An Ontology-Mapping Service for Agent-Based Automated Negotiation

Henrique Lopes Cardoso, Daniel Dinis Teixeira and Eugénio Oliveira

LIACC, DEI / Faculdade de Engenharia, Universidade do Porto
R. Dr. Roberto Frias, 4200-465 Porto, Portugal
{hlc, daniel.teixeira, eco}@fe.up.pt

Abstract. The automation of B2B processes requires a high level of interoperability between potentially disparate systems. We model such systems using software agents (as enterprise’s representative agents), which interact using ACL protocols. This paper presents, in the context of an Electronic Institution, an ontology-mapping service that enables the automation of negotiation protocols where the involved agents may use different ontologies to represent their domain knowledge. The ontology-mapping service employs two approaches used for lexical and semantic similarity, namely N-Grams and WordNet, and poses few requirements on the ontologies’ representation format. Examples are provided that illustrate the integration of ontology-mapping with automated negotiation.

1 Introduction

Technological support to the creation of B2B relationships is arising in many forms. The most ambitious ones intend to automate (part of) the process of creation and execution of contracts, mainly through multi-agent system (MAS) approaches.

The agent technology roadmap [9] identifies as key problem areas the development of infrastructures for open agent communities, as well as the need for trust and reputation mechanisms. Electronic institutions, together with ontologies and related services, address the needed infrastructures. Norms, electronic contracts and their enforcement are pointed out as means to achieve trust in open environments. We have been developing an Electronic Institution platform motivated by the need to develop services that assist the coordination efforts among agents which, representing different real-world entities, interact with the aim of establishing business relationships.

An intrinsic problem that must be dealt with when approaching open systems is that each of a set of heterogeneous entities may potentially use a different domain ontology. There may be syntactic or semantic discrepancies in these ontologies: the same information may be stored in different representation formats, disparate terminologies for the same concepts may exist, or even the same terminology may be used for distinct concepts. This heterogeneity is a critical impediment to efficient business information exchange and to the automation of B2B processes.
The most simplistic way of solving this problem (often called the interoperability or heterogeneity problem [10, 16]) would be to define either a common ontology or a shared one which could be understood by all agents participating in business interactions. However, open environments (where a central design is neither possible nor desirable) populated with heterogeneous agents make the common ontology case unfeasible. Each agent will typically use a different ontology, and enterprises will not consider converting all the content of their ontologies if the target ontology is less expressive or not considered as a de facto standard.

The Foundation for Physical Intelligent Agents (FIPA) [5] has analyzed the interoperability problem in heterogeneous Multi-Agent Systems (MAS) and has proposed an Ontology Agent (OA) for MAS platforms [4]. Among other responsibilities, the OA may provide the translation service of expressions between different ontologies or different content languages by itself, possibly as a wrapper to an ontology server. In this paper we present an implementation of such a service, embedded in an Electronic Institution. The ontology-mapping service is aligned with a negotiation mediation service, allowing negotiation to take place between entities using different domain ontologies.

The paper is organized as follows. Section 2 contextualizes the usage of an ontology-mapping service in agent-based automated contracting. Section 3 details the service itself. In section 4 examples that exploit the service are provided. Section 5 concludes.

## 2 Interoperability in Agent-Based Automated Contracting

There is a strong research effort towards the automation of B2B contracting processes. In particular, multi-agent systems technology is being used to establish business contracts by automatically negotiating agreements.

An Electronic Institution (EI) is a software platform that aims at (i) supporting agent interaction as a coordination framework, making the establishment of business agreements more efficient; and at (ii) providing a level of trust by offering an enforceable normative environment [8]. The ontology service described in this paper is essential to serve the first of these aims. Particularly when addressing an open environment, where a central design is not possible, agents representing different enterprises (henceforth enterprise agents) may use different domain ontologies, which have to be matched in order to make the (automated) establishment of agreements possible.

The EI will offer a set of services related to contract establishment and execution. A major service concerns negotiation mediation, through which an enterprise agent may automatically find and negotiate with potential partners. Negotiation is typically based on appropriate negotiation protocols and contract templates. The beginning of the negotiation mediation process is where ontology services come into play.

In open environments, different domain-dependent vocabulary may be used by different business entities. Ontology services are important to allow for negotiation to take place. Figure 1 illustrates these concepts.
The negotiation mediation service acts as a mediator between an enterprise agent and a set of potential partners (other enterprise agents). Each of these has a set of competences based on certain classes of components that it is able to supply. When asked for, each enterprise agent can negotiate the supply of a component of a certain class, if that class is in his competence list. Figure 2 illustrates the start of the negotiation process when there is no ontology service for solving the heterogeneity problem: enterprise agents on the right side may be prevented from participating in the negotiation because of an ontology mismatch.

