MAFRA — A MApping FRAmework for Distributed Ontologies

Alexander Maedche[1], Boris Motik[1], Nuno Silva[1,2], and Raphael Volz[1]

[1] Forschungszentrum Informatik at the Univ. Karlsruhe,
D-76131 Karlsruhe, Germany
http://www.fzi.de/WIM
{maedche,motik,silva,volz}@fzi.de
[2] ISEP Instituto Superior de Engenharia,
Instituto Politecnico do Porto, Portugal
http://www.dei.isep.ipp.pt

Abstract. Ontologies as means for conceptualizing and structuring domain knowledge within a community of interest are seen as a key to realize the Semantic Web vision. However, the decentralized nature of the Web makes achieving this consensus across communities difficult, thus, hampering efficient knowledge sharing between them. In order to balance the autonomy of each community with the need for interoperability, mapping mechanisms between distributed ontologies in the Semantic Web are required. In this paper we present MAFRA, an interactive, incremental and dynamic framework for mapping distributed ontologies.

1 Introduction

The current WWW is a great success with respect to the amount of stored documents and the number of users. However, the ever-increasing amount information on the Web places a heavy burden of accessing, extracting, interpreting and maintaining information on the human users of Web. Tim Berners-Lee, the inventor of the WWW, coined the vision of Semantic Web, providing means for annotation of Web resources with machine-processable metadata providing them with background knowledge and meaning (see [2]). Ontologies as means for conceptualizing and structuring domain knowledge are seen as the key to enabling the fulfillment of the Semantic Web vision.

However, the de-centralized nature of the Web makes indeed inevitable that communities will use their own ontologies to describe their data. In this vision, ontologies are themselves distributed and the key point is the mediation between distributed data using mappings between ontologies [16]. Thus, complex mappings and reasoning about those mappings are necessary for comparing and combining ontologies, and for integrating data described using different ontologies. Existing information integration systems and approaches (e.g., TSIMMIS [6], Information Manifold [8], Infomaster[1], MOMIS[2], Xyleme [3]) are "centralized" systems of mediation between users and distributed data sources, which exploit mappings between a single mediated schema and

[1] http://infomaster.stanford.edu/infomaster-info.html
[2] http://sparc20.ing.unimo.it/Momis/
[3] http://www.xyleme.com

schemas of data sources. Those mappings are typically modelled as views (over the mediated schema in the local-as-view approach, or over the sources schemas in the global-as-view approach) which are expressed using languages having a formal semantics. For scaling up to the Web, the "centralized" approach of mediation is probably not flexible enough, and distributed systems of mediation are more appropriate.

Building on this idea and on existing work, we introduce MAFRA, an Ontology MApping FRAmework (MAFRA) for distributed ontologies in the Semantic Web. Within MAFRA we provide an approach and conceptual framework that provides a generic view onto the overall distributed mapping process. In particular, in this paper we focus on representation and execution aspects of mappings. However, the proposed framework offers support in all parts of the ontology mapping life-cycle.

Organization of this paper. In section 2 we introduce the underlying conceptual architecture of MAFRA. In section 3 we focus on mapping representation and present the current status of our semantic bridging ontology and discuss its features. Section 4 presents the realized mapping implementation within KAON - an ontology and Semantic Web application framework[4]. Before we conclude a short discussion of related and future work is given in section 5.

2 Conceptual Framework

An ontology mapping process, as defined in [14], is the set of activities required to transform instances of a source ontology into instances of a target ontology. By studying the process and analyzing different approaches from the literature we observed a set of commonalities and assembled them into the MAFRA conceptual framework, outlined in Figure 1. The framework consists of five horizontal modules describing the phases that we consider fundamental and distinct in a mapping process. Four vertical components run along the entire mapping process, interacting with horizontal modules.

2.1 Horizontal Dimension of MAFRA

Within the horizontal dimension, we identified following five modules:

Lift & Normalization. This module focuses on raising all data to be mapped onto the same representation level, coping with syntactical, structural and language heterogeneity [19]. Both ontologies must be normalized to a uniform representation, in our case RDF(S), thus eliminating syntax differences and making semantics differences between the source and the target ontology more apparent [14]. This lift process is not further elaborated in this paper - we shall simply assume that the source and target ontologies are already represented in RDF-Schema with their instances in RDF. Also one essential step of this first phase is normalization. Three distinct ordered tasks are performed in our approach: *(i)* tokenization of the entities, *(ii)* elimination of resulting stop words and *(iii)* expansion of acronyms. The result is a list of normalized lexica.

