Towards to Similarity Identification to help in the Agents' Negotiation

Andreia Malucelli[1,2], Eugénio Oliveira[1]

[1] LIACC-NIAD&R, Faculty of Engineering, University of Porto, R. Dr. Roberto Frias, 4200-465 Porto, Portugal
malu@ppgia.pucpr.br, eco@fe.up.pt
[2] PUCPR – Pontifical Catholic University of Paraná, R. Imaculada Conceição,1155 80215-901 Curitiba PR, Brazil

Abstract. Enterprise delegates Agents' Negotiation is a simpler task if the enterprises involved in the transaction have homogeneous representation structures as well as the same domain of discourse, thus the use of a common ontology eases semantic problems. However, in real-life situations, real problems involve heterogeneity and different ontologies often developed by several persons and tools. Moreover, domain evolution, or changes in the conceptualisation might cause modifications on the previous ontologies once there is no formal mapping between high-level ontologies. We are proposing a method to be used by an Ontology-Services Agent to make Agents to understand each other despite their different ontologies. The method uses the natural language description of each involved item/product/service and combining statistical, clustering and suffix stripping algorithms finds out similarities between different concepts represented in different ontologies. Keywords: ontologies, multi-agent systems, similarity identification, negotiation.

1 Introduction

In a decentralized and distributed approach, interoperability refers to the way we communicate with people and software agents, the problems which hampers the communication and collaboration between agents. In B2B transactions, it is a simpler task if the enterprises involved in the transaction have homogeneous representation structures as well as the same domain of discourse, thus the use of a common ontology makes the communication problem easy. The use of a common ontology guarantees the consistency and the compatibility of the shared information in the system. However, in real-life situations, real problems involve heterogeneity and ontologies often developed by several persons continue to evolve over time. Moreover, domain changes or changes in the conceptualisation might cause modifications on the ontology. This will likely cause incompatibilities [1] and makes the negotiation and cooperation process difficult.

By making the enterprises agents interoperable, we enable them to meet the basic requirement for multilateral cooperation. There are two major types of cooperative interaction which may be identified in a multi-agent system: the first concerns which agents perform which tasks (the task allocation problem) and the second concerns the sharing of information (both results and observations on the outside world) between agents. Purpose heterogeneity is primarily concerned with the former type and semantic heterogeneity with the latter [2].

In B2B transactions, due to the nature of the goods/services traded, these goods/services are described through multiple attributes (e.g. price and quality), which imply that negotiation process and final agreements between seller and supplier must be enhanced with the capability to both understand the terms and conditions of the transaction (e.g. vocabularies semantics, currencies to denote different prices, different units to represent measures or mutual dependencies of products).

A critical factor for the efficiency of the future negotiation processes and the success of the potential settlements is an agreement among the negotiating parties about how the issues of a negotiation are represented in the negotiation and what this representation means to each of the negotiating parties.

Our objective is to help in the interoperability problem, enhancing agents with abilities to provide services to and accept services from other agents, and to use these services so exchanged to enable agents to effectively negotiate together. We are using Multi-Agent System as the paradigm for the system architecture since enterprises are independent and have individual objectives and behavior. The focus here, in this paper, is on ontologies, whose specification includes a term (item/product) that denotes the concept, their characteristics (attributes) with the correspondent types, a description explaining the meaning of the concept in natural language, and a set of relationships among concepts. It is a really weak form of integration, because integration is not the objective of our work. Our approach aims at creating a methodology that assesses semantic similarity among concepts from different ontologies without building on a priori shared ontology. It is one of the services provided [3] by an Ontology-Services Agent (OSAg) for trying to help during the agents' negotiation process.

Next section discusses heterogeneity, interoperability and ontology, including partial ontology examples. Section 3 presents the architecture of the proposed system. The similarity identification method is explained in the section 4 and finally we conclude the paper in section 5.

2 Heterogeneity, Interoperability and Ontology

Heterogeneity is both a welcome and an unwelcome feature for system designers. On the one hand heterogeneity is welcomed because it is closely related to system efficiency. On the other hand, heterogeneity in data and knowledge systems is considered an unwelcome feature because it proves to be an important obstacle for the interoperation of systems. The lack of standards is an obstacle to the exchange of data between heterogeneous systems [4] and this lack of standardization, which hampers communication and collaboration between agents, is known as the interoperability problem [5].

