Developing Semantically Interoperable E-Commerce Systems

Jurriaan van Diggelen, Frank Dignum
Institute of Information and Computing Sciences
Utrecht University, the Netherlands
{jurriaan, dignum}@cs.uu.nl

ABSTRACT

This paper discusses semantic interoperability issues in agent-based E-commerce systems. The literature reports various techniques to enable agents to understand the meanings of the messages exchanged. We will argue how these different techniques can be combined in one agent communication protocol to obtain the best of each world. The resulting communication protocol enables agents to sufficiently understand each other to participate in successful collaboration.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*multiagent systems, coherence and coordination*; D.2.12 [**Software Engineering**]: Interoperability—*data mapping*

General Terms

Standardization, Languages

Keywords

Electronic Commerce, Semantic Interoperability, Ontologies

1. INTRODUCTION

With the advent of the internet, a physical connection has been established between computers from all over the world. This offers unprecedented opportunities for electronic commerce. One of these opportunities is described in a future scenario presented by AgentLink [16]. It envisions a fully autonomous software-based travel agent assisting a European customer in planning a holiday trip to the United States. The travel agent takes a number of tasks upon itself, such as finding a cheap flight, arranging transport in the US and booking accommodation. The agent does not blindly follow orders. It suggests other possibilities when appropriate and revises previous plans when unexpected situations arise.

The travel agent must be capable of interacting with other agents in an open and dynamic environment. This raises a number of interoperability issues. Many of these issues have already been addressed by global standards. For example, TCP/IP has been adopted as a standard to establish network interoperability and XML has become a standard for achieving syntactic interoperability. However, the complex type of interaction required by the travel agent scenario raises the need for yet another type of interoperability, i.e. *semantic interoperability*. This means that agents should also be capable of understanding the *meaning* of the messages exchanged. For example, the travel agent might encounter another agent representing a French airline company that refers to a *flight* as a *voyage en avion*. If the travel agent does not know that a *voyage en avion* means the same as a *flight*, misunderstandings could arise which would obstruct their cooperation.

Traditionally, semantic interoperability problems have been attacked in the same way as all other interoperability problems, i.e. by standardization. This corresponds to the most popular definition of an ontology: an ontology is a specification of a shared conceptualization [13]. Different system components sharing the same ontology can exchange their knowledge fluently as their knowledge representations are compatible with respect to the concepts regarded as relevant and with respect to the names given to these concepts. Nevertheless, for e-commerce applications, ontology standardization has fundamental limitations. Mainly, this is because a standardized ontology forces an agent to abandon its own world view and adopt one that is not specifically designed for its task [6]. This may result in a suboptimal situation.

To overcome this problem, various proposals have been made. Ontology alignment has been proposed as a technique that enables agents to keep their individual ontologies by making use of *mappings* between the different ontologies [25]. An ontology agent [10] has been proposed to facilitate agent communication by registering ontologies and performing services such as translating between ontologies. Ontology Negotiation has been proposed as a technique to enable agents to reconcile their heterogeneous ontologies during their conversation [4, 34].

In this paper, we will not speak our for one particular approach as each of them has its advantages and disadvantages. Rather, we will argue how the different techniques can be combined in one agent communication protocol to obtain the best of each world. Central to our approach is that not every agent needs to understand every other agent *completely*. The speaker should only be capable of communicating the information that is relevant to the hearer. This

ICEC'07, August 19–22, 2007, Minneapolis, Minnesota, USA.

prevents that ontology reconciliation requires every agent to know every other ontology in the world, which would clearly be unfeasible.

The paper is organized as follows. Section 2 discusses the ontology standardization approach. Given the current practice, we will set out some requirements for agent communication. Section 3 discusses the ontology alignment approach. We will show how this technique can be used to overcome the shortcomings of the standardization approach and argue which requirements this imposes on agent communication. Section 4 discusses dynamic ontology reconciliation and the requirements for agent communication. Section 5 introduces the agent communication protocol which meets the requirements set out in the previous sections. We conclude in Section 6.