[4] http://kaon.semanticweb.org

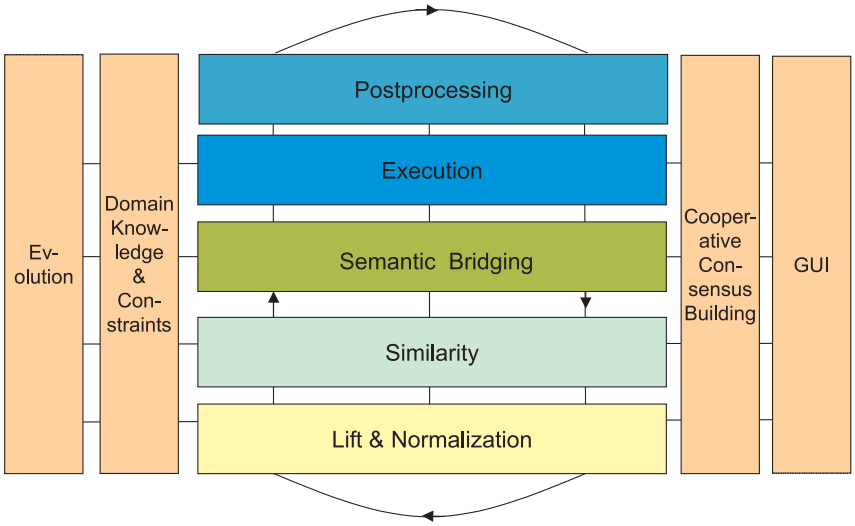

Fig. 1. Conceptual Architecture

Similarity. This module establishes similarities between entities from the source and target ontology, thus, it supports mapping discovery. Several different similarity measures have been proposed in literature [14, 3, 5, 10, 1].

We adopted a multi-strategy process (similar to [5]), that calculates similarities between ontology entities using different algorithms. The first strategy focuses on acquiring a *lexical similarity* between each entity in source entity with each and all entities in target entity. For that WordNet and an altered Resnik algorithm [15] are used. Subsequently, a next step calculates the so called *property similarity*, that is responsible to acquire the similarity between concepts based on their properties, either attributes or relations. The *bottom-up similarity* intends to propagate the similarity (or dissimilarity) from lower parts of the taxonomy to the upper concepts. It uses the property similarity as input and propagates the values to the top. This similarity gives a good overall view of similarity between taxonomies. Complementarily, the *top-down similarity* propagates similarities from top to bottom, and assumes special relevance when top level concepts have a higher or lower similarity. A detailed description and an evaluation of our similarity measures and the overall discovery module is provided in a companion paper [17].

Semantic Bridging. Based on the similarities computed in the previously described phase, the semantic bridging phase is responsible for establishing correspondence between entities from the source and target ontology. It intends to specify bridges between entities in a way that each instance represented according to the source ontology is translated into the most similar instance described according to the target ontology. This simple principle motivate our approach in semantic bridge specification following the evidence that RDFS ontologies normally rely and exploit the underlying OO part of

RDFS, namely the taxonomic structure in the form of a graph, and in particular cases, the form of a tree. The semantic bridging phase is divided in five distinct steps:

First, concept bridging chooses according to the similarities found in previous phase, pairs of entities to be bridged. The same source entity may be part of different bridges. Two distinct cases may arise: First, the source concept corresponds to either one of the target concepts. This implies that the source instance will give rise to one instance of just one of the target concepts. Second, the source concept correspond to many distinct target concepts, which implies that the source instance will give rise to one instance of many target concepts. The automatic process tries to find the best choice based on heuristics and lexical relations. For example, if the target concepts have the source concept as hypernym that tends to show that source instance should be translated to either one of the target concepts. The antonym relation (extracted from WordNet) may also be used for confirming of this case. On the other hand if no hypernym relation exist it tends to correspond to the second case.

Second, the property bridging step is responsible to specify the matching properties for each concept bridge. As for concepts, a property may be part of several matchings, which implies the same two cases previously mentioned for concepts. Therefore, the same strategy may be used in here. It is important to emphasize that properties in our approach are of two types, distinguishing between attributes and relations. If source and target properties are of different types the transformation specification information is required, where the domain expert is asked to supply this information.

Third, the inferencing step focus in endowing the mapping with bridges for concepts that do not have a specific counterpart target concept. In fact, a source concept c_s^1 may not always have a target concept counterpart c_t^1. However, if a match exists between the source concept c_s^0 (a super concept of c_s^1) and c_t^0, than an implicit similarity exists between c_s^1 and c_s^0.

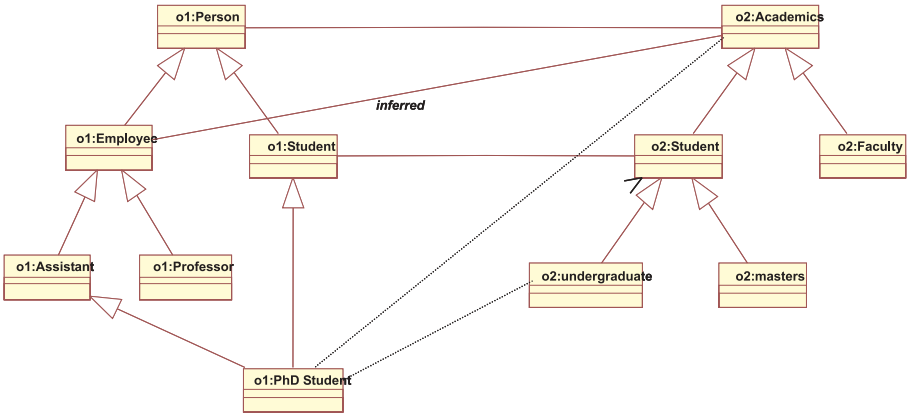

Fig. 2. Inferring best possible bridge

This scenario is depicted in Figure 2. Even if the concept EMPLOYEE has no direct counterpart in the target ontology, instances of this concept should be translated into

ACADEMICS instances. This can be automatically inferred because EMPLOYEE is sub concept of PERSON, which in turn is bridged with ACADEMICS. However this is not always a straight forward solution because ambiguity arises in some situations. To infer a bridge to PHD_STUDENT concept is one of such situations. This concept is sub concept of two concepts, which means that any instance of PHD_STUDENT is also an instance of both EMPLOYEE and STUDENT. However, such qualification do not exists in target ontology. In this situations we use available domain knowledge, namely the exploitation of previous mappings where such concepts were bridged. However, for the moment this decision is up to the domain expert. Inferred bridges are always sub bridges of some higher bridge and should not state the target entity. In this example, the process creates an inferred bridge that relies on between PERSON and ACADEMICS to execute the translation. This is called encapsulation in the OO paradigm.

