A USER-DRIVEN AND A SEMANTIC-BASED ONTOLOGY MAPPING EVOLUTION APPROACH

Hélio Martins and Nuno Silva
GECAD - Knowledge Engineering and Decision Support Research Group
School of Engineering – Polytechnic of Porto, Porto, Portugal

Keywords: Ontology, Ontology Evolution, Ontology Mapping, Ontology Mapping Evolution.

Abstract: Systems or software agents do not always agree on the information being shared, justifying the use of distinct ontologies for the same domain. For achieving interoperability, declarative mappings are used as a basis for exchanging information between systems. However, in dynamic environments like the Web and the Semantic Web, ontologies constantly evolve, potentially leading to invalid ontology mappings. This paper presents two approaches for managing ontology mapping evolution: a user-centric approach in which the user defines the mapping evolution strategies to be applied automatically by the system, and a semantic-based approach, in which the ontology's evolution logs are exploited to capture the semantics of changes and then adapted to (and applied on at) the ontology mapping evolution process.

1 INTRODUCTION

Ontologies are a key concept in knowledge-based systems in general and in the Semantic Web in particular (Berners-Lee et al., 2001). However, actors do not always agree on the information being shared, justifying the use of distinct ontologies, even if corresponding to the same domain of application. Information integration arises therefore as a core process in these application areas. To solve/minimize the interoperability problem, ontology mapping proved to be an efficient solution (Silva, 2004). 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 (Silva, 2004). The result of the ontology mapping specification process (at conceptual level) is an ontology mapping document containing the semantic relations. However, in dynamic environments, ontologies evolve, causing the ontology mapping document to become invalid. In heterogeneous environments where interoperability among systems depends essentially upon the ontology mappings, semantic relations must be adapted to reflect the ontologies evolution (Figure 1). The two original ontologies (*O1* and *O2*) are mapped through the *M* mapping document (more than two ontologies may exist in the integration scenario, but for the sake of simplicity, only two are considered here).

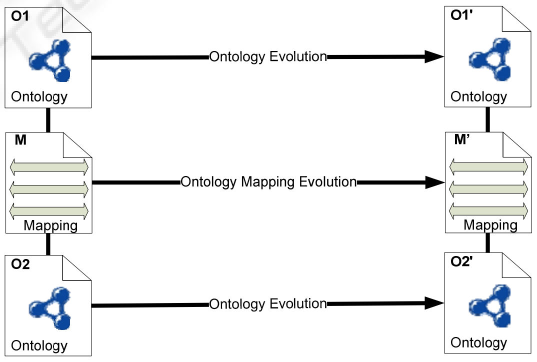

Figure 1: Problem definition scenario.

However, *O1* and/or *O2* may evolve, giving rise to *O1'* and *O2'* respectively. This process is named ontology evolution, and may cause inconsistencies in the ontology mapping document. To solve those inconsistencies, two solutions were identified: (i) generation of a completely new ontology mapping document, i.e. a new set of semantic relations between entities of versions of source/target ontologies and (ii) adapt the original ontology mapping document to the ontologies' evolution, i.e. correcting invalid semantic relation and generating new required semantic relations according to ontology changes, giving rise to a new ontology mapping document (M'). Considering all the difficulties and problems faced by the ontology mapping process (Euzenat, et al., 2007), it is worth considering the second approach.

The goal of this work is therefore to research and develop a solution for supporting the evolution and adaptation of the ontology mapping document according to the mapped ontologies' evolution.

In the reminder of the paper, we present the context of this work in Section 0. In Section 0 we present the foundations of our ontology mapping evolution process. In Section 0 a user-driven approach is presented, followed by the description of the semantic-based approach in Section 0. In Section 0 we describe the evaluation experiences. In Section 0, the related work and provide our concluding remarks in Section 0.

2 CONTEXT

2.1 Ontology Evolution

Ontology evolution can be defined as the timely adaptation of ontologies to changes which arises and the consistent management of these changes (Stojanovic, 2004). According to this definition, two key concepts were indentified: (i) ontology changes and (ii) consistency management.