2. ONTOLOGY STANDARDIZATION

Many authors have stressed that an ontology captures consensual knowledge, which is not private to an individual, but accepted by a group [30]. This corresponds to how ontologies were originally conceived in the early 90s, namely as a specification of a shared conceptualization [13]. At the time, ontologies were used to support *knowledge sharing and reuse* [22]. They were envisioned as key components to realize the goals of the Knowledge Sharing Effort [14], an initiative to enable libraries of reusable knowledge components that can be invoked over networks. Using these libraries, developers would no longer have to start their knowledge bases from scratch. Instead, they could assemble a number of reusable components (such as reasoning modules and ontologies), and focus on the specialized task which would be new to their specific system. A standardized ontology can be said to enable knowledge sharing as it provides a way to specify content-specific agreements. Once the ground rules for modeling a domain are specified in an ontology, multiple parties can agree to adhere to that specification. This would enable them to share knowledge, as their knowledge bases would all have the same underlying structure.

This vision has stimulated various types of research. A very ambitious approach is to develop an *all-purpose* ontology as pursued in the Cyc project [15]. If the Cyc ontology would indeed be usable for all purposes, it would be the ideal candidate for a standard ontology of all agents. This would solve all semantic interoperability problems, as every agent would share one and the same ontology: the Cyc ontology. However, it is highly unlikely that the Cyc project will succeed in the goal of creating an ontology that is applicable to all e-commerce applications. The all-purpose ontology would need to contain all product names of every company in the world, which is hardly conceivable. Furthermore, even if it would be feasible to create such an ontology, it would be too large to be practicable. To illustrate this point, according to a recent article [17], the Cyc knowledge base currently contains more than 2.2 million assertions and more than 250,000 terms.

A less drastic way to standardize ontologies are top-level ontologies. A top-level ontology describes very general concepts that are the same for every domain and are not dependent on task and purpose. The aim is to use this ontology as a ground for more specialized (task-specific) ontologies. This already creates some degree of interoperability between different components that are based on the same upper ontology. Examples of top-level ontologies are the upper-level ontology by Sowa [28], SUMO [23] and BFO [11].

Mostly, a standard ontology is developed for a specific domain. For example, standardized ontologies have been proposed for the financial domain [21], for the travel domain [1] and for representing information about the crew of a space shuttle [2]. Domain specific ontologies are made publicly available in ontology libraries (e.g. ontolingua [8]). In this way, a developer can easily select one or more ontologies that best suit the needs of an agent. Developers can guarantee semantic interoperability between agents by agreeing to use the same standardized ontology. Furthermore, even without such an agreement between developers, two agents representing knowledge about the same domain could coincidentally use the same underlying standard ontology. It is expected that, when building software with ontologies becomes common practice, every domain will have its own standard and widely used ontologies. Agent developers will be eager to equip their agents with a popular ontology to make them interoperable with as many other agents as possible.

2.1 Using standardized ontologies in agent communication

Given the current practice of applying standardized ontologies, we will now discuss the consequences for agent communication. If every agent would use the same standard all-purpose ontology, ontology issues would require little attention from the agent communication community. However, as has been argued above, this scenario is highly unlikely. We make the following two observations. Firstly, ontology standardization typically leads to partially shared ontologies, not necessarily to fully shared ontologies. This is the case for top-level ontologies. Also when agents use multiple domain-specific ontologies, some of the ontologies may be shared, while others may not. Secondly, ontology standardization may lead to ontologies being shared without the agents being aware of this. Two agents may use the same standard ontology, because their developers independently chose the same popular standard ontology for modeling that domain.

Figure 1 shows two example ontologies of agents Ag-1 and Ag-2 that could result from an ontology standardization effort. We assume simple ontologies consisting of concepts in a subconcept hierarchy. An arrow between two concepts represents a subconcept relation (and against the flow a superconcept relation). Two concepts in two different branches in the ontology are disjoint. Concepts that are equivalent or overlap are connected with a line with the $\equiv$ or $\oplus$ symbol in it. For readability, we have left out concept relations that are derivable from other relations.

In the context of ontology standardization, a number of other things are noteworthy. Firstly, the concepts are prefixed with the ontology name followed by a colon, i.e. they are namespaced. This avoids naming conflicts by ensuring that concepts from different ontologies always have different names. Secondly, the ontologies are composed of two standardized ontologies: *so1* for representing flight information and *so2* for representing location information. Furthermore, the ontologies are extended with additional concepts from ontologies *o1* and *o2* that are required to meet the domain-specific needs of the agents. In this way, the phenomena discussed earlier are clearly present in this example.