Fourth, the refinement step intends to improve quality of bridges between a source concept and sub concepts of target concepts. In fact this is a complementary procedure of the similarity phase. Besides this step is optional, it becomes important if a good mapping quality is necessary.

Fifth, the transformation specification step intends to associate a transformation procedure to the translation, in a way that source instance may be translated into target instances. This task may be automatized in some extend, specially in well known situations, which can be acquired through experience. However this task is fundamentally a domain expert step. There are two main issues that are extremely dependent on the domain expert: *(i)* the alternative bridge conditions specification arising in concept bridging and property bridging, and *(ii)* the specification of mapping between different types of properties.

Execution. This module actually transforms instances from the source ontology into target ontology by evaluating the semantic bridges defined earlier. In general two distinct modes of operation are possible, namely offline (static, one-time transformation) and online (dynamic, continuous mapping between source and the target) execution. A description of our offline execution engine is provided in section 4.

Post-processing. The post-processing component takes the results of the execution module to check and improve the quality of the transformation results. The most challenging task of post-processing is establishing object identity - recognizing that two instances represent the same real-world object [7]. The post-processing process is not further elaborated in this paper.

2.2 Vertical Dimension of MAFRA

The vertical dimension of MAFRA contains modules that interact with horizontal modules during the overall mapping process. Following four modules have been identified. However, we will only focus on the GUI component in this paper.

Evolution. This aspect focuses on keeping semantic bridges obtained by the “Semantic Bridge” module, which must be kept in synchrony with the changes in the source and target ontologies. We refer the interested reader to [18] where we describe a user-driven ontology evolution strategy.

Cooperative Consensus Building. The cooperative consensus building aspect is responsible for establishing a consensus on semantic bridges between two communities participating in the mapping process. This is a requirement as one has to choose frequently from multiple, alternatively possible mappings .The amount of human involvement required to achieve consensus may be reduced by automating the mapping process as much as possible.

Domain Constraints and Background Knowledge. The quality of similarity computation and semantic bridging may be dramatically improved by introducing background knowledge and domain constraints, e.g. by using glossaries to help identify synonyms or by using lexical ontologies, such as WordNet or domain-specific thesauri, to identify similar concepts.

Graphical User Interface. Mapping is a difficult and time consuming process, which is not less difficult than building an ontology itself, i.e. deep understanding of both conceptualizations required on human side, thus extensive graphical support must be given and it is a separate issue how this can be achieved in an optimal way. The graphical user interfaces (GUI) is further elaborated in section 4.

3 Semantic Bridging

As mentioned in subsection 2.1, the role of the semantic bridging component is to semantically relate entities from the source and target ontologies. A role of a semantic bridge is to encapsulate all necessary information to transform instances of one source ontology entity to instances of one target ontology entity.

3.1 Dimensions of Semantic Bridges

The nature of semantic bridges may be understood by considering different dimensions, each describing one particular aspect of a semantic bridge. By analyzing ontologies used on the Semantic Web, we identified following five dimensions of semantic bridges:

- Entity dimension: Semantic bridges may relate the ontology entities *(i)* concepts (modeling classes of objects from the real world), *(ii)* relations (modeling relationships between objects in the real world), and, *(iii)* attributes (modeling simple properties of objects in the real world) and *(iv)* extensional patterns (modeling the content of the instances).
- Cardinality dimension: This dimension determines the number of ontology entities at both sides of the semantic bridge, ranging from $1:1$ to $m:n$. However, we have found that in most cases $m:n$ is not a common requirement, so $1:n$ and $m:1$ suffice. Even when $m:n$ are encountered, often they may be decomposed into m $1:n$ bridges.
- Structural dimension: This dimension reflects the way how elementary bridges may be combined into more complex bridges. We distinguish between the following different relations that may hold between bridges:

    - **Specialization** allows a bridge to reuse definitions from another bridge and provide additional information (e.g. a bridge relating Employee concepts from two ontologies may be a specialization of a more general bridge relating Person concepts),
    - **Abstraction** is a variation of the type of the super-classes. When this attribute is set, the specified bridge should not be executed independently, but only as super-class of another.
    - **Composition** relation between to bridges specifies that a bridge is composed of other bridges,
    - **Alternatives** relation between bridges specifies a set of mutually exclusive bridges.
- Constraint dimension: The constraint dimension permits to control the execution of a semantic bridge. It reflects relevant constraints applied during the execution phase to instances from the source ontology. Constraints act as conditions that must hold in order the transformation procedures is applied onto the instances of the source ontology, e.g. the bridge evaluate only if the value of the source instance matches a certain pattern.
- Transformation dimension: This dimension reflects how instances of the source ontology are transformed during the mapping process. Transformations assume different complexity and variety depending on the ontologies being bridged.