Different levels of change abstraction are often used (Stojanovic, 2004), (Klein, 2004), (Plessers, 2005), which might be summarized into:

- Basic changes, that change one entity in the ontology model only (e.g. Removing a Concept from the ontology);
- Composite changes, that change more than one entity in the ontology (e.g. Copy a Concept).

These changes are part of an evolution ontology that conceptualizes all kind of changes that can be performed on the ontology, the relation between them and meta-information (e.g. date, owner of change request). During the ontology evolution process, the changes are stored in an instance of the evolution ontology, serving as ontology evolution log, e.g.:

```
<RemoveEntity
      rdf:ID="i-151"
      referencesConcept="http://s1.pt#C2">
<causesChange rdf:resource="#i-66"/>
<causesChange rdf:resource="#i-92"/>
<previousChange rdf:resource="#i-99"/>
</RemoveEntity>
```

Consistency defines the degree of uniformity, standardization, and freedom from contradiction among the parts of a system or component (IEEE, 1990). In (Stojanovic, 2004) the author states that ontology consistency can be considered as an agreement among ontology entities with the respect to the semantics of the underlying ontology language. This assumes special relevance because the execution of a single ontology change (e.g. removing a concept) may cause inconsistencies in other parts of the ontology (e.g. subClassOf relation between concepts). To solve these inconsistencies, many approaches use the concept of derived or deduced changes. Because deduced changes may result in additional deduced changes, this becomes an iterative process. As consequence, it is relevant to distinguish between the changes requested by the ontology engineer (representing their intentions) and the deduced changes (to solve inconsistencies). Notice that beyond the agreement of the semantics of the ontology language, ontology consistency is related with the ontology's purpose. Yet, this dimension is not addressed in this paper.

Complementarily, the concept of ontology evolution strategy proposed in (Stojanovic, 2004) assumes special relevance in the context of this work. The evolution strategy concept is used to give the user (i.e. the ontology engineer) the possibility to achieve a consistent ontology according to the semantics of the ontology language and a set of best-practices. This concept is based on:

- One resolution point, i.e. a dilemma that might occur during the resolution of changes;
- Elementary ontology evolution strategies, i.e. the potential ways for resolving one resolution point.

A pair between one resolution point and one elementary strategy is named ontology evolution strategy. Table 1 presents the resolution points and respective elementary evolution strategies that are useful for this work. For example, if a concept becomes orphan (i.e. no subClassOf relation remains for the concept), one may choose to: (i) delete the concept, reconnect the concept to the parent of the removed concept, or reconnect it to the root concept.

Table 1: Ontology evolution strategies.

Evolution Strategy	
Resolution Point	**Elementary strategy**
Orphaned concept	Delete Reconnect to parent Reconnect to root
Property propagation	Don't Propagate Propagate direct only Propagate all

Equivalent reasoning is made for the other resolution point. Evolution strategies will be discussed in section 0.

2.2 Ontology Mapping

Ontology mapping proved to be an efficient solution (Rahm, et al., 2001), (Euzenat, et al., 2007) for helping solve the interoperability problem. The IEEE dictionary (IEEE, 1990) defines interoperability as "the ability of two or more systems or components to exchange information and to use the information that has been exchanged". The ontology mapping process is not a trivial process, requiring a deep understanding of both ontology conceptualizations and their semantic similarities. The output of the ontology mapping process is an ontology mapping document containing semantic relations between source ontology entities and target ontology entities. In our context, an ontology mapping document is an instantiation of the SBO (Semantic Bridge Ontology) (Silva, 2004). The SBO is the ontology that describes the semantic relations holding between ontologies entities, providing not only a conceptualization mechanism but also a representation and exchange mechanism of semantic relationships between ontologies. It specifies, classifies and describes the types of ontology mapping relations, inter relates them and provides other modelling constructs necessary to express ontology mapping documents.