Heterogeneity here, in this paper, means agents communicating using different ontologies. Four types of heterogeneity are distinguished by [4]: (i) **paradigm heterogeneity,** occurs if two systems express their knowledge using different modeling paradigms, (ii) **language heterogeneity,** occurs if two systems express their knowledge in different representation languages, (iii) **ontology heterogeneity** occurs if two systems make different ontological assumptions about their domain knowledge, (iv) **content heterogeneity,** occurs if two systems express different knowledge.

Paradigm and language heterogeneity are types of non-semantic heterogeneity and the ontology and content heterogeneity are types of semantic heterogeneity.

In our proposed system each agent has its own private ontology although about the same knowledge domain, but each ontology was developed by different designers and may expresses knowledge differently.

In literature, ontologies are classified into different types based on different ideas. [6] presents two typologies, according to the level of formality and according to the level of granularity. According to *the level of formality*, three ontologies types are specified: (i) **informal ontology** is the simplest type; it is comprised of a set of concept labels organized in a hierarchy, (ii) **terminological ontology** consists of a hierarchy of concepts defined by natural language definitions, (iii) **formal ontology** further includes axioms and definitions stated in a formal language. According to the *level of granularity*, six ontologies types are specified: (i) t**op-level ontology** defines very general concepts such as space, time, object, event, etc., which are independent of a particular domain. (ii) **general ontology** defines a large number of concepts relating to fundamental human knowledge. (iii) **domain ontology** defines concepts associated with a specific domain. (iv) **task ontology** defines concepts related to the execution of a particular task or activity. (v) **application ontology** defines concepts essential for planning a particular application. (vi) **meta-ontology or generic or core ontology** defines concepts, which are common across various domains; these concepts can be further specialized to domain – specific concepts.

In our proposed system, the ontologies are classified in the level of formality as terminological ontologies because they include concepts organized in a hierarchy and the concept definitions are expressed in natural language. According to level of granularity they are classified as domain ontologies, in our case in the specific cars' assembling domain.

Cars' assembling domain is a suitable scenario because it involves several services suppliers' enterprises and consequently several different negotiations. To make it possible the cars' assembly, the service supplier enterprise (cars' assembler) needs to buy several parts/components. For each one of these parts/components there are several potential suppliers, which offer different prices, facilities, quality, delivery time, and others attributes. It is necessary to select among all the interested enterprises the ones which send the best offers and furthermore, it is mandatory a negotiation based on several criteria.

Even with terminology standards used by cars' factories, the same term may have different meanings, or the same meaning may be associated with different terms and different representations. A scenario using this domain will be explored as a study-case. The ontology creation process for our particular domain (cars' assembling domain) involved searching literature on cars' assembling domain and discussion with experts. After careful consideration and test of several different ontology building tools, we have selected the appropriated ones. First we have modeled our ontology by means of UML and then ontology-building tools WebODE [7], Protégé [8] and OntoEdit [9] have been used.

Fig. 1 and Fig. 2 show a part of a UML diagram of the built ontologies. Fig. 1 represents the Customer Enterprise Agent Ontology and Fig. 2 represents the Supplier Enterprise Ontology. Though example we may observe some differences causing interoperability problem during the negotiation process. For example, in the ontology A there are *wheel* and *handwheel* concepts and in the ontology B there is only the wheel concept, here meaning *handwheel*. Other differences as *Motor* x *Engine* and *Tire* x *Tyre* may be observed. The ontologies are composed by concepts, each concept has a set of characteristics, each one of the attributes has a type (not showed in this diagram) and each one of the concepts has relationship with other concepts. The way the Ontology-Services Agent, using a similarity-based algorithm, solves the problem is presented in Section 4.

