through web services in e-commerce, specially in the domain of travel businesses.

Concerning performance issues, some simple observations have been made. Considering the lack of reports describing experiences using ontology mapping tools, the report contained in Dou et al. (2003) constitutes a simple but valuable reference. In that work, authors report the experience in transforming a dataset of 21,164 instances respecting the Gedcom ontology (Gedcom, 2003), into instances respecting the Gentology ontology (Gentology, 2003). These are two very similar ontologies, whose mapping requires only simple semantic relations. The ontology mapping developed using the MAFRA Toolkit specifies the semantic relations presented in the previously mentioned report and others harvest from the examples found in the OntoMerge web pages. No distinctions have been detectable from both transformations. Ontologies are represented in DAML, which is not directly supported by the MAFRA Toolkit. However, a representation translator from DAML to RDFS is available, which transforms ontologies in a few seconds. Dataset is represented in RDF, thus excusing any transformation in MAFRA Toolkit execution. On the contrary, OntoMerge requires transformations of both ontologies and dataset. OntoMerge reports a 22 min execution time in a Pentium III at 800 MHz, while the MAFRA Toolkit achieved the same results in less than 2 min in a Pentium II at 350 MHz. If a Pentium 4M 2.0 Mhz is used, MAFRA requires less then 1 min. Recently, the OntoMerge authors reported the same test execution in a similar time as MAFRA Toolkit, which seems a more reasonable scenario.

9 Conclusion

The work described in this paper has been developed within the SANSKI project, in which semi-automatic ontology mapping technology is developed to increase interoperability between artificial agents operating in the semantic web. Several domains of research are addressed in this project but, in this particular paper, the focus is the automatic specification and representation of semantic bridges between ontologies entities.

Ontology mapping and Semantic Bridging Ontology (SBO) have been formally presented which promote a univocal understanding of the process and representation. SBO represents our conceptualisation of the ontology mapping problem, describing the inputs, outputs, components and their dependencies and constraints. A service-oriented architecture of SBO suggests that semantic relations are ultimately dependent on the type of transformations allowed in the system. In that sense, the domain expert know-how is acquired and integrated in the system not as one monolithic structure but multiple modules evolving and adapting separately and according to the domain expert requirements. The system architecture of the MAFRA Toolkit implements this feature allowing the easy integration of additional transformation Services and their self-description.

In this paper, we propose to extend this approach to other phases of the process so transformation Services are responsible not only for the transformation, but for other phases of the process. In particular, we deem it possible to exploit this approach in the automation of the bridging phase. For that, we propose to extend the service concept such that it encompasses other features related to the bridging process. In particular, every Service is responsible for the characterisation of necessary and sufficient conditions in which certain sets of sources and target ontology entities have a good probability of being semantically related.

Our current efforts have two main priorities. The first concerns the improvement of capabilities of Services to choose and decide the SemanticBridges. We consider that this goal is achieved by providing new matchers and refining the service specifications automatically. For that, we envisage the exploitation of machine learning techniques, such that the system would be capable of observing the user behaviour and finding the circumstances in which the user actions fit in. According to the constraints found, the system would propose to update the Services specifications.

The second effort relates to the application of the multidimensional service-oriented architecture to other phases of the process. Currently, the most promising area is the evolution of the mapping document according to the ontology changes. The multidimensional service-oriented architecture provides a good starting point in the sense that once again the competence and know-how to deal with specific changes are forwarded to Services and not encompassed in monolithic centralised rules.

Acknowledgements

This work is partially supported by the Portuguese MCT-FCT project POCTI/2001/GES/41830. Many thanks to Alexander Maedche for his ideas and support during my stay at FZI. Thanks to Oliver Fodor for his feedback on the application of MAFRA Toolkit in the Harmonise project and to Siyamed Sinir for his efforts in the application of the MAFRA Toolkit in the Artemis and Satine projects. Thanks to Jorge Santos for his revisions and many valuable discussions.