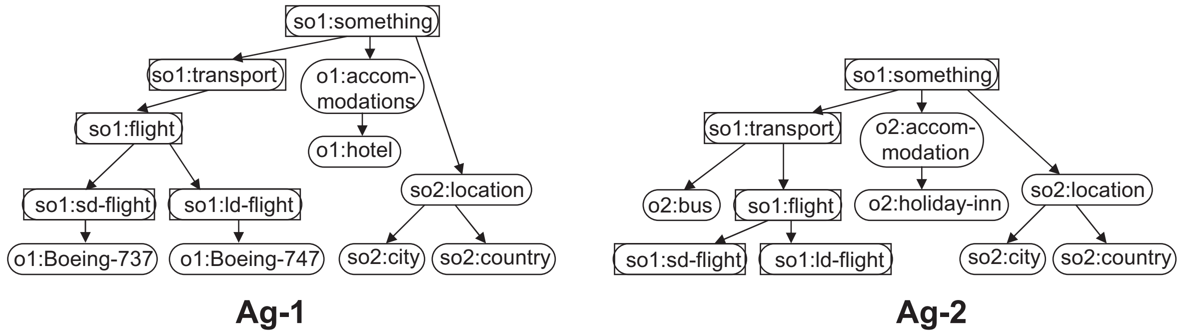

Figure 1: Example scenario of ontology standardization

The ontologies are not fully shared. For example, the developers did not agree on a standardized ontology for modeling the concept accommodation. Consequently, the concepts *o1:accommodation* and *o2:accommodations* are not shared. Also, Ag-1 has added some specific concepts to its ontology such as *o1:Boeing-737* and *o1:Boeing-747*. The shared concepts that the agents' developers have agreed upon are represented with a box around them. Examples of these are the concepts in ontology *so1*. We will refer to these as *common concepts*. So-called *unknowingly shared* concepts are known by both agents without the agents being aware of this, i.e. they are shared without explicit agreement. Examples of unknowingly shared concepts are the concepts in *so2*.

2.2 Requirements for agent communication

For an agent communication protocol to deal with these phenomena, a number of requirements can be identified. These requirements are listed below:

Requirement 1.

1. *Agents should maximally exploit the potential use of common concepts*
2. *Agents should exploit the use of unknowingly shared concepts.*

The first requirement can be fulfilled by allowing the speaker to compose its message using a more general concept than what it actually intended to convey. For example, Ag-1 should be allowed to translate its concept *o1:Boeing-737* to the more general common concept *so1:sd-flight* while sending a message to Ag-2. From the perspective of Ag-2, there is no information loss, as its ontology cannot contain the more specific concept *o1:Boeing-737* anyway. The benefit of message generalization is that Ag-2 can understand *sd-flight* while it cannot understand the specific concept *Boeing-737*.

The second requirement can be fulfilled by allowing the speaker to use non-common concepts (which may turn out to be unknowingly shared). For example, Ag-1 should be able to use *so2:location*, which happens to be understandable to Ag-2. However, if Ag-1 would use the concept *o1:Boeing-737* which is not understandable to Ag-2, the agents should find a way to convey the information differently. When the agents have discovered that a concept is unknowingly shared, the concept instantaneously becomes a common concept.

We will come back to these issues when we will discuss the complete communication protocol in Section 5.

2.3 Limitations

It is widely agreed that ontology standardization cannot solve all semantic interoperability problems. The limitations can best be understood by recognizing that ontologies are highly task-dependent [6]. An agent needs an ontology that is tailored to its own task. By assembling standardized ontologies for the purpose of interoperability, this aspect of ontologies is not taken seriously. The agent soon gets burdened with a huge ontology that is for the largest part composed with concepts that are irrelevant for the agents task. For example, to solve all semantic interoperability problems in an e-commerce setting, an agent would have to be equipped with an ontology that is composed of every ontology in the system. Clearly, such an ontology would be too large to be of practical use.

In the next section, we will discuss a technique that can be used to overcome these problems.

3. ONTOLOGY ALIGNMENT

Ontology alignment allows the agents to maintain their own task-specific ontologies. Communication is enabled by a set of pre-defined *mappings*, which specify the relations between the agents' ontologies. The mappings are mostly created by humans. For this purpose tools have been developed that assist people in mapping ontologies. Examples of such tools are Chimaera [19], Prompt [25] and FCA-Merge [31].