Fig. 1. The interplay between negotiation mediation and ontology services

Fig. 2. Negotiation-mediation without an ontology-mapping service

The ontology-mapping service can be used when some enterprise agent does not understand the content of a CFP message (i.e. the component class under negotiation). The agent may recur to the service in order to find out if he supplies components of a
class matching the one asked for. Figure 3 illustrates this process. In this case an enterprise agent on the right side is able to participate in the negotiation thanks to the ontology-mapping service, which gave him a mapping between the asked component class and one described in his ontology.

This kind of service is mostly important if we consider that the negotiation process is to be automated through the use of enterprise (software) agents. In the next section we detail the workings of the ontology-mapping service.

3 Ontology-Mapping Service

Our background scenario is based on a set of enterprise agents requesting or supplying certain classes of components, for which they use a negotiation mediation service. Despite their potential interest in the same components, it is not guaranteed that they use the same names to define them. Suppose that a customer wants an ‘alarm’ and a supplier has exactly the component that this customer is looking for; however, in the supplier’s ontology the component is known as a ‘siren’. An automated negotiation process will fail if this ontology mismatch is not dealt with. Our approach is based on a service whose aim is to make a mapping between concepts defined in two different ontologies.

This section describes how the mapping process takes place. This process is based on the principle that if two different ontologies represent the same domain, then there
is a high probability that the described concepts have a similar syntax and share similar attributes [10]. We will start by describing the minimum set of assumptions that enable the usage of the ontology-mapping service.

3.1 Assumptions on Ontology Representation

Even in open environments, a minimum set of conventions is needed to enable the interaction between heterogeneous agents, be it an ACL, negotiation protocols, and so on. In the B2B domain, it is generally assumed that parties have a common understanding on domain-independent business vocabulary. Concepts like proposal, deal or price must be part of a common base ontology. If we want enterprise agents to automatically negotiate contracts, they should also have a common understanding of what a delivery or a payment means.

In order to render an ontology-mapping service, a minimum set of requirements is also needed regarding the representation of components in different ontologies. Each ontology must be describable in terms of classes and attributes (see Figure 4). Each component is an instance of a class that defines its type. Each class has a name and a set of typed attributes. The mapping-service will be based on matching class names and class attributes.

![Diagram of classes, attributes and components](image)

**Fig. 4. Classes, attributes and components**

The following sub-section describes how the mapping process takes place.

3.2 Ontology-Mapping

Ontology mapping is the process of finding correspondences between the concepts of two ontologies. If two concepts correspond, then they mean the same thing or closely related things. The mapping process is based on two approaches. The first approach is *N-grams* [2]: an algorithm that takes as input two strings and computes the number of common sub-strings between them. The other approach consists of using *WordNet* [17], which is a free lexical database containing semantic and lexical relations between words. Succinctly, the *N-Grams* algorithm computes a lexical similarity between two words, while *WordNet* computes a semantic one. These two approaches are applied to the names of the classes and also to their attributes.
**N-Grams.** The N-Grams [2] algorithm takes as input two strings and computes the number of common n-grams between them. An n-gram is a sequence of n characters; for each string, the algorithm computes the set of all possible n-grams that are in each string. A pre-processing step consists of normalizing both strings: all non-alphanumeric characters are replaced with ‘_’. The second step is to get the n-grams from each string (sub-strings of length n). Finally, the algorithm counts the number of n-grams of the first string that match an n-gram of the second string. The number of matches is used to calculate the outcome of the algorithm – a value of similarity is obtained from the formula:

\[
\text{Value} = \frac{\text{number of matches}}{\text{number of n-grams in first string}}
\]

The value obtained is within the range [0, 0; 1, 0]. The algorithm is parameterized with the value to n (the size of each n-gram). In our approach we chose a value of n=3 to produce 3-grams.

The N-Grams similarity approach has been used as an alternative to word-based systems. It has the merit of being robust in misspelling cases, which can be expected to be abundant in a scenario with multiple ontologies for the same domain.

**WordNet.** WordNet [17] is a lexical database designed for automatic processing that provides an effective combination of traditional lexicographic information and modern computing. WordNet contains thousands of words, including nouns, verbs, adjectives and adverbs. These words are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Other comparable systems exist [7], but with essentially different purposes, e.g. CYC (a general knowledge base and commonsense reasoning engine) or EDR (a dictionary with a bilingual English-Japanese emphasis). Comparing to WordNet, CYC is a more general-purpose system, while EDR has a different scope. WordNet fits our purposes for being an essentially linguistic knowledge base for English.