References

Artemis (2004). Available from: http://www.srdc.metu.edu.tr/webpage/projects/artemis

Beneventano, D., Bergamaschi, S., Guerra, F. and Vincini, M. 'The MOMIS approach to information integration', *Proceedings of the International Conference on Enterprise Information Systems*, Setúbal, Portugal, pp.194–198.

Bergamaschi, S., Castano, S. and Vincini, M. (1999) 'Semantic integration of semistructured and structured data sources', *SIGMOD Record*, Vol. 28, No. 1, pp.54–59.

Bernstein, P.A. and Rahm, E. (2001) *On Matching Schemas Automatically*, Redmond, WA, USA: Microsoft Research, MSR-TR 2001-17.

Critchlow, T., Ganesh, M. and Musick, R. (1998) 'Automatic generation of warehouse mediators using an ontology engine', in A. Borgida, V. Chaudhri and M. Staudt (Eds) *5th Workshop KRDB-98*, Seattle (WA), USA, pp.81–88.

Crubézy, M. and Musen, M.A. (2003) 'Ontologies in support of problem solving', in S. Staab and R. Studer (Eds) *Handbook on Ontologies*, Springer-Verlag, Heidelberg, Germany, pp.321–341.

Doan, A., Madhavan, J., Domingos, P. and Halevy, A. (2002) 'Learning to map ontologies on the semantic web', in *World-Wide Web Conference*.

Dou, D., McDermott, D. and Qi, P. (2002) 'Ontology translation by ontology merging and automated reasoning', in *EKAW Workshop on Ontologies for Multi-Agent Systems*, Sigüenza, Spain, pp.3–18.

Dou, D., McDermott, D. and Qi, P. (2003) 'Ontology translation on the semantic web', in *International Conference on Ontologies, Databases and Applications of Semantics*, Catania (Sicily), Italy, pp.952–969.

Fensel, D., Benjamins, R., Motta, E. and Wielinga, B. (1999) 'UPML: a framework for knowledge system reuse', in *International Joint Conference on Artificial Intelligence*, Stockholm, Sweden, pp.16–23.

Finnish Tourist Board (2003) http://www.mek.fi

Fodor, O., Dell'Erba, M., Ricci, F., Spada, A. and Werthner, H. (2002) 'Conceptual normalisation of XML data for interoperability in tourism', in *Workshop on Knowledge Transformation for the Semantic Web (KTSW 2002) at ECAI'2002*, Lyon, France, pp.69–76.

Gedcom (2003) http://www.daml.org/2001/01/gedcom/gedcom.daml

Gennari, J.H., Tu, S.W., Rothenfluh, T.E. and Musen, M.A. (1994) 'Mapping domains to methods in support of reuse', *International Journal of Human-Computer Studies*, No. 41, pp.399–424.

Gentology (2003) http://orlando.drc.com/daml/Ontology/Genealogy/3.1/Gentology-ont.daml

Gruber, T.R. 'Towards principles for the design of ontologies used for knowledge sharing', in N. Guarino and R. Poli (Eds) *Formal Ontology in Conceptual Analysis and Knowledge Representation*, Deventer, The Netherlands: Kluwer Academic Publishers.

Hammer, J. and Medjahed, B. (1993) 'An approach to resolving semantic heterogeneity in a federation of autonomous, heterogeneous database systems', *Journal for Intelligent and Cooperative Information Systems*, Vol. 2, No. 1, pp.51–83.

Harmonise (2003) www.harmonise.org

Horrocks, I. (1998) 'The FaCT system', *Automated Reasoning with Analytic Tableaux and Related Methods (Tableaux'98)*, Oisterwijk, Netherlands: Springer-Verlag, Heidelberg, Germany, pp.307–312.

Hsieh, D. (1990) 'A logic to unify semantic network knowledge systems with object-oriented database models', in *SRI International', SRI-CSL-90-15.*

Kalfoglou, Y. and Schorlemmer, M. (2003) 'Ontology mapping: the state of the art', *The Knowledge Engineering Review*, Vol. 18, No. 1, pp.1–31.