Two different topologies can be used to implement ontology mappings: point-to-point mappings and using an intermediate ontology. In the point-to-point topology, a mapping is defined for every pair of ontologies. The speaker uses these mappings to translate directly from its own ontology to the hearer's ontology. The disadvantage of the point-to-point topology is that many mappings have to be established in order to align the agents' ontologies. To reach complete interoperability in a system of n agents, $\frac{1}{2}n^2 - n$ mappings must be established (equal to the number of non-reflexive, non-directed arches in a fully connected graph). A way to reduce the number of mappings is to use an *intermediate ontology* or *interlingua* [7] which indirectly aligns the agents' ontologies. In this way, the speaker may translate from its own ontology to the intermediate ontology, which the hearer translates again to its own ontology.

Concepts can be mapped in different ways. One way is to state that two concepts are equivalent, i.e. an equivalence mapping. Another way of concept mapping uses *into map-*

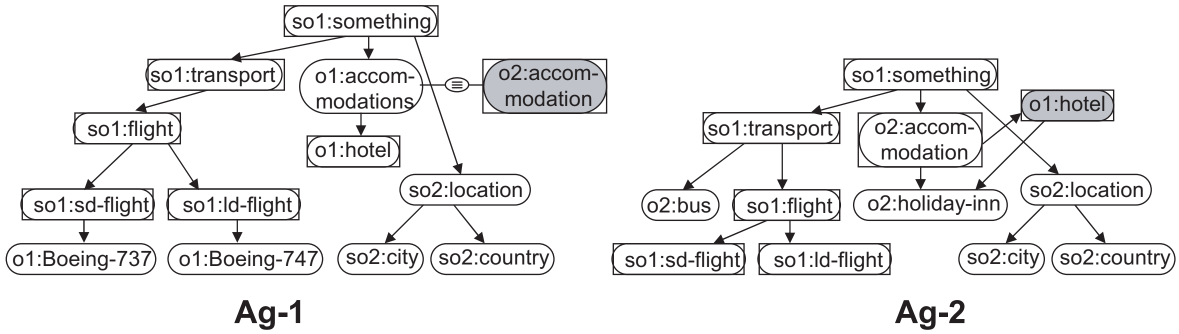

Figure 2: Example of combined ontology standardization and alignment

pings and *onto mappings* [29, 5]. An into mapping states that one concept is more specific than the other, whereas an onto mapping states that one concept is more general than the other.

In our approach, we avoid the introduction of specialized mapping operators by introducing a special type of concept, i.e. an *acquired concept*. Acquired concepts are defined additionally to the concepts already present in the agent's ontology, i.e. the agent's *native concepts*. Contrary to the agent's native concepts, acquired concepts are not used for storing knowledge and reasoning. They only serve as a way to specify a mapping between agents' ontologies. Figure 2 illustrates the use of acquired concepts for ontology alignment, where acquired concepts are represented as shaded.

An equivalence mapping can be easily represented by adding an acquired concept to the ontology which is defined as an equivalent concept with one of the native concepts. For example, Figure 2 represents that *o1:accommodations* is equivalent with *o2:accommodation*. An into mapping can be represented by adding an acquired concept as a subconcept of a native concept (e.g. *o1:hotel* is mapped into *o2:accommodation*). An onto mapping can be represented by adding an acquired concept as a superconcept of a native concept (e.g. *o1:hotel* is mapped onto *o2:holiday-inn*).

Acquired concepts can be used both for representing point-to-point mappings as well as for representing an intermediate ontology. The first topology can be realized by adding the native concepts of one agent's ontology as acquired concepts to the other agent's ontology, for every pair of agents. The second topology can be realized by first establishing a distinguished set of concepts, and adding these as acquired concepts to all agents' ontologies.

The benefit of using acquired concepts to represent concept mappings is that the standard facilities for ontological reasoning are applicable. Nevertheless, it imposes some requirements on agent communication, which will be discussed next.

3.1 Requirements for agent communication

Because acquired concepts are not used for knowledge representation and reasoning, an agent should only use acquired concepts for communication purposes. This requirement is stated below.

Requirement 2.
Agents should only apply acquired concepts for communication purposes.

This requirement can be straightforwardly met by specifying that the hearer of a message translates acquired concepts to its native ontology. The speaker is allowed to use acquired concepts in its messages in the same way as it would use native concepts.