3.2 Semantic Bridging Ontology (SBO)

Within our approach four different types of relations between entities, a particular semantic bridge exists. A specification of all available semantic bridges, organized in a taxonomy, is a semantic bridging ontology (SBO). To actually relate the source and target ontology, the mapping process creates an instance of SBO containing semantic bridge instances, each encapsulating all necessary information to transform instances of one source entity to instances of the target entity. Figure 3 describes the most important entities of the semantic bridging ontology. We refer to the five, previously described semantic bridge dimensions:

- Three basic types of entities are considered: Concepts, Relations and Attributes,
- The class SEMANTIC BRIDGE is the most generic bridge, it defines the relations to source and target entities. It is specialized according to the entity type and according to cardinality. Though, there are many combinations of entity types and cardinality bridges that are not explicitly specified, it is important to mention that they can be easily specialized from more general bridges.
- The class SERVICE represents a class used to reference resources that are responsible to connect to, or describe transformations. This class is intended to be used to describe these transformations resources. Because services are normally external to the execution engine, it is required to describe some fundamental characteristics like name, interface (number and type of arguments) and location. Argument and its sub classes Arg and ArgArray permits to describes these characteristics in a simple and direct form.

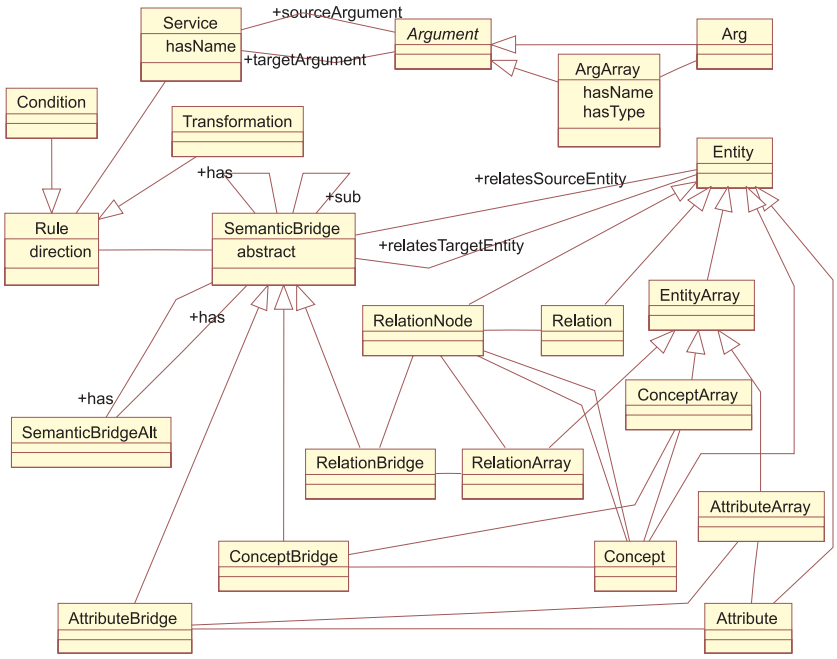

Fig. 3. Bridging Ontology view in UML

- RULE is the general class for constraints and transformation-relevant information, which provides a relation to the service class.
- The class TRANSFORMATION is mandatory in each semantic bridge except if the semantic bridge is set as abstract. It uses the inService relation to link to the transformation procedure, and any execution engine and function specific attributes in order to specify extra requirements;
- The class CONDITION represents the conditions that should be verified in order to execute the semantic bridge. Condition is operationally similar to transformation in the sense that it must specify all the extra requirements for the function that test the conditions. Because any semantic bridge may have a condition, it allows to control complex transformations according to both the schema and instances data, specially in combination with SemanticBridgeAlt and the Composition constructs.
- The COMPOSITION modelling primitive identified above is supported by the hasBridge relation in the SEMANTICBRIDGE class. It has no cardinality limit nor type constraint which allows any semantic bridge to aggregate many different bridges. Those semantic bridges are then called one by one, and processed in the context of the former.
- The ALTERNATIVE modelling primitive is supported by the SemanticBridgeAlt class. It groups several mutual exclusive semantic bridges. The execution parser checks each of the bridges condition rules and the first bridge which conditions hold is executed while the others are discarded.

In the following, we will describe how the semantic bridging ontology has been represented so it may be used within Semantic Web applications.

SBO represented in DAML+OIL. DAML+OIL[5] has been choosen to represent the semantic bridge ontology[6]. DAML+OIL builds on and extends RDF-Schema and provides a formal semantics for it. One of the goals in specifying the semantic bridge ontology was to maintain and exploit the existent constructs and minimize extra constructs, which would maximize as much as possible the acceptance and understanding by general Semantic Web tools.

3.3 Example

Let us consider Figure 4 where a small part of two different ontologies are represented. The ontology on the left side (o1) describes the structure of royal families and associated individuals. These concepts are combined with events, both individual events (birth date and death date) and family events (marriages and divorces). The ontology on the right side (o2), characterizes individuals using a very simple approach. It is mainly restricted in representing if the individual is either a Man or a Woman. The goal of this example is to specify a mapping between the source and target ontology, using the developed semantic bridge ontology). A mapping structure represented according to SBO tends to arrange bridges in a hierarchical way.