Maedche, A., Motik, B., Silva, N. and Volz, R. (2002a) 'MAFRA – a MApping FRAmework for distributed ontologies in the semantic web', in *Workshop on Knowledge Transformation for the Semantic Web at ECAI'2002*, Lyon, France, pp.60–68.

Maedche, A., Staab, S., Studer, R., Sure, Y. and Volz, R. (2002b) 'SEAL – tying up information integration and website management by ontologies', *IEEE Data Engineering Bulletin*, Vol. 25, No. 1, pp.10–17.

Miller, R.J., Haas, L.M. and Herněndez, M.A. (2000) 'Clio: schema mapping as query discovery', in A. El Abbadi et al. (Eds) *26th Very Large Database Conference*, Morgan Kaufmann, Cairo, Egypt, pp.77–88.

Mitra, P. and Wiederhold, G. (2001) 'An algebra for semantic interoperability of information sources', in N. Bourbakis (Ed.) *2nd. IEEE Symp. on BioInformatics and Bioengineering (BIBE'2001)*, IEEE Computer Society, Bethesda (MD), USA, pp.174–182.

Omelayenko, B. (2002) 'RDFT: a mapping meta-ontology for business integration', in *Workshop on Knowledge Transformation for the Semantic Web at ECAI'2002*, Lyon, France, pp.76–83.

Omelayenko, B. and Fensel, D. (2001) 'A two-layered integration approach for product information in B2B E-commerce', in K. Bauknecht, S.-K. Madria and G. Pernul (Eds) *Second International Conference on Electronic Commerce and Web Technologies*, Springer Verlag, pp.226–239.

Park, J.Y., Gennari, J.H. and Musen, M.A. (1998) 'Mappings for reuse in knowledge-based systems', *11th Workshop on Knowledge Acquisition, Modelling and Management*.

Rahm, E. and Bernstein, P.A. (2001) 'A survey of approaches to automatic schema matching', *The VLDB Journal*, Vol. 10, No. 4, pp.334–350.

Resnik, P. (1999) 'Semantic similarity in a taxonomy: an information-based measure and its application to problems of ambiguity in natural language', *Journal of Artificial Intelligence Research*, Vol. 11, pp.95–130.

Satine (2004) http://www.srdc.metu.edu.tr/webpage/projects/satine/

Sheth, S.A. and Larson, J.A. (1990) 'Federated database systems for managing distributed, heterogeneous, and autonomous databases', *ACM Computing Surveys*, Vol. 22, No. 3, pp.183–236.

SIGRT (2003) http://www.dgturismo.pt/irt/

Silva, N. and Rocha, J. (2003a) 'MAFRA – an ontology MApping FRAmework for the semantic web', in W. Abramowicz and G. Klein (Eds) *6th International Conference on Business Information Systems*, Colorado Springs (CO), USA.

Silva, N. and Rocha, J. (2003b) 'Semantic web complex ontology mapping', in *Web Intelligence*, Halifax, Canada, pp.82–88.

Stuckenschmidt, H. and Visser, U. (2000a) 'Semantic translation based on approximate re-classification', in *Workshop Semantic Approximation, Granularity and Vagueness at KR 2000*, Breckenridge (CO), USA.

Stuckenschmidt, H. and Wache, H. (2000b) 'Context modelling and transformation for semantic interoperability', in *International Workshop Knowledge Representation meets Databases at ECAI'2000*, Berlin, Germany, pp.115–126.

TourinFrance (2003) http://www.tourisme.gouv.fr

Visser, P.R.S., Jones, D.M., Bench-Capon, T.J.M. and Shave, M.J.R. (1997) 'An analysis of ontological mismatches: heterogeneity versus interoperability', in *AAAI Spring Symposium on Ontological Engineering*, Stanford (CA), USA.

W3C (2003) http://www.w3.org/TR/rdf-schema/