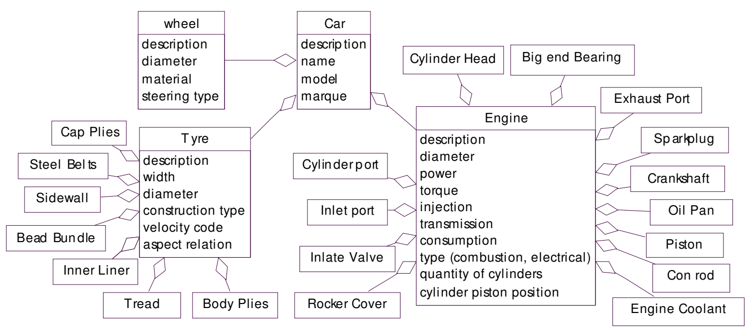
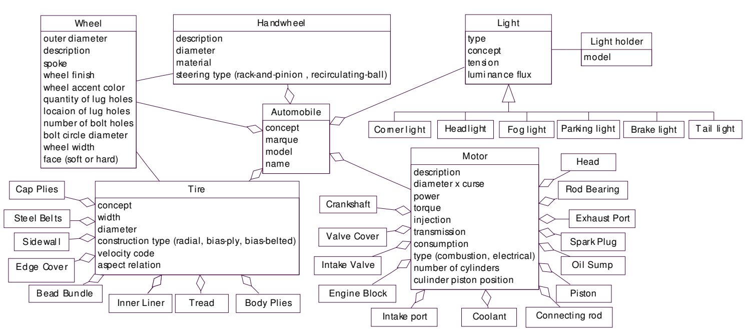


Fig.1. Ontology A: Part of the Customer Enterprise Agent Ontology



Fig, 2. Ontology B: Part of the Supplier Enterprise Agent Ontology

3 System Architecture

This framework includes 4 types of agents: facilitator agent, enterprise agents (good/product/services suppliers and customer), and ontology-services agent. The facilitator agent and enterprise agents - suppliers and customers, are cooperating together through a website with the objective of providing or getting goods/products/services, in collaboration, but keeping their own preferences and objectives. An ontology-services agent is involved in all the process for monitoring and facilitating the communication and negotiation between agents.

The **Facilitator Agent (FAg)** is the entity that matches the right agents and supports the negotiation process. The enterprise (customer and suppliers) agents and ontology-services agent have to register themselves to be able to communicate. Each agent has its own private ontology, built in a private and unknown (to the overall system) process. **Customer Enterprise Agents (CEAg)** represent enterprises interested in buying components to build a final product. Several suppliers in the world may have these components with different prices and conditions. Each CEAg sends a message (Identification of Needs) to the facilitator announcing which composed product/service is needed to configure. **Supplier Enterprise Agents (SEAg)** represent enterprises interested in providing some kind of product/service/good. Whenever a needed product, the facilitator agent conveys this announcement to all registered interested supplier enterprise agents. **Ontology-Services Agent (OSAg)** keeps monitoring the whole conversation trying to help when some message is not fully understood by some of the participants. The OSAg service for helping in the similarity identification is explained in the next section. Fig. 3 shows an instance of the multi-agent system. Each Enterprise Agent (Supplier or Customer) has its own architecture and functionalities (some developer will design and build the ontology with some tool and, later, the agent will access the generated file/database).

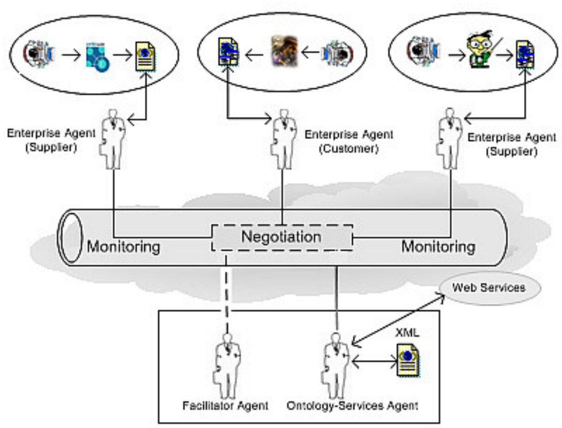


Fig. 3. System Architecture

4 Similarity Identification

Several different tools and techniques for mapping, aligning, integration, merging [10], [11], [12], [13], [14] of ontologies are available but there is no automatic way to do that. It is still a difficult task and for the success of these processes it is necessary to detect ontology mismatches and solve them. Recent research about ontological discrepancies have been done [4], [15], however none of the available tools tackle all the types of discrepancies [16].