One might raise the objection that an agent should also be allowed to represent its knowledge in terms of acquired concepts, i.e. that the ontology of the knowledge base should include acquired concepts. This objection is misguided for the following reason. The knowledge-base of an agent is not an independent component, but is strongly interwoven with the agent's internal workings. Changing the ontology of the knowledge base would also require a change in the agent's action selection criteria and goal selection criteria and so forth. Adding the relations of an acquired concept with the agent's native concepts is one thing, but specifying how the agent should use an acquired concept in planning and goal selection is far more difficult. Thus, the application of acquired concepts is, *by definition*, restricted to communication.

3.2 Limitations

The use of ontology alignment for e-commerce systems suffers a disadvantage: it presumes that all mappings can be established at design-time, before the agents start interacting. This assumption does not always hold because of two reasons. Firstly, in an *open* system, it is not known beforehand which agents will constitute the system. It is therefore impossible to state at design-time which ontology mappings are needed at agent interaction time. Secondly, it is not clear when design-time stops and interaction time starts. As many development methodologies point out [27], system development should proceed in a number of iterative cycles. According to the so-called *development cycle*, improvements are implemented in each cycle based on the findings from the previous cycle. This means that agents, and their ontologies in particular, also evolve at interaction time [24]. Ontology alignment is not flexible enough to deal with changing ontologies, as it assumes that one moment exists after which the agents' ontologies no longer change.

4. DYNAMIC ONTOLOGY RECONCILIATION

Some concepts may still not be understandable to other agents after the ontology standardization and alignment effort. In Figure 2, *o1:Boeing-737* and *o2:bus* are such concepts. If the problem is insurmountable, the agents must be able to reconcile the ontological difference at agent interaction time. A wide range of proposals exist which can be applied for this purpose. Before we discuss how dynamic ontology reconciliation techniques can be incorporated in agent communication, we will discuss two categories of these techniques: ontology mediators and automatic ontology mapping techniques.

4.1 Ontology mediators

One way to obtain an ontology mapping at agent interaction time is to use a *mediation service* [35], or *ontology agent*. According to the FIPA Ontology Service Specification [10], an ontology agent facilitates agent communication by registering ontologies, generating mappings between ontologies, and by making translations between ontologies. In other words, an ontology agent provides a central point which can be consulted by agents with communication problems. This approach opens the possibility to reconcile heterogeneous ontologies at agent interaction time, thus being more flexible than ontology alignment. Other approaches that use mediation services are KRAFT [26] and OBSERVER [20]. Although the mediation approach might appears as an attractive solution for interoperability problems in multi-agent systems, it does not solve the problem of how to establish these mappings themselves. Basically, it relocates the problem from the individual agents to the ontology agent. Furthermore, the ontology agent must be trusted by the agents using them to provide the correct ontology mappings.

4.2 Automatic ontology mapping techniques

When no (trusted) ontology mediator is present to resolve the ontological heterogeneities, the agents must be able to establish an ontology mapping themselves. This can be done by using automatic ontology mapping techniques. These techniques can be subdivided in the following four categories: extensional methods, terminological methods, structural methods, semantics-based methods [18]. We will briefly discuss each of them below.

Extensional methods compare the instances of classes. For example, *o1:accommodations* can be derived to be equivalent to *o2:accommodation* by observing that all instances of *o1:accommodations* (e.g. URL's of the web pages of accommodations) are also instances of *o2:accommodation* (see [33] for an application with news agents). For this technique to work, a number of criteria must be met. Firstly, the instances must be represented uniformly. For example, the technique fails if one agent uses a URL to represent an instance of accommodation, and the other agent uses a string with the name of the accommodation. Secondly, the agents must be able to compare instances of classes. Either, the instances in their knowledge bases overlap, or they can classify unseen instances to classes in their ontology. For example, suppose that *o1:accommodations* contains only instances of accommodation in the United States, and *o2:accommodation* contains only instances of accommodation in Great Britain. If Ag-2 cannot recognize that an instance of *o1:accommodations* qualifies as an instance of *o2:accommodation*, the technique breaks down.

Terminological methods compare the strings that are used as the names of classes. The methods range from finding a common substring in two strings, using Wordnet [9], to using translation dictionaries to translate between different languages. Whereas these approaches might produce good results on some occasions, they presume that the developers of the ontologies have chosen self-evident names. In an e-commerce setting, the ontology may very well contain product names that are not self-evident at all and do not occur in Wordnet or translation dictionaries, e.g. *DC10*. Terminological methods fail on such occasions.