Furthermore, we make use of WordNet::Similarity [11], a WordNet-based Perl module which measures the similarity or relatedness between a pair of concepts, according to a several different measures. Measures of similarity quantify how much two concepts are similar, based on information contained in a hierarchical model. For instance, an ‘automobile’ can be considered more like a ‘boat’ than a ‘tree’, if ‘automobile’ and ‘boat’ share ‘vehicle’ as a common ancestor. Measures of relatedness compare concepts using relations like “has part”, “is made of” or “is an attribute” instead of a hierarchical model. For instance a ‘wheel’ would give a good relatedness with a ‘car’, since a ‘wheel’ is an attribute of a ‘car’ [11].

In previous work [10] we concluded that the most appropriate measure for our scenario is the Leacock-Chodorow (LCH) measure. LCH is a similarity measure based on path lengths between concepts. This measure finds the shortest path between two concepts and scales that value by the maximum path length in which they occur.

In our domain, class and attribute names are not necessarily simple words. For instance, an attribute can be a composition of words, such as ‘vision_angle’. A pre-processing step is needed, since the attribute’s name is not in WordNet’s database. However, the words ‘vision’ and ‘angle’ are. A preprocessing step consists of dividing names and considering all the words that compose it.
Mapping Process. Since we don’t know at the beginning if two words have a lexical or semantic similarity, the ideal would be to apply both measures for each pair of words. However, this is not feasible because of performance issues. WordNet requires a client/server architecture with socket communications, which introduces a large latency. Therefore, we firstly apply N-Grams and only if the result is not satisfactory we make use of WordNet.

Ontology mapping starts with a list of component classes that can be matched with the requested (i.e. target) component class. Each class in the list will be tested. We choose the best matching class provided that it has a satisfactory value:

1. Let $bc$ be the best matching class and $bs$ its matching score
2. For each class $c$ in the list
   a. Compute $c$’s matching score with target class
   b. Update $bc$ and $bs$
3. If $bs$ is satisfactory then return $bc$, otherwise return null

The matching score (2.a above) between a class and the target class is the average of two values: the similarity score of their names and their attributes. The attribute matching process is done only for attributes of the same type, and is successful only if there is a mapping for every attribute of the target class:

1. Compute the class name similarity score $ns$
2. Compute the attribute list similarity score $as$
   a. Let $as$ be the attribute list similarity score
   b. For each attribute $at$ in the target class attribute list
      i. Find the “unmatched” class attribute $a$ with best similarity score
      ii. Update $as$
      iii. Mark $a$ as “matched”
3. Return the average of $ns$ and $as$

Similarity is calculated by first applying N-Grams and eventually using WordNet:

1. Compute the N-Grams similarity score $ngs$
2. If $ngs$ is satisfactory then return $ngs$
3. Compute the WordNet similarity score $wns$
4. Return max of $ngs$ and $wns$

The following section describes an example that shows a scenario with suppliers and customers having different ontologies and where this mapping process is applied.

4 Example

In order to exemplify the usage of the ontology-mapping service and the mapping of several classes, a scenario is described in the following sub-sections. The scenario includes suppliers and customers interested in components from the domotics domain. It was tested in our Electronic Institution platform developed with the JADE framework [6].

Ontologies were created using the Protégé ontology editor [13] and saved in OWL files. This format allows defining classes of components in an object oriented model, where sub-classes inherit attributes from super-classes. Each enterprise agent instantiates components in an OWL file extended from the ontology definition.
4.1 Scenario

The scenario contains six agents (see Figure 5). Five of them are suppliers (Supply1 to Supply5) and one is a customer (Request1). The customer uses the same ontology (B) as Supply1, Supply2 and Supply3. On the other hand, Supply4 and Supply5 have defined their components based on a different ontology (A). The arrows in Figure 5 show which classes of components are supplied by each of the suppliers. The customer Request1 is interested in composing a package with four different components: a ‘Command’, a ‘Switch’, an ‘Alarm’ and a ‘Camera’. In ontology A these kinds of components are known, respectively, as ‘Control’, ‘Cutout’, ‘Siren’ and ‘Photographic_Equipment’. Suppliers Supply4 and Supply5 need to use the ontology-mapping service if they are to enter the negotiation for each of the requested components. Supply4 is the only agent who has a ‘Camera’ (‘Photographic_Equipment’ in his ontology); therefore, it is absolutely necessary that the mapping is correctly done, otherwise Request1 will not be able to negotiate this component, which will prevent him from composing the intended package.