The SBO is fundamentally composed of the following entities:

- Concept Bridge is the semantic relation that maps ontologies' concepts. At transformation time, it will create target concept instances. For example: CB(O1:Person, O2:Individual), will create an instance of O2:Individual for each existing O1:Person;
- Property Bridge, maps source properties to target properties. At execution time, properties are created in the scope of the target instance. For example, PB({O1:firstName, O1:lastName},{O2:name},concat) will concatenate the value of O1:firstName and O1:lastName into the value of O2:name. The service (e.g. Concat, CopyAttribute) defines the arguments of the Property Bridge;
- hasBridge relation associates a PropertyBridge to a ConceptBridge. This relation provides the scope to the Property Bridge. Otherwise, the Property Bridge would not realize to which concept the property value is to be attached;
- subBridgeOf relation between ConceptBridges is the mapping equivalent to the ontological "subClassOf" relation.

The ontology mapping process is responsible for the instantiation of the SBO, generating a mapping document. Figure 2 illustrates an example of a mapping document.

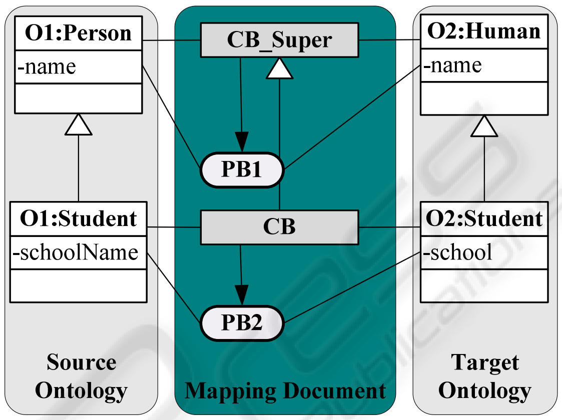

Figure 2: Example of ontology mapping document.

Once instantiated, the semantic bridges can be exploited at data level (instance level) for information transformation/exchange.

3 MAPPING EVOLUTION

Ontology mapping evolution is the process whereby entities of ontology mapping document are adapted with the eventual changes in the source and/or target ontologies, trying to preserve as much of the semantics of the original mapping relations as possible.

3.1 Two-phases Iterative Process

Changes to ontologies affect the ontology mapping in two ways: whether it is an addition or a removal. In the case of removal, existing semantic relationships are affected and must evolve. Instead, if adding new ontology entities, new semantic relations might be necessary.

Accordingly, the proposed ontology mapping evolution process comprises of two iterative phases:

- Correcting Invalid Entities, consists of identifying and correcting the invalid mapping entities according to the ontologies evolution, preserving as much as possible the semantics of the original mapping;

▪ Matching New Ontology Entities, consists of discovering and evaluating the similarity between entities and the specification of (correct) mapping relationships.

Matching new entities is closely related to the matching process (Euzenat, et al., 2007) and less (or nothing) with the mapping evolution process, thus will not be addressed further in this paper. Instead, the following sections concern the process of Correcting Invalid Entities.

3.2 The Semantics of SBO

In order to correct an invalid SBO entity, one has to identify it first. The entity is invalid when the SBO semantics are not obeyed. Depending on the SBO entity, different semantic restrictions apply.

An **invalid Concept Bridge** is one that has at least one invalid argument (Table 2).

Table 2: Concept Bridge's invalid conditions.

Invalid Argument	Causes
Source Concept	It is not a concept, or The concept value does not exist, or It is not a source ontology's concept
Target Concept	It is not a concept, or The concept value does not exist, or It is not a target ontology's concept
Conditions	An invalid ontology entity is used
Extensional spec.	An invalid condition.

An **invalid Property Bridge** is one that has at least one invalid argument according to the applied service (Table 3).

Table 3: Property Bridge's invalid conditions.