Structural methods compare the structures of classes in an ontology. For example, if two concepts have the same number of attributes, they are judged as being similar to a certain degree. Such methods, perhaps in combination with other methods, might help in finding the right ontology mappings, but do not provide reliable evidence. For example the concepts *so2:location*, *so2:city*, *so2:country* are structured in exactly the same way as *so1:flight*, *so1:sd-flight*, *so1:ld-flight* (one superconcept, two subconcepts).

Semantics-based methods make use of the formal semantics of ontology languages to derive ontology mappings. For example, subsumption or satisfiability tests are used, which are well-studied reasoning tasks in description logic. The limitations of these methods are that they only become usable after a preprocessing phase in which a number of concept mappings are declared. When no relations at all are known to exist between ontologies, these techniques are not usable to derive ontology mappings.

4.3 Requirements for agent communication

From Section 4.1 and 4.2, we can conclude that no universally applicable technique exists to dynamically reconcile heterogeneous ontologies. Ontology mediators may not always possess the desired ontology mapping, and different techniques for automatic ontology mapping have a different scope of application. What the different techniques have in common, is that they are resource-consuming. Automatic ontology mapping techniques are usually based on machine learning techniques and require much computing power. Finding an ontology mediator that knows the correct mapping may also be a difficult task. Additionally, it is well conceivable that such ontology agents will charge money for their services.

In this paper, we will not commit to any particular dynamic ontology reconciliation technique. Rather, we will argue how such a technique should be embedded in agent communication, knowing that it is resource consuming. In our framework, dynamic automatic ontology reconciliation boils down to adding an acquired concept at run-time. We call this process concept explication. To improve efficiency, the agents should only apply concept explication when strictly needed. This is stated in the following requirement.

REQUIREMENT 3.
Agents should explicate concepts when needed and only when needed

As argued in Section 2.2, the speaker is allowed to state its message in more general common concepts, when what it really intends to convey can only be expressed in non-common concepts. As has been argued, the resulting information loss may not be noticeable for the receiver. For example, there is absolutely no need for Ag-2 to learn the concepts *o1:Boeing-737* and *o1:Boeing-747* as this would not add anything to what can be communicated between Ag-1 and Ag-2. Sometimes, however, the information loss affects the effectivity of communication. In the extreme case, the agents would communicate everything using the common concept *so1:something*. Clearly, such communications would lead to an intolerable amount of information loss. Therefore, when the speaker sends a message which is overgeneralized, the hearer should recognize this and only then apply concept explication. After the speaker has taught the concept to the hearer, the hearer adds the concept as an acquired concept to its ontology. This extra common concept allows the message to be sent more precisely next time. The precise workings of this mechanism are discussed in the following section.

5. AGENT COMMUNICATION PROTOCOL

In this section, we will give the precise workings of the agent communication protocol that meets the requirements 1, 2 and 3. An overview of the communication protocol is given in Figure 3.

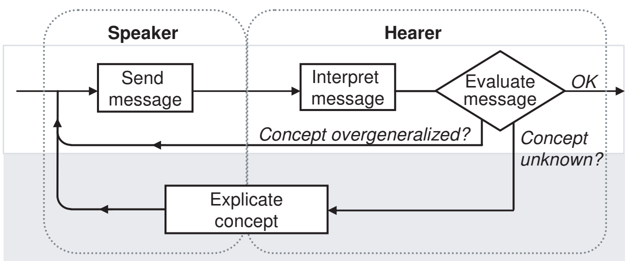

Figure 3: Communication protocol

As has been argued in Section 4, we will not give a general specification of how the agents explicate concepts. Depending on the domain, an appropriate method for concept explication must be implemented. For *Send message*, *Interpret message* and *Evaluate message*, we will give generic template decision procedures which should be used by every agent in the system. In discussing the protocol, the following terminology is useful. concept c is called a *particularization* of d if c is a subconcept of d or c overlaps with d. A concept c is called an *implied concept* of d if c is a superconcept of d or c is equivalent with d.

We will now specify *Send message* as follows:

Specification 1. *Send Message*
The speaker translates what it intends to convey to:

- *the most specific common implied concept, or*
- *an implied concept that is more specific than the most specific common implied concept*

As argued in Section 2.2, the speaker should be allowed to compose its message in more general common concepts, in order to make maximal use of common concepts. This is specified in the first option. However, when multiple common superconcepts are available, the speaker must choose the *most specific* one. For example, when Ag-1 intends to convey *o1:Boeing-737*, it may use the most specific superconcept *so1:sd-flight*, but it may not use the common superconcept *so1:flight*. In this way, the speaker minimizes information loss.