![Fig. 5. Scenario: agents, ontologies and classes of components](image)

In addition to the class names, the ontologies also differ in the attribute names for each class. Table 1 summarizes the attribute names for both ontologies. We can notice that the attribute named ‘price’ is the only one which is actually the same in both ontologies. For all other attribute names, some are lexically similar, while others have only a semantic resemblance. For instance, ‘has_wireless’ in ontology B is lexically similar to ‘wireless’ in ontology A. On the other hand, the attribute ‘sight_grade’ in ontology A has no lexical similarity with the attribute ‘vision_angle’ in ontology B, and yet they mean the same thing. Hence it is easy to anticipate that the mapping of the attributes ‘wireless’ and ‘has_wireless’ will be solved by N-Grams, while WordNet will solve the pairing of ‘sight_grade’ with ‘vision_angle’.

According to this scenario, we expect that the ontology-mapping service is able to map classes from ontology A with classes from ontology B. For instance, when Supply4 receives a CFP from the negotiation mediator (see Figure 3) asking for a ‘Camera’, he looks at his ontology and does not find that class. Consequently, he will ask the ontology-mapping service, which will give him the respective mapping, telling him that a ‘Camera’ is the same as a ‘Photographic_Equipment’. Additionally, the service will also give him a mapping between the attributes of ‘Camera’ and those of ‘Photographic_Equipment’.
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Class</th>
<th>Ontology A</th>
<th>Ontology B</th>
<th>Attribute</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>price</td>
<td>all</td>
<td></td>
<td></td>
<td>price</td>
<td>all</td>
</tr>
<tr>
<td>range</td>
<td>Control</td>
<td></td>
<td></td>
<td>reach</td>
<td>Command</td>
</tr>
<tr>
<td>cipher</td>
<td>Control</td>
<td></td>
<td></td>
<td>code</td>
<td>Command</td>
</tr>
<tr>
<td>numButton</td>
<td>Control</td>
<td></td>
<td></td>
<td>number_button</td>
<td>Switch</td>
</tr>
<tr>
<td>decibel</td>
<td>Siren</td>
<td></td>
<td></td>
<td>db</td>
<td>Alarm</td>
</tr>
<tr>
<td>wireless</td>
<td>Photographic_Equipment</td>
<td>lens_size</td>
<td>Camera</td>
<td>has_wireless</td>
<td>Camera</td>
</tr>
<tr>
<td>sight_grade</td>
<td>Photographic_Equipment</td>
<td>vision_angle</td>
<td>Camera</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lens_dimension</td>
<td>Photographic_Equipment</td>
<td></td>
<td></td>
<td>lens_size</td>
<td>Camera</td>
</tr>
</tbody>
</table>

### 4.2 Results

Table 2 shows the values obtained from mapping ‘Camera’ with ‘Photographic_Equipment’. As we can see, the mapping between class names was obtained using WordNet, after a foreseeable failure of N-Grams – there is no lexical similarity between the two words. This value is 0.81 and represents 50% of the final score for this class. Attribute ‘price’ has a perfectly matching attribute according to N-Grams, hence the confidence of 1.00. Attribute ‘has_wireless’ had a satisfactory matching with attribute ‘wireless’ using N-Grams (0.64). As for attributes ‘lens_size’ and ‘vision_angle’, they both did not get a satisfactory result using N-Grams. A better result was obtained using WordNet: ‘lens_size’ matched ‘lens_dimension’ with 0.85; ‘vision_angle’ matched ‘sight_grade’ with 0.73. The global score for attribute matching is the average of each individual attribute matching score: (1.00 + 0.64 + 0.85 + 0.73) / 4 = 0.81. The final score is then the average of both (class and attributes) scores: (0.81 + 0.81) / 2 = 0.81.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Class</th>
<th>Ontology A</th>
<th>Ontology B</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>price</td>
<td>all</td>
<td></td>
<td></td>
<td>0.81 (WordNet)</td>
</tr>
<tr>
<td>has_wireless</td>
<td>all</td>
<td></td>
<td></td>
<td>1.00 (N-Grams)</td>
</tr>
<tr>
<td>lens_size</td>
<td>Camera</td>
<td></td>
<td></td>
<td>0.64 (N-Grams)</td>
</tr>
<tr>
<td>vision_angle</td>
<td>all</td>
<td></td>
<td></td>
<td>0.85 (WordNet)</td>
</tr>
<tr>
<td>sight_grade</td>
<td>Camera</td>
<td></td>
<td></td>
<td>0.73 (WordNet)</td>
</tr>
</tbody>
</table>