Invalid Argument	Reason
Source Argument	It is not a source ontology entity, or It doesn't respect the service's interface
Target Concept	*idem*
Conditions	An invalid ontology entity is used
Extensional spec.	An invalid condition.

By analyzing the SBO mapping document and respective ontologies it is possible to identify the invalid mapping entities.

3.3 Ontology Mapping Changes

The list of changes of ontology mapping largely depends on the ontology mapping language.

Similar to other evolution approaches such as object-oriented schema evolution proposed in (Banerjee, et al., 1987) and ontology evolution approaches (Klein, 2004) and (Stojanovic, 2004), and according to the semantics of the SBO, a set of ontology mapping changes were defined. In order to improve the process, two abstraction levels of ontology mapping changes were defined.

Elementary Ontology Mapping Changes. Represent single changes in one SBO entity only. Both additive and subtractive changes are considered (Table 4). These changes form the backbone of the mapping evolution system, in the sense that they represent the modification at the lowest level of complexity.

Table 4: Extract of elementary ontology mapping changes.

#id	Elementary Ontology Mapping Changes
1	AddConceptBridgeSourceConceptValue
2	RemoveConceptBridgeSourceConceptValue
3	AddConceptBridgeTargetConceptValue
4	RemoveConceptBridgeTargetConceptValue
5	RemoveHasBridge
6	AddHasBridge

Composite Ontology Mapping Changes. Represent changes as the combination of two or more elementary ontology mapping changes, organized as one logical unit that should be executed as a whole. Composite changes (Table 5) allow a higher level of abstraction, representing a semantic context of the mapping evolution process, because they group together a set of meaningful and complementary (elementary) changes, representing a semantic change.

Table 5: Extract of composite ontology mapping changes.

Composite Ontology Mapping Changes	#id Elem. Change
ChangeConceptBridgeSourceConceptValue	1,2
ChangeConceptBridgeTargetConceptValue	3,4
ChangePropertyBridgeHasBridge	5,6

3.4 Sorting the Invalid Entities List

Once the list of invalid entities in the mapping is populated, it is necessary to sort it. The order in which entities are processed is very important. In fact, considering that entities are invalid because of others, and because correcting one may correct others, choosing the one that minimizes changes is very important.

Figure 3 depicts the order in which the entities are processed.

Because a Property Bridge is always defined in the context of a Concept Bridge, Concept Bridges must always be corrected first, so respective Proper-

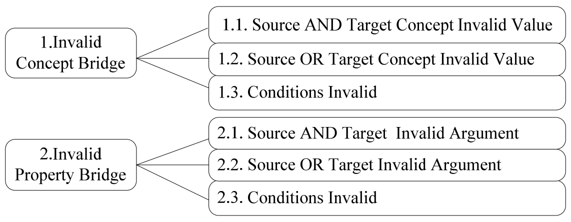
1.Invalid Concept Bridge
1.1. Source AND Target Concept Invalid Value
1.2. Source OR Target Concept Invalid Value
1.3. Conditions Invalid
2.Invalid Property Bridge
2.1. Source AND Target Invalid Argument
2.2. Source OR Target Invalid Argument
2.3. Conditions Invalid

Figure 3: Order of invalid entities to process.

ty Bridges are adapted accordingly. For example, changing a Concept Bridge source concept argument implies updating their Property Bridges' arguments.

Additionally, for each Concept Bridge there is also an established order of correction: first those with invalid source and target concept values, then those who have only one invalid concept value, and later, those with invalid conditions.

Once all Concept Bridges are addressed and respective Property Bridges' arguments are updated, invalid Property Bridges are addressed next.

By following this order, the process avoids unnecessary computation and minimizes ontology mapping changes, maximizing consistency and maintaining the original semantics.

It is now possible to start correcting the invalid entities. The next section describes a semi-automatic user-driven approach, in which the user decides, from a list prepared by the system, the corrective action to execute. The Section 0 describes a completely automatic approach based on the semantics of the ontology evolution information.