Requirement 1.2 states that the speaker should exploit the use of unknowingly shared concepts. This is specified in the second option of Specification 1. However, the speaker is only allowed to use a non-common concept in its message when this enables it to convey its message more specifically than would be possible using a common concept. Therefore, the specification states that if the agent uses a non-common concept, its should be more specific than the most specific common implied concept. In this way, the specification avoids incomprehension by the hearer as much as possible.

Message interpretation is specified as follows.

Specification 2. *Interpret Message*
The hearer translates the concept in the message to the most specific implied native concept.

As described in Requirement 2, the agent should only use acquired concepts in communication. Therefore, when the agent interprets a message, it translates the concept to its native ontology. To minimize information loss, the hearer must translate to the *most specific* implied native concept.

Because a concept in a message may be unknown or overgeneralized, the hearer must evaluate the message and provide an appropriate response. Message evaluation is specified as follows.

Specification 3. *Evaluate message*
Let c be the concept which is used in the message.

- *If c does not occur in the hearer's ontology, Then the hearer evaluates "Concept unknown"*
- *If c is sent using "Inform" and the hearer's ontology contains concepts which are*
  - *native particularizations of c, and*
  - *are not a common subconcept of c, and*
  - *are not a subconcept of a common subconcept of c*

  Then the hearer evaluates "Concept overgeneralized"
- *Else the hearer evaluates "OK"*

The first rule regards unknown concepts. When the speaker uses a non-common concept in the message which is not understood by the hearer, the hearer replies "Concept unknown", after which the speaker explicates the concept.

The second rule regards concepts that are known but might be overgeneralized. There are two methods for ensuring that the messages are not overgeneralized.

The first method is performed by the speaker. If it translates what it does not state its message in more general terms, it lets this be known by sending a message of the "ExactInform" type. If, on the other hand, it sends a message that is stated in more general concepts, it sends a message of the "Inform" type. When an "ExactInform" message is sent, the hearer knows that the message is not overgeneralized.

The second method applies when an "Inform" message is sent, in which case it is more difficult to recognize overgeneralization. The hearer recognizes overgeneralization by reasoning as follows. Upon hearing that an object belongs to the concept used in the message, it knows that the object is a member of every implied concept of the concept in the message and that the object is not a member of all concepts that are disjoint with the concept in the message. This knowledge cannot be a symptom of overgeneralization. However, the hearer remains ignorant about the particularizations of the concept used in the message. This ignorance may be a symptom of overgeneralization. Thus, native particularizations may indicate overgeneralization. This is stated in the first condition of the second rule.

However, not all native particularizations indicate overgeneralization. Remember that Specification 1 requires the speaker to use the *most specific* common implied concept (or to a non-common subconcept of that). The aim of this measure was to prevent the speaker from becoming more general than necessary. However, it can also be used to enable the hearer to form a belief about what the speaker intended to convey and what it did not. In philosophy of language, such a derivation is known as a *conversational implicature* [12]. In our proposal, it works as follows: the hearer knows that the speaker does *not* intend to convey the common subconcepts of the concept used in the message, otherwise it would have used a different concept in the message. It therefore knows that these concepts do not indicate overgeneralization. This is stated in the second condition of the second rule. Moreover, the hearer can reason as follows. It also knows that the speaker did *not* intend to convey any non-common subconcepts of the common subconcept of the concept used in the message. This is stated in the third condition of the second rule.

EXAMPLE 1.

1. *Consider the ontologies in Figure 1. Ag-2 intends to convey* o2:accommodation. *Ag-2 translates to the most specific common implied concept* so1:something *and sends an Inform-message. Ag-1 replies that "Concept overgeneralized", because its ontology contains the native particularization* o1:accommodations. *Ag-2 translates* o2:accommodation *to the non-common concept* o2:accommodation *and sends an ExactInform-message. Ag-1 replies "Concept unknown". Ag-2 explicates the concept.* o2:accommodation *is added to Ag-1's ontology as an acquired common concept which is equivalent to* o1:accommodations *and Ag-2's ontology represents that* o2:accommodation *is common (as in Figure 2). Ag-2 sends the message again with concept* o2:accommodation. *Ag-1 interprets the message as the most specific implied native concept* o1:accommodations, *and responds "OK".*

2. *Consider the ontologies in Figure 2. Ag-2 intends to convey* o2:holiday-inn. *Ag-2 translates to* o1:hotel *and sends an Inform-message. Ag-1 interprets the message as* o1:hotel. *Ag-1 evaluates the message as "OK", because its ontology cannot contain more specific information, i.e. there are no native particularizations of* o1:hotel.