First, the mapping must define the two ontologies being mapped. Additionally, one may specify top-level semantic bridges which serve as entry points for the translation, even if there are not mandatory. In this case the translation engine starts executing the ”Individual-Individual” bridge.

```
<Mapping rdf:ID="mapping">
    <relatesSourceOntology rdf:resource="&o1;"/>
    <relatesTargetOntology rdf:resource="&o2;"/>
    <hasBridge rdf:resource="#Individual-Individual"/>
</Mapping>
```

Notice that the target ontology intends to create instances of either WOMAN or MAN, but not of INDIVIDUAL. In object oriented terminology the INDIVIDUAL concept is said to be abstract. It is therefore required to state that this concept bridge should not be used to create instances, but serve just as support to sub bridges, like it happens in object oriented paradigm. SBO uses the abstract property in these circumstances. If no abstract property is specified or if it is set to FALSE, then the concept bridge is considered as non-abstract.

It is now necessary to set the alternative between INDIVIDUAL and either WOMAN or MAN. This situation is specified by a SemanticBridgeAlt. In this case the alternatives are two ConceptBridge’s: ”Individual-Woman” and ”Individual-Man”. Bridges may be numerically ordered which can useful if the last bridge has no specified condition. Both rdf:_n like syntax and the one presented are allowed to specify the order.

[5] http://www.daml.org/2001/03/daml+oil-index.html

[6] The SBO ontology is available online at http://kaon.semanticweb.org/2002/04/SBO.daml

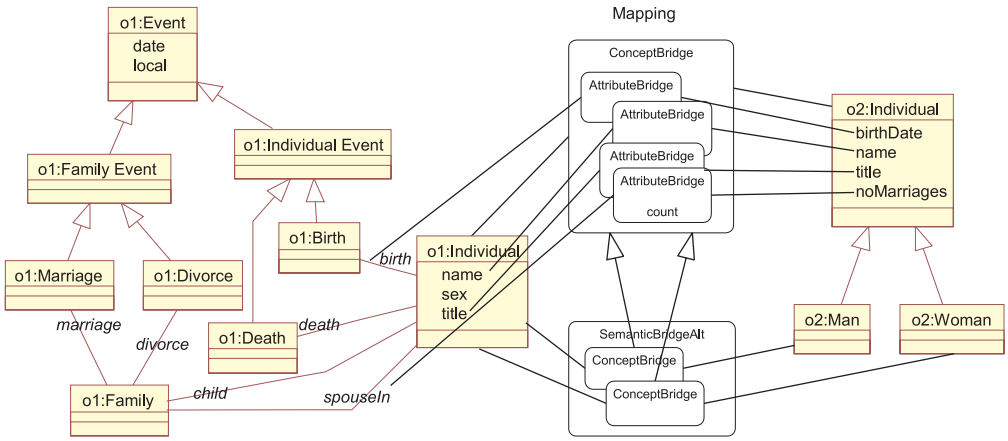

Fig. 4. UML representation of two small ontologies

```
<SemanticBridgeAlt rdf:ID="ManOrWoman">
    <hasBridge><Seq ordinal="1"><bridge rdf:resource="#Individual-Woman"/></Seq>
    </hasBridge>
    <hasBridge><Seq ordinal="2"><bridge rdf:resource="#Individual-Man"/></Seq>
    </hasBridge>
</SemanticBridgeAlt>
```

The alternative ConceptBridge's are presented next: "Individual-Woman" and "Individual-Man".

```
<ConceptBridge rdf:ID="Individual-Woman">
    <subBridgeOf rdf:resource="#Individual-Individual"/>
    <relatesSourceEntity rdf:resource="#Individual"/>
    <relatesTargetEntity rdf:resource="#Woman"/>
    <whenVerifiedCondition rdf:resource="#isFemale"/>
</ConceptBridge>

<ConceptBridge rdf:ID="Individual-Man">
    <subBridgeOf rdf:resource="#Individual-Individual"/>
    <relatesSourceEntity rdf:resource="#Individual"/>
    <relatesTargetEntity rdf:resource="#Man"/>
</ConceptBridge>
```

Both bridges rely on the "Individual-Individual" bridge to translate MAN and WOMAN inherited attributes from INDIVIDUAL. Hence, both are specified as sub-bridges of "Individual-Individual" concept bridge. Additionally, "Individual-Woman" concept bridge specifies the whenVerifiedCondition property to "isFemale". As remarked bellow, this condition is responsible to test if the individual is of feminine sex. If the condition is verified the bridge is executed. Otherwise, and because the condition is tested in the context of a SemanticBridgeAlt, the next concept bridge in the alternative is processed. The next concept bridge in the alternative is "Individual-Man" which has no associated condition, and therefore it is unconditionally executed.

Respecting the translation process, consider that an INDIVIDUAL instance is to be translated. The translation engine seeks for bridges relating INDIVIDUAL to any target ontology entity. Three are found, but one of them is abstract and is therefore rejected. The other two are both defined in the context of a SemanticBridgeAlt. The SemanticBridgeAlt choosing/exclusion process starts. One of the bridges (or eventually none if none of the associated conditions is verified) is selected. The concept bridge must then create a target instance which will serve as context for complementary bridges.