4 USER-DRIVEN APPROACH

The correction process is potentially ambiguous because several corrective mapping changes are allowed for the same invalid entity (Table 6).

Table 6: Invalid mapping situations and corrective mapping changes.

Invalid Mapping Situation	**Corrective mapping changes**
Source AND Target Concept	Remove Concept Bridge;

Source XOR Target Concept	Remove Concept Bridge;
	Change Concept Bridge Concept Value (to super concept);

Source AND Target Argument	Remove Property Bridge;

Source XOR Target Argument	Remove Property Bridge;
	Change Property Bridge Argument Value;

Extensional Specification	Remove Concept/Property Bridge;
	Remove Invalid Condition;

Generic Condition	Remove Concept/Property Bridge; Remove Invalid Condition.

However, as previously mentioned, the execution of a single ontology mapping change (e.g. remove a Concept Bridge) may cause inconsistencies in others mapping entities (e.g. subBridgeOf and hasBridge relations). Thus, aside from the original change, some other changes might need to be performed to solve such problems.

In fact, two of the corrective mapping changes identified in Table 6, potentially lead to ambiguous situations (notice that the first invalid mapping situation in Table 6 is not ambiguous):

- Removing a Concept Bridge, triggers the removal of the *hasBridge* relationships, generating orphaned Property Bridges (ambiguous situation). Additionally, triggering the removal of *subBridgeOf* relationships lead to orphaned Concept Bridges;
- Changing Concept Bridge Concept Value may cause inconsistencies in the Property Bridge's argument values, because the bridged properties may not have the new concept value in their domain. As a consequence, one has to decide how to manage the Property Bridges (ambiguous situation).

These ambiguities promote the concept of **mapping evolution strategy**, similar to that used in the scope of ontology evolution process. A mapping evolution strategy is a pair between (i) one mapping resolution point (ambiguous situation or a dilemma that occurs during the ontology mapping process and) and (ii) one elementary mapping evolution strategy (a set of possible ways to solve a resolution point). Table 7 summarizes the resolution points and respective corrective strategies.

Table 7: List of ontology mapping resolution points and respective corrective strategies.

Resolution Point	**Elementary Mapping Strategy**

Orphaned Property Bridge	Don't Propagate any Property Bridge (Remove Property Bridges);
	Propagate direct Property Bridges to sub Concept Bridges;
	Propagate (All) Property Bridges to sub Concept Bridges.

Orphaned Concept Bridge	Delete orphaned Concept Bridge;
	Reconnect orphaned Concept Bridge to parent;
	Keep orphaned Concept Bridge.

For each mapping resolution point, one or many elementary mapping strategies are possible.

The concept of Mapping Evolution Strategy serves as parameterization of the system. In other words, users choose how to solve each ambiguous situation either on a local basis (i.e. the user decides every time a resolution point occurs) or global (i.e. the user decides to apply always a specific strategy for every specific type of resolution point).

5 SEMANTIC-BASED APPROACH

Consider the ontology evolution scenario depicted in Figure 4, in which a "RemoveConcept" change has been applied to the target concept T:C. While it is obvious that the CB Concept Bridge has to evolve, the changes to apply depend on the user decision, and might not be the best solution.

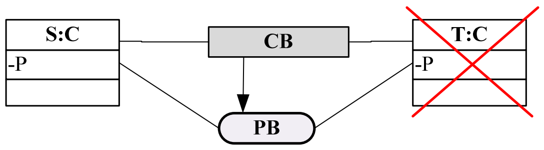

Figure 4: Close-up of ontology evolution and mapping.

Because the ontology evolution strategies capture the user's intentions during that process, we claim that they provide useful information for the ontology mapping evolution process.