Some problems in finding similarity are related to the following facts: (i) the ontologies use different concept/term names for the same meaning and description. Example: tyre and tire; (ii) the ontologies use the same concept/term name for different meaning and description. Example: wheel and wheel (where one of them means hand wheel), (iii) the ontologies use the same concept/term name for the same meaning. However, the description includes different characteristics (attributes) and relations.

Similarity evaluations among ontologies may be achieved if their representations share some components. If two ontologies have at least one common component (relations, hierarchy, types, etc) then they may be compared. Usually characteristics provide the opportunity to capture details about concepts. In our approach we are using relations and characteristics' types as common components in all the ontologies. There are a set of relations and characteristics that have to be known and used by all the ontologies for initial tests. The concepts are also linked by a number of relations.

The relations used in our approach are: (i) *part_of* relationship, organizes the concepts according to a decomposition hierarchy (composed_by), (ii) *is_a* relationship, a concept is a generalization of the concept being defined, (iii) *equivalent* relationship, the concepts are similar, (iv) *use* relationship, a concept uses functionalities from other concept.

The value types of characteristics used are: (i) *integer*, represents positive and negative integer values (ii) *string*, represents text information (iii) *discrete domain*, represents a set of fixed values (iv) *material*, represents information about what substance the object is made of (v) *numeric*, represents the not integer values.

The OSAg will be monitoring all the communication and negotiation and for helping it will search information in its own ontology, which is a basic ontology built with basic structures in the cars' assembling domain, which will be updated whenever a new concept is discovered. OSAg has also to get additional information from the agents using exchanged messages (see Fig. 4). An example of the structure of one exchanged message between OSAg and SEAg may be observed below, based on the ontologies showed in Fig. 1 and Fig. 2. In the message "ask-about", the OSAg is asking information about the engine concept, and in the "reply" message, the CEAg is answering the questions, filling the template. Each agent has to be able to read its own ontology and understand the template.

ask-about

```
:sender ontology-services agent
:receiver customer enterprise agent
:language KQML
:content (:concept (engine)
          :description (description)
          :part_of ( [ ]∨[concept] )
          :is_a ( [ ]∨[concept])
          :has_part ( [ ]∨[concept] )
          :equivalent ( [ ]∨[concept] )
          :use ( [ ]∨[concept] )
          :num_characteristics ( [ ] ∨ number)
          :num_integer ([ ]∨ number ∧ relevance)
          :num_numeric ([ ]∨ number ∧ relevance)
          :num_discrete_domain ([ ]∨ number ∧ relevance)
          :num_string ([ ]∨number ∧ relevance))
```

reply

```
:sender customer enterprise agent
:receiver ontology-services agent
:language KQML
:content (:concept (engine)
          :description (engine is a motor that converts thermal energy to mechanical work )
          :part_of (car )
          :is_a ([ ] )
          :has_part (inlate valve, rocker cover, cylinder port, inlet port, engine coolant, con rod, piston, oil pan, crankshaft, sparkplug, exhaust port, big end bearing, cylinder head)
          :equivalent ([ ])
          :use ( [ ] )
          :num_characteristics (10)
          :num_integer (6, 2)
          :num_numeric ([ ])
          :num_discrete_domain (2,2)
          :num_string (2,0))
```

Fig. 4. Exchanged messages to get aditional information

The process is described as follow:

1. CEAg sends a KQML message to the FAg informing about the basic requirement for that particular item/product/service.
2. FAg sends an announcement to the registered SEAg, which probably provide the required item.
3. Each one of the SEAg that provides the item/service required send an advertisement to the CEAg.
4. Some of the SEAg may have the announced item but may not understand it because SEAg may have a different ontology and the item may be specified in a different way. If the SEAg does not understand the announced item, it will send an "unknown" message to FAg. The OSAg, which is monitoring the communication, detects the message and try to help.
5. If 4 occurs, OSAg sends a message to CEAg asking for detailed information about the item required, as showed in the ask-about message example above.
6. After 5, OSAg will exchange messages with the correspondent SEAg asking for the concepts descriptions.
7. Using appropriated algorithms OSAg will find the correspondent concept to the announced item. This process is explained in the subsection 4.1.
8. If some description was not found or more than one was found as similar, new tests are needed to try to find proof of similarity. OSAg will exchange messages with the correspondent SEAg sending and asking for new informations using synonymous, relationships between concepts, type, quantity and relevance of the characteristics.