Complementary attribute bridges are in this example simple 1:1 attribute bridges, relating one attribute from o1 to an attribute in the target ontology, through the associated transformation.

```
<AttributeBridge rdf:ID="name-name">
    <relatesSourceEntity rdf:resource="#name"/>
    <relatesTargetEntity rdf:resource="#name"/>
    <accordingToTransformation rdf:resource="#copyName"/>
</AttributeBridge>

<Transformation rdf:ID="copyName">
    <mapSourceArgument>
        <MapArg><from rdf:resource="#name"/><to>sourceString</to></MapArg>
    </mapSourceArgument>
    <mapTargetArgument>
        <MapArg><from>targetString</from><to rdf:resource="#name"/></MapArg>
    </mapTargetArgument>
    <inService>CopyString</inService>
</Transformation>
```

Concerning the transformation, it intends to map between the bridge entities and the transformation service arguments. This mapping specification varies according to the service be requested, either in type, cardinality and used tags. For example, the ”copyName” transformation specifies the ”CopyString” service to be called. This service expects to receive a source argument called ”sourceString” and the output is named ”targetString”. The transformation maps ”sourceString” with the attribute ”o1:Individual.name” and ”targetString” to the ”o2:Individual.name”. ”title-title” attribute bridge is very similar to the previous and is not be presented.

In contrast, ”marriages” attribute bridges are slightly different from previous ones. Notice that the source entity is not an attribute but a relation to another concept. Normally an AttributeBridge would not be correctly applied. However, since this is a very common mapping pattern the translation engine allows to process the relation as an attribute. That could eventually be a problem if the translation service expects an attribute. However, the ”CountRelations” service expects a relation which is the case of ”spouseIn” and therefore no problem occurs.

```
<AttributeBridge rdf:ID="mariages">
    <relatesSourceEntity rdf:resource="#spouseIn"/>
    <relatesTargetEntity rdf:resource="#noMariages"/>
    <accordingToTransformation rdf:resource="#countSpouses"/>
</AttributeBridge>

<Transformation rdf:ID="countSpouses"> <putServiceArgument>
        <MapArg><from>relation</from><to rdf:resource="#spouseIn"/></MapArg>
    </putServiceArgument>
    <mapTargetArgument>
        <MapArg><from>count</from><to rdf:resource="#noMariages"/></MapArg>
    </mapTargetArgument>
    <inService>CountRelations</inService>
```

```
</Transformation>

<AttributeBridge rdf:ID="birth-birthDate">
    <relatesSourceEntity rdf:resource="#birth"/>
    <relatesTargetEntity rdf:resource="#birthDate"/>
    <accordingToTransformation rdf:resource="#Birth"/>
</AttributeBridge>

<Transformation rdf:ID="Birth">
    <putServiceArgument>
        <MapArg><from>1</from><to rdf:resource="#birth"/></MapArg>
    </putServiceArgument>
    <putServiceArgument>
        <MapArg><from>2</from><to rdf:resource="#date"/></MapArg>
    </putServiceArgument>
    <mapTargetArgument>
        <MapArg><from>targetString</from><to rdf:resource="#birthDate"/></MapArg>
    </mapTargetArgument>
    <inService>RoyalDate</inService>
</Transformation>
```

Finally, the ”isFemale” condition is considered. This condition is responsible to verify if an instance of an individual is of feminine sex. In this case the pattern refers to the fact that the value of sex attribute has value ”F”. Normally, the services applied in a condition return a boolean value. However, this constraint would depend on the translation engine once it is possible to create a table of correspondences between boolean types and other types. For example, it would be reasonable to consider a true result if the service returns a set of entities or false if it return a empty set.

```
<Condition rdf:ID="isFemale">
    <putServiceArgument>
        <MapArg><from>1</from><to rdf:resource="#sex"/></MapArg>
    </putServiceArgument>
    <putServiceArgument>
        <MapArg><from>pattern</from><to>F</to></MapArg>
    </putServiceArgument>
    <inService>CascadeAndMatch</inService>
</Condition>
```

4 Implementation

MAFRA is currently under development within the KAON Ontology and Semantic Web Framework[7]. For the moment we achieved the implementation of four modules of MAFRA: The automatic similarity discovery module, the semantic bridging representation, the graphical user interface and the execution engine.

A screen-shot of the user interface for mapping specification is presented in Figure 5. In this example two ontologies have been opened side by side, and in between an instance of the semantic bridging ontology is created using a simplified user interface.

The developed mapping tool represents the domain expert interface with the similarity and semantic bridging modules, and the possibility to interact within the mapping process. The user participation is fundamental and must be promoted. We adopted a tree view similar to the most common ontology editors. The mapping tool defines two tree views for the ontologies being mapped (in the left and in the right) and a central tree