The ontology-mapping service gave results of high confidence for all the classes considered. Results are summarized in Table 3.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Class</th>
<th>Ontology A</th>
<th>Ontology B</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control</td>
<td>Command</td>
<td></td>
<td></td>
<td>0.97</td>
</tr>
<tr>
<td>Cutout</td>
<td>Switch</td>
<td></td>
<td></td>
<td>0.82</td>
</tr>
<tr>
<td>Siren</td>
<td>Alarm</td>
<td></td>
<td></td>
<td>0.90</td>
</tr>
<tr>
<td>Photographic_Equipment</td>
<td>Camera</td>
<td></td>
<td></td>
<td>0.81</td>
</tr>
</tbody>
</table>

The user interface at Figure 6 shows, for Supply4 (who had components belonging to classes ‘Cutout’, ‘Siren’ and ‘Photographic_Equipment’), that all these classes...
were correctly mapped by the ontology-mapping service. Supply5 also had a list of mapped classes for ‘Control’ and ‘Cutout’.

Figure 6 shows information regarding the negotiation of the four components that were part of the package intended by Request1. In the second column we can see the number of agents who negotiated those components. We can conclude, looking again at Figure 5, that all agents who had a component to supply were involved in the negotiation, regardless of the ontology they had adopted. Three agents negotiated the ‘Command’ component: Supply1, Supply2 and Supply5. All agents negotiated the ‘Switch’ component. Two agents negotiated the ‘Alarm’ component: Supply3 and Supply4. Finally, the only supplier who had a ‘Camera’ – Supply4 (a ‘Photographic_Equipment’ in his ontology) – was the only one who entered the negotiation of that component.

Figure 7 shows information regarding the negotiation of the four components that were part of the package intended by Request1. In the second column we can see the number of agents who negotiated those components. We can conclude, looking again at Figure 5, that all agents who had a component to supply were involved in the negotiation, regardless of the ontology they had adopted. Three agents negotiated the ‘Command’ component: Supply1, Supply2 and Supply5. All agents negotiated the ‘Switch’ component. Two agents negotiated the ‘Alarm’ component: Supply3 and Supply4. Finally, the only supplier who had a ‘Camera’ – Supply4 (a ‘Photographic_Equipment’ in his ontology) – was the only one who entered the negotiation of that component.

The use of the ontology-mapping service made it possible for agent Request1 to successfully negotiate all the components of the package it intended to assemble.

5 Conclusions

The heterogeneity problem in ontology specification is a strong impediment to the development of interoperable automated tools. In our case, we address this interoperability issue from a multi-agent system perspective: agents need to solve
their ontological differences in order to be able to automatically negotiate on behalf of their owners.

The research literature devoted to Electronic Institutions does not emphasize the importance of having ontology mapping services. In this respect, our approach is original, as far as we know. Some authors [3] point out the need for having a common ontology available for all parties inside the institution, describing both general and domain-dependent concepts. These approaches therefore avoid the heterogeneity problem.

Other authors have tackled the problem of ontology disparity in the past. However, most of them do not integrate their approaches with agent interaction protocols. Furthermore, some approaches force modifications in the original ontologies [1], require the inspection of instances described in those ontologies [15], impose the creation and usage of a new merged ontology [12], or assume more requirements on the original ontologies’ representations [14].

Our approach does not require an enterprise agent to reveal possibly sensitive information regarding his competencies. All is required is that he is able to describe his ontology in terms of classes and attributes, nothing else. Moreover, the original ontologies can be maintained, which is an important advantage in a B2B context, where an ontology switch can be an expensive task.

Another important feature of our approach is the contextualization and integration of the ontology-mapping service with a negotiation protocol for agent-based automated negotiation. The service enables the use of such automation in open settings, which would otherwise be unfeasible.

We are aware that our experiments are based on simplified artificial scenarios. In fact, most experiments reported in the literature so far, are toy problems. Real experiments with ontology mapping and integration are missing, probably caused by the lack of available real-world ontologies on the Web. The basic principle that we rely on – the fact that two different ontologies representing the same domain will describe concepts with (probably) a similar syntax and share similar attributes – leads us to believe that our approach will scale up to more complex ontologies.

Acknowledgments. The first author is supported by FCT (Fundação para a Ciência e a Tecnologia) under grant SFRH/BD/29773/2006.

References