The goal is then to automatically identify the best ontology mapping evolution strategy, based on the ontology evolution information provided. This should be understood not as a group of atomic changes (e.g. RemoveConcept), but as well-structured semantic information that fully captures the ontology engineer intentions. To exploit such information the concept of ontology evolution strategy described in section 0 will be used. Unfortunately, the chosen evolution strategy is not explicitly registered in the log file, but only the elementary changes, inter-related through the "consequent" relation. For example, consider Figure 5 as a wider perspective of the scenario presented in Figure 4, in which are represented not only the original "RemoveConcept" change, but also the deduced changes:

```
RemoveConcept(T:C)
RemoveSubClassOf(T:C,T:CSuper)
RemoveSubClassOf(T:CSub,T:C)
AddSubClassOf(T:CSub,Root)
RemoveDomain(T:C,P)
AddDomain(T:CSub,P)
AddDomain(T:CSub,PSuper)
```

The set of deduced changes provides evidence of the application of a specific ontology evolution strategy. From these, it is possible to generalise the

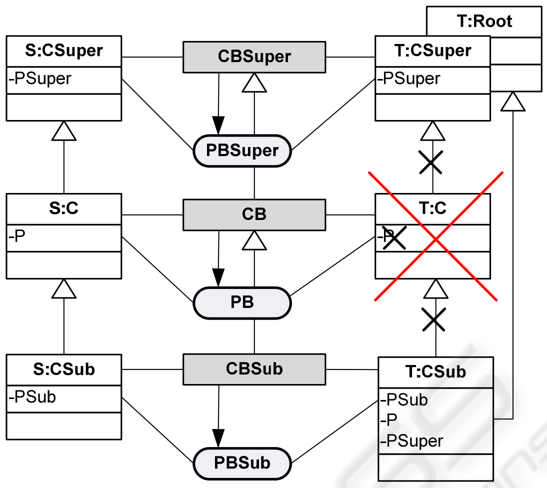

Figure 5: Example of ontology evolution and mapping.

necessary and sufficient conditions for the identification of a specific ontology evolution strategy.

Two resolution points may occur as a consequence of the "RemoveConcept" change: Orphaned Concept and Property Bridge Propagation. The orphaned concept resolution point is addressed by three elementary strategies: Reconnect to Root, Reconnect to Parent and Delete Concept (Table 1).

The necessary and sufficient conditions (i.e. the deduced changes) for the identification of the "Reconnect Orphaned Concepts to Root" strategy are defined in the next SWRL-like rule:

```
RemoveConcept(?TC)^
RemoveSubClassOf(?TC,?TCSuper)^
RemoveSubClassOf(?TCSub,?TC)^
AddSubClassOf(?TCSub,ROOT) ->
ReconnectOrphanedConceptToRoot(?TC).
```

The RemoveConcept change potentially gives rise to "Orphaned Property" resolution points too, which in turn are resolved by three elementary resolution strategies. In the case when the "Propagate All Properties" strategy is applied (as in the case depicted in Figure 5), the conditions/deduced changes are specified as:

```
ReconnectOrphanedConceptToRoot(?TC)^
Domain(?TC,?P)^
RemoveDomain(?TC,?P)^
AddDomain(?TCSub,?P)^
AddDomain(?TCSub,?PSuper) ->
PropagateAllProperties(?P).
```

For every invalid semantic bridge, the ontology mapping evolution process tries to identify the ontology evolution strategy adopted. When at least one strategy is identified, the process applies a set of ontology mapping changes that solve the invalid mapping context.

The set of changes to apply was previously identified as the best resolution process for the ontology context considering the user intentions applied at the ontology evolution. For every pair <invalid mapping entity, ontology evolution strategy> a set of ontology mapping changes are defined. For example, the pair <invalid concept bridge, ReconnectOrphanedConceptToRoot>, is addressed by the following rule:

```
1   InvalidConceptBridge(?CB) ^
2   TargetConceptValue(?CB,?TC) ^
3   not(Concept(?TC)) ^
4   ReconnectOrphanedConceptToRoot(?TC)^
5   SourceConceptValue(?CB,?SC) ^
6   SubBridgeOf(?CB,?CBSuper) ^
7   SubBridgeOf(?CBSub,?CB) ^
8   TargetConceptValue(?CBSuper,?TCSuper)->
9   not(TargetConceptValue(?CB,?TC) ^
10  TargetConceptValue(?CB,?TCSuper) ^
11  not(SubBridgeOf(?CBSub,?CB)).
```