[7] http://kaon.semanticeweb.org

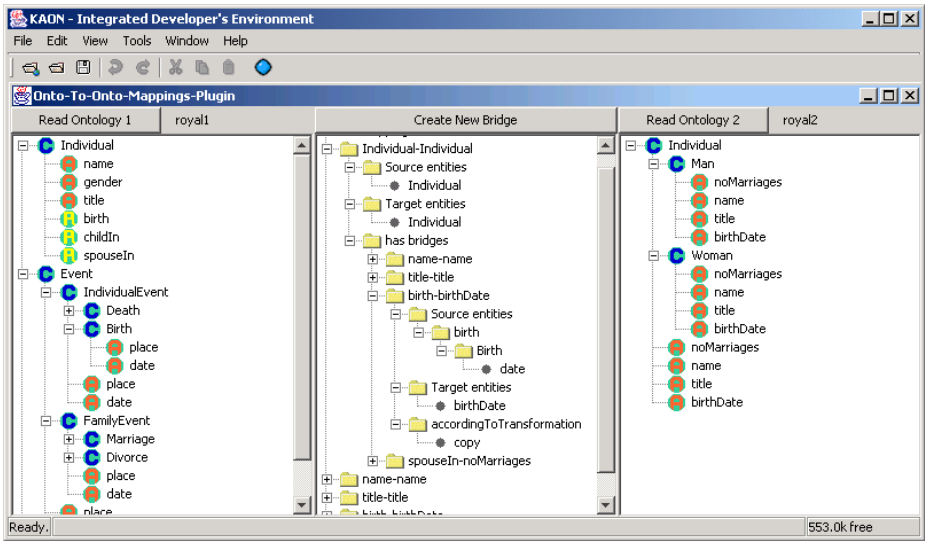

Fig. 5. Creating Mappings Using KAON Tools

view representing the mapping. Bridges are manipulated through drag and drop actions. Entities from ontologies are dragged and dropped in a bridge and are stored either in the source or target entities folder. The same happens when specifying the mappings between bridges parameters and services arguments. For the moment it is not possible to edit transformation and condition procedures. They are read/parsed into the interface through a menu command.

The execution engine has been implemented in Java, exploiting the features of KAON, and it represents the first step of out efforts in developing a general translation engine for SBO instances. The execution engine uses a mapping instance, which is an instantiation of the SBO, and a set of source ontology instances. The transformation engine parses the mapping into the KAON ontology model and executes it. The process runs for each concept instance that have an associated concept bridge. The internal structure of the execution engine resemble very much the semantic bridge ontology model. A class is defined for some of the major components of the SBO which implement the functionally described in section 3:

- The mapping class is responsible to read source instances and call the associated bridge, if any. However, as described before, a source instance may have multiple associated bridges which implies the mapping checks it and call the alternative bridge instead.
- The AlternativeBridge class is responsible to try the execution of each of its composing bridge, one after another until one of them is executed.
- The ConceptBridge class encompasses all the information related to the instance, and it encodes the necessary functionality to to carry out the task. Mostly, the ConceptBridge class has four ordered tasks: *(i)* check if the whenVerifiedCondition holds; if it holds *(ii)* create an empty target instance, *(iii)* call the subBridge's

bridges (concept and attribute bridge) if some exists, and *(iv)* call the hasBridge's bridges.
- Attribute and Relation Bridge, even if conceptually different their functioning is very similar. The execution context of these bridges is an concept instance. This instance was previously created and received from the concept bridge. The transformations are executed and the resulting values are associated with the current instance.
- The Service class is responsible to map the bridge parameters (entities) with the transformation procedure arguments and to call the procedures.

5 Related Work

Much research has been done in the area of information integration. Existing information integration systems and approaches (e.g., TSIMMIS [6], Information Manifold [8], Infomaster[8], MOMIS[9], Xyleme [10]) are "centralized" systems of mediation between users and distributed data sources, which exploit mappings between a single mediated schema and schemas of data sources. Those mappings are typically modeled as views (over the mediated schema in the local-as-view approach, or over the sources schemas in the global-as-view approach) which are expressed using languages having a formal semantics. For scaling up to the Web, the "centralized" approach of mediation is probably not flexible enough, and distributed systems of mediation are more appropriate.

Furthermore, mapping approaches can mainly be distinguished along the following three categories: discovery, [14, 3, 5, 10, 1], mapping representation [9, 1, 11, 13] and execution [4, 11]. However, none of the proposed solutions has really encompassed the overall mapping process specially considering the evolution and consensus building of semantic bridges. Having this in mind, we have introduced the Ontology MApping FRAmework (MAFRA) as a basis for managing and executing mapping between distributed ontologies in the Semantic Web. Within MAFRA we provide an approach and conceptual framework that provides a generic view and figure onto the overall mapping process. In this paper we have set a specific focus on the semantic bridging phase corresponding to the mapping representation category. The approaches which resemble our approach more closely are [13] and [12]. Basically, our work has been motivated by the work done in [13], where an ontology has been specified for the translation between the domain-knowledge-base components and problem-solving-method components. The approach that comes nearest to ours has been described in [12]. They describe an approach for integrating vocabularies including means for mapping discovery and representing mappings with a focus on B2B applications (product catalogues) has been described. In contrast to our work, the RDFT ontology describes a set of core bridges to *(i)* lift XML tags to the RDF model and *(ii)* to define bridges between RDF(S) classes and properties and to *(iii)* translate transformation results back to XML. In the paper [12] it remains unclear, how execution specific information in the form of our constraint and transformation dimension is attached to the bridges.

[8] http://infomaster.stanford.edu/infomaster-info.html
[9] http://sparc20.ing.unimo.it/Momis/
[10] http://www.xyleme.com

6 Conclusion and Future Work

Ontologies may used for achieving a common consensus within a user community about conceptualizing, structuring and sharing domain knowledge. Based on the application scenario provided by Ontologging we have motivated that it is unrealistic to assume that one single ontology for different communities of users is realistic in real-world applications. We argue that decentralization has been one of the key elements for the scalability of the World Wide Web and its underlying applications. In order to balance the autonomy of each community with the need for interoperability, mapping mechanisms between ontologies have been proposed. In this paper we presented the Ontology Mapping Framework (MAFRA) supporting the interactive, incremental and dynamic ontology mapping process in the context of the Semantic Web. In this paper a specific focus has been set on the semantic bridging phase where we have provided a detailed description of a semantic bridge meta-ontology, that is instantiated when mapping between two domain ontologies.