4.1 Using Description to Similarity Identification

We are proposing the use of metrics, methodologies and algorithms well known in database and information retrieval area for trying to find similarity among the words that compose the concept description.

Usually, in a specific domain, when experts are describing the concepts that form the ontology, they use some technical and specific words, and we may find similar words in the concept descriptions. We are proposing to select the relevant words used in the descriptions and to compare them to find similarities.

To make it easier to understand, consider the example of the KQML message above (Fig. 4), where OSAg asks information about *engine* (the required item), and the CEAg informs about the concept included described in its own ontology "**engine** is a **motor** that **converts thermal energy** to **mechanical work**".

First, it is necessary a process for selecting/extracting the most representative words (showed in bold) from the description, which will represent the concept engine.

engine	motor	converts	thermal	energy	mechanical	work

The OSAg will also extract the most representative words from the description of the concepts in the ontology B, to have also a representation of the concepts. As an example, we now consider two other concepts, *motor* and *tire*, to be compared with engine.

Motor "it is a **machine** that **converts** other **forms** of **energy** into **mechanical energy** and so **imparts motion**".

machine	converts	forms	energy	mechanical	imparts	motion

Tire "**consists** of a **rubber ring** around the **rim** of an **automobile wheel**".

consists	rubber	ring	automobile	wheel

The use of similarity algorithms between the required concept and the candidate concept would not give representative results, because we have semantic similarity and comparing strings would only work for cases as tire and tyre comparison. Using, for example, edit distance [17] for comparing the strings engine and motor we will get the similarity (1-6/6) = 0 and comparing engine with tire we will get the similarity (1-3/6) = 0.5, where engine and motor have the same meaning and should have a higher similarity value. For solving this problem our purpose is to use a combination of methods to find similarities between the words extracted from the descriptions, and some weights are used for the most representative words.

A similarity matrix is generated between the set of words extracted from required concept description with each one of the set of words extracted from the candidates concepts descriptions. We have two similarity matrix in this example, one among the words extracted from engine and motor descriptions, and another one built with words extracted from engine and tire descriptions. The matrix has its values calculated using edit distance and suffix stripping [18] algorithm.

We are using also in our algorithm, weights for the most relevant words. In the case of a similarity between words equal to 1, a sum of the weight equal to 1 is attributed for the correspondent value in the matrix, and if the required concept word (engine) is contained in the description of the candidate concept, this word gets a weight value of

1 and a result is summed with all the values in the matrix. To have the final result we calculate the matrix sum, but due to the matrix size difference, it is necessary calculate the average, sum the matrix elements and divide by the number of matrix elements.

In our example we got the similarity value between engine and motor of 0.35, where we found 3 identical words and the candidate concept in the description of the required concept. It concludes a difference due to the weights. The similarity value between engine and tire is 0.06. The method calculation shows that engine and mo-tor concepts are more similar than engine and tire.

5 Conclusions

We have proposed a heterogeneous multi-agent architecture suitable for semantic interoperability. Each agent has its own private ontology although in the same knowledge domain. Also each ontology is developed by different designers and expresses knowledge differently. The ontologies are classified regarding the level of formality as terminological ontologies, once their concepts are organized in a hierarchy and the concept definitions are expressed in natural language. According to the level of granularity, they are classified as domain ontologies, in our case in the specific car's assembling domain.

Our approach aims at creating a methodology for extracting similarities from the concept descriptions of the required item and the candidate items to find which one of those candidates may be the requested one. Each agent accesses its own ontology, without building any a priori shared ontology, and sends the needed information to the specific Ontology-Service Agent (OSAg). Relationships among concepts, characteristics, types and synonymous are also important information and may help in the process if the natural language description is not enough to identify the similarities.

A similarity matrix is generated between the description of the required item and the descriptions of the candidate concepts. The matrix has its values calculated using edit distance and suffix stripping algorithm. Rand Statistic is calculated to compare and find out the most promising candidate concept that matches the former unknown concept.