The predicate in line 1 identifies the invalid concept bridge (?CB). Line 2 through 4 determines that the target concept (?TC) value is missing and that it has been removed and the "reconnect orphaned concept to root" strategy has been applied to its sub concepts. Line 5 through 8 instantiate several variables (e.g. ?CBSuper is instantiated with all super bridges of the invalid concept bridge). The right hand side of the rule defines the mapping evolution changes. Line 10 and 11 changes the invalid concept bridge's target concept value from ?TC to ?TCSuper. Line 11 removes the subBridgeOf relationship between ?CBSub and ?CB.

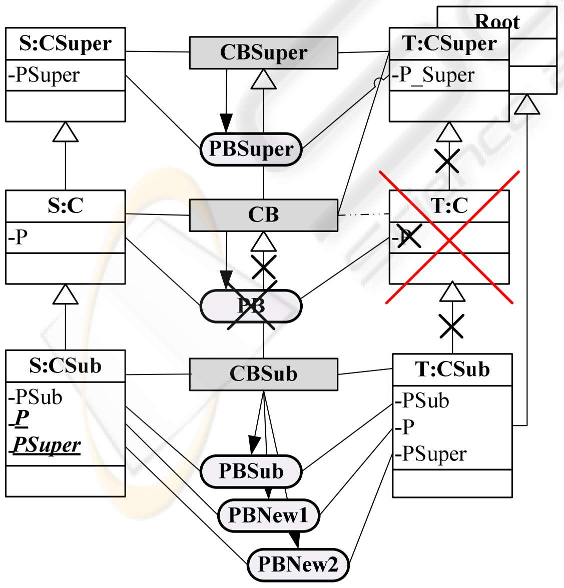

Figure 6: Mapping scenario example evolved.

Similar expressions are defined for invalid source concept and the remaining evolution strategies.

Consider the mapping scenario depicted in Figure 5. As a consequence of removing concept T:C, both the "Reconnect Orphaned Concept To Root" and the "Propagate All Properties" strategies were applied. In response to these evolution changes, the corresponding rules were executed, giving rise to the mapping scenario depicted in Figure 6.

6 EVALUATION

In respect to the user-based approach, no evaluation has been made because when dealing with user-defined strategies, the results are biased and should not be evaluated by experience or test cases since the results are subjective (Stojanovic, 2004).

In respect to the second approach, we claim that because the mapping evolution changes captured in the rules, mimic the changes adopted during the ontology evolution process, they are (at least indirectly) dependent on the user-defined strategies. However, because there is a gap between the ontology evolution intensions and those at the ontology mapping evolution, we conducted some tests based on two ontologies and four mappings. In fact, despite there being no ontology evolution logs currently available in the community, the research team internally carried out the evolution of two ontologies in the knowledge domain of R&D: (O1, O1'), (O2, O2') with 15-25 concepts (e.g. researcher, publication) with 5-10 properties each. The ontology evolution process adopted global strategies. Two ontology engineers with upper intermediate proficiency level in ontology mapping, manually-mapped each ontology pair M1(O1,O2), M2(O1-O2'), M3(O1'-O2), M4(O1'-O2') for reference, resulting in eight mapping documents.

The ontology mapping evolution process adopted global strategies. We then applied the automatic semantic-based ontology mapping evolution approach to both M2 and to one of the M4 mappings. The obtained results were very similar to the manually-specified mappings, but based on differences, some minor refinements in the mapping evolution rules were made. These refinements were then included in the current version of the system. Later, we applied the same approach to the remainder mappings (i.e. both M3 and the other M4). The results were again very similar but not equal because users refined one of the mappings with a local strategy. However, if that change was not done, the results would be equal.