In the future much work remains to be done. First, depending on the domain ontologies, data sources, application scenarios, user participation, capabilities and other factors further semantic bridges may be necessary. For example, procedural mechanisms may complement the taxonomy of semantic bridges. Thus, we consider the semantic bridging ontology as evolving. Second, considering the mapping process as a consensus building process of two communities, we will on the basis of our technological infrastructure KAON, perform an experiment how multi-user mapping may be efficiently supported. Third, we will develop an integrated LIFT tool that allows to lift several existing data representations including relational databases, XML-Schema, DTDs onto the same data model. Executing a dynamic mapping process keeping the autonomy of the different input data will be a challenging task.

Acknowledgements. Research for this paper was financed by European Commission, IST, project "Ontologging" (IST-2000-28293) and by Marie Curie Fellowship on Semantic Web Technologies. Special thanks to Gabor Nagypal for fruitful discussions on defining the semantic bridging ontology and Oliver Fodor for stimulating discussions on the lift component and cooperative mapping. Thanks to the students Frank Westerhausen and Zoltan Varady who did the implementation work for the graphical user interface and the static transformation engine.

References

1. S. Bergamaschi, S. Castano, D. Beneventano, and M. Vincini. Semantic integration of heterogeneous information sources. In *Special Issue on Intelligent Information Integration, Data & Knowledge Engineering*, volume 36, pages 215–249. Elsevier Science B.V., 2001.
2. T. Berners-Lee. *Weaving the Web*. Harper, San Francisco, 1999.
3. W. Cohen. The whirl approach to data integration. *IEEE Intelligent Systems*, pages 1320–1324, 1998.
4. T. Critchlow, M. Ganesh, and R. Musick. Automatic generation of warehouse mediators using an ontology engine. In *Proceedings of the 5 th International Workshop on Knowledge Representation meets Databases (KRDB'98)*, 1998.

5. A. Doan, J. Madhavan, P. Domingos, and A. Halevy. Learning to map between ontologies on the semantic web. In *Proceedings of the World-Wide Web Conference (WWW-2002)*, 2002.
6. J. Hammer, H. Garcia-Molina, K. Ireland, Y. Papakonstantinou, J. Ullman, and J. Widom. Information Translation, Mediation, and Mosaic-Based Browsing in the TSIMMIS System. In *Exhibits Program of the Proceedings of the ACM SIGMOD International Conference on Management of Data, page 483, San Jose, California, June 1995.*, 1995.
7. S. Khoshafian and G. Copeland. Object identity. In *Proceedings of the 1st ACM OOPSLA conference, Portland, Oregon, September 1986.*, 1985.
8. Alon Y. Levy, Anand Rajaraman, and Joann J. Ordille. Querying Heterogeneous Information Sources Using Source Descriptions. In *Proceedings of VLDB-96, 1996*, 1996.
9. J. Madhavan, P. A. Bernstein, and E. Rahm. Generic schema matching with cupid. In *Proceedings of the 27th International Conferences on Very Large Databases*, pages 49–58, 2001.
10. A. Maedche and S. Staab. Computing Similarities between Ontologies. In *Proceedings of the 13th European Conference on Knowledge Engineering and Knowledge Management EKAW-2002, Madrid, Spain*, 2002.
11. P. Mitra, G. Wiederhold, and M. Kersten. A graph-oriented model for articulation of ontology interdependencies. In *Proceedings of Conference on Extending Database Technology (EDBT 2000)*. Konstanz, Germany, 2000.
12. B. Omelayenko. Integrating Vocabularies: Discovering and Representing Vocabulary Maps. In *Proceedings of the First International Semantic Web Conference (ISWC-2002), Sardinia, Italy, June 9-12, 2002.*, 2002.
13. J. Y. Park, J. H. Gennari, and M. A. Musen. Mappings for reuse in knowledge-based systems. In *Technical Report, SMI-97-0697, Stanford University*, 1997.
14. E. Rahm and P. Bernstein. A survey of approaches to automatic schema matching. *VLDB Journal*, 10(4):334–350, 2001.
15. P. Resnik. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *Journal of Artificial Intelligence*, 11(11):95–130, 1999.
16. M.C. Rousset. Standardization of a web ontology language. *IEEE Intelligent Systems, March/April 2002*, 2002.
17. N. Silva. Discovering Mappings between Distributed Ontologies. In *Internal Report, University of Karlsruhe, July 2002.*, 2002.
18. L. Stojanovic, A. Maedche, B. Motik, and N. Stojanovic. User-Driven Ontology Evolution. In *Proceedings of the 13th European Conference on Knowledge Engineering and Knowledge Management EKAW-2002, Madrid, Spain*, 2002.
19. P.R.S. Visser, D.M. Jones, T.J.M. Bench-Capon, and M.J.R. Shave. An analysis of ontology mismatches: Heterogeneity versus interoperability. In *AAAI 1997 Spring Symposium on Ontological Engineering, Stanford CA., USA*, pages 164–72, 1997.