References

1. Klein, M., Kiryakov, A. Ognyanoff, D. and Fensel, D.: Finding and specifying relations between ontology versions. In Proceedings of the ECAI-02 Workshop on Ontologies and Semantic Interoperability. Lyon, July 22. 2002
2. Roda, C., Jennings, N.R., Mandanu, E.H.: The impact of heterogeneity on cooperating agents. In Proceedings of AAAI Workshop on Cooperation among Heterogeneous Intelligent Systems, Anaheim, USA, 1991
3. Malucelli, A., Oliveira, E., Ontology-Services to Facilitate Agents' Interoperability. In Proceedings of the Sixth Pacific Rim International Workshop on Multi-Agents (PRIMA

2003). Eds. Jaeho Lee, Mike Barley, Springer-Verlag. LNAI 2891, Korea. Novembro. (2003) 170-181.

4. Visser, P.R.S., Jones, D.M., Bench-Capon, T.J.M. and Shave, M.J.R.: An Analysis of Ontology Mismatches; Heterogeneity vs. Interoperability. AAAI'97 Spring Symposium on Ontological Engineering, Stanford (1997)
5. Willmott, S., Constantinescu. I, Calisti, M.: Multilingual Agents: Ontologies, Languages and Abstractions. In Proceedings of the Workshop on Ontologies in Agent Systems. 5th International Conference on Autonomous Agents. Montreal. Canada. (2001)
6. Kavouras, M.: A Unified Ontological Framework for Semantic Integration. In Proceeding of the International Workshop on Next Generation Geospatial Information, Cambridge (Boston), Massachusetts, USA, October (2003) 19-21
7. Arpírez, J.C., Corcho, O., Fernández-López, M., Gómez-Pérez, A.: WebODE in a nutshell. AI Magazine 24(3):37-48. Fall (2003)
8. Gennari, J., Musen, M.A., Fergerson, R. W., Grosso, W.E., Crubézy, M., Eriksson, H., Noy, N.F., Tu, S.W.: The Evolution of Protégé: An Environment for Knowledge-Based Systems Development The Evolution of Protégé: An Environment for Knowledge-Based Systems Development. Technical Report. SMI Report Number: SMI-2002-0943 2002.
9. Sure, Y., Erdmann, M., Angele, J., Staab, S., Studer, R., Wenke, D.: OntoEdit: Collaborative Ontology Development for the Semantic Web. In Proceedings of the First International Semantic Web Conference (ISWC 2002), June 9-12 Sardinia, Italia. 2002.
10. Hakimpour, F., Geppert, A.: Resolving Semantic Heterogeneity in Schema Integration: an Ontology Based Approach. In the Proceedings of the International Conference on Formal Ontology in Information Systems (FOIS 2001) Maine, USA (2001) 297-308
11. McGuinness, D.L., Fikes, R., Rice, J., and Wilder, S.: An environment for merging and testing large ontologies, in Cohn, A., Giunchiglia, F. Selman, B. (eds.), KR2000: Principles on Knowledge Representation and Reasoning, San Francisco (2000) 483-493
12. Noy, N.F. and Musen, M.A.: PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment. In Proceedings of AAAI'00, Austin (2000)
13. Pinto, H.S.: Towards Ontology Reuse. In Proceedings of AAAI99's Workshop on Ontology Management, WS-99-13, AAAI Press (1999) 67-73
14. Stumme, G and Maedche, A.: Ontology Merging for Federated Ontologies on the Semantic Web. Workshop on Ontologies and Information Sharing. IJCAI'01, Seattle, USA (2001)
15. Klein, M.: Combining and relating ontologies: an analysis of problems and solutions. Workshop on Ontologies and Information Sharing, IJCAI'01, Seattle, USA (2001)
16. Hameed, A. Sleeman, D., Preece, A.: Detecting Mismatches Among Experts' Ontologies Acquired through Knowledge Elicitation. Research and Development in Intelligent Systems XVIII (Proceedings of ES 2001), BCS Conference Series, Springer-Verlag (2001) 9-22
17. Sahinalp, S.C., Tasan, M., Macker, J., Ozsoyoglu, Z.M.: Distance Based Indexing for String Proximity Search. In Proceedings of 19th International Conference on Data Engineering. Bangalore, India. (2003) 125
18. Porter, M.F.: An algorithm for suffix stripping. Readings in information retrieval. Morgan Kaufmann Publishers Inc. San Francisco, CA, USA (1997) 313-316