7 RELATED WORK

Handling ontology mapping documents is a relative new research field, but similar problem has been tackled in other contexts such as object oriented (Banerjee, et al., 1987) or incremental view maintenance (Ceri, et al., 1991). These approaches present a taxonomy of changes, its semantics and a set of deterministic rules. ToMAS (Velegrakis, et al., 2003) is a tool for automatically detecting and adapting invalid or inconsistent mappings due to changes in either schemas or theirs constraints, even if the changes did not make any of the mappings syntactically incorrect. ToMAS also exploits knowledge about user choices that is embodied in the existing mappings. Further, while the changes are not restricted to atomic type schema element, the approach is schema centric and does not exploit the semantics of the changes requested by the users.

Our approach adopts ideas from these approaches but it goes a step further by exploiting the evolution log, automating the process.

8 CONCLUSIONS

In this paper, the ontology mapping evolution problem has been characterized in terms of goals, inputs, constraints and outputs, and an abstract ontology mapping evolution process has been defined, comprised of two-iterative phases respecting the removed and new entities in the ontology, respectively.

Based on the semantics of SBO, a set of invalid conditions and respective corrective changes were identified. Yet, because different potential solutions exist for the same invalid situation, a decision has to be made. A user-driven approach has been developed based on the structure and semantics of the SBO. This approach guaranties that the mapping document is corrected, but the user has to decide which actions to take for every ambiguous situation. Besides the list of possible corrective changes, no further support is provided to the user.

Instead, the proposed semantic-based approach automatically suggests the best corrective strategy, based on the log information provided by the ontology evolution process. The evaluation performed showed that this is a valid and useful approach, but further extensive evaluation has to be carried out. However, this evaluation is not easy due to the lack of logs.

A limitation of the semantic-based approach concerns with the blind application of rules, based on the necessary and sufficient conditions. In fact, it is possible that an ontology evolution strategy has been applied, even if only for some. Deciding whether the strategy has been applied or not, is an open issue. Additionally, because ontology mapping largely depends on the ontology mapping language, generalizing the proposed approaches to other ontology mapping (or alignment) languages is a big challenge for future work.

ACKNOWLEDGEMENTS

The work described in this document is partially funded by the MCTES-FCT funded EDGAR project (POSI/EIA/61307/2004) and the Coalesce project (PTDC/EIA/74417/2006).

REFERENCES

Banerjee, Jay, et al. 1987; Semantics and implementation of schema evolution in object-oriented databases. SIGMOD Rec. Ca,US: 1987, Vol. 16, pp. 311-322.

Berners-Lee, Tim, Hendler, James e Lassila, Ora. 2001; The Semantic Web. Scientific American. 2001, Vol. 284, pp. 34-43.

Ceri, Stefano e Widom, Jennifer. 1991; Deriving production rules for incremental view maintenance. In VLDB. 1991, pp. 577-589.

Euzenat, Jérôme and Shvaiko, Pavel. 2007. Ontology Matching. Berlin: Springer, 2007.

IEEE. 1990; Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries. N.Y, U.S.: 1990.

Klein, Michel. 2004; Change Management for Distributed Ontologies. 2004. PhD Thesis.

Plessers, P., De Troyer, O. 2005; Ontology Change Detection using a Version. The Semantic Web – ISWC. s.l. : Springer, 2005.

Rahm, Erhard and Bernstein, Philip A. 2001; A survey of approaches to automatic schema matching. In VLDB. s.l. : Springer, 2001.

Silva, Nuno. 2004. Multi-Dimensional Service-Oriented Ontology Mapping. 2004. PhD Thesis.

Stojanovic, Ljiljana. 2004. Evolution, Methods and Tools for Ontology. 2004. PhD Thesis.

Velegrakis, Yannis, Miller, Renée J. and Popa, Lucian. 2003. Mapping Adaptation under Evolving Schemas. VLDB. Berlin: 2003.