3. *Consider the ontologies in Figure 2. Ag-2 intends to convey* o2:bus, *which Ag-2 translates to* so1:transport. *Ag-2 sends an Inform-message. Ag-1 evaluates the message as OK, because all native particularizations of* so1:transport *are either common, or a subconcept of a common concept. Therefore, it knows that, if more specific information could have been conveyed, Ag-1 would have used a more specific concept.*

4. *Consider the ontologies in Figure 2. Ag-2 intends to convey* so2:location. *Ag-2 translates to the most specific common implied concept* so1:something *and sends an Inform-message. Ag-1 replies that "Concept overgeneralized". Ag-2 translates* so2:location *to the non-common concept* so2:location *and sends another message. Ag-1 understands the message, and responds "OK". After this conversation, both agents represent the concept* so2:location *as common concepts.*

This example shows how the dialogue mechanism fulfills the requirements introduced in this paper. The agents exploit the potential use of common concepts (Requirement 1.1) by expressing themselves in more general common concepts without losing information in the communication process. As appears from Example 1.2 and 1.3, this prevents the unnecessary explication of concepts. The use of unknowingly shared concepts (Requirement 1.2) is demonstrated in Example 1.4. The agents effectively apply acquired concepts in agent communication, as appears from Example 1.2 (Requirement 2). Finally, the agents explicate concepts when needed and only when needed (Requirement 3). As demonstrated in Example 1.1, the agents adequately recognize information loss which leads to the necessary application of concept explication. The agents also recognize when concept explication is *not* needed. As demonstrated in Example 1.2, Ag-1 recognizes that no information is lost, as its ontology cannot represent more specific information. Example 1.3 shows the use of conversational implicatures to recognize that as much information as possible has been conveyed.

5.1 Formal background

This paper demonstrates how various techniques for ontology reconciliation can be combined to develop semantically interoperable E-commerce systems. The informal style of presentation serves this purpose well. Nevertheless, many applications require a more rigorous treatment in order to formally verify the communication mechanism. We have presented such a formal analysis in [34]. Using formal communication protocols and description logic ontologies [3], we provided solid proofs that the communication mechanisms possess the desirable properties. A brief discussion on these desirable properties is given below.

To analyze the issues relevant to ontology reconciliation, it must be possible to compare two concepts from two different ontologies. Whereas formal ontology languages such as description logics [3] provide accurate means to specify the relations between concepts in the same ontology, they leave the relation between concepts in different ontologies undefined. We have addressed this problem by introducing the notion of a *god's eye view ontology*, i.e. the ontology that would arise if the Venn-diagram representations of each agent's ontology would be placed on top of each other.

Within the god's eye view ontology, the notions of sound and lossless communication can be formally defined. Sound communication means that the concept as interpreted by the hearer is a superconcept of the concept the speaker intends to convey. Lossless communication means that the concept as interpreted by the hearer is the most specific superconcept of the concept the speaker intends to convey. We have proven that the mechanisms for sending and interpreting messages as specified in 1 and 2 guarantee sound communication. We have proven that the evaluation mechanism of Specification 3 guarantees lossless communication.

When the agents start in a situation which is most difficult, they do not share any concepts with each other. In order to reduce the required number of concept explications to a minimum, the agents should bring about as few common concepts as possible. The smallest amount of common concepts which can be used for lossless communication of every concept is called an optimal communication vocabulary [32]. We have proven that a slightly extended version of the communication mechanism presented here is guaranteed to establish an optimal communication vocabulary [34].

6. CONCLUSION

In this paper, we have discussed various approaches proposed in the literature for solving semantic interoperability problems. For E-commerce systems, every approach is useful in some sense. Therefore, we have incorporated all techniques in one agent communication protocol, to obtain the best of each world. The resulting communication protocol demonstrates that ontologies can be effectively reconciled without requiring every agent to know every other agent's ontology. Communication proceeds using general common concepts whenever this does not give rise to unnecessary information loss. When too much information is lost in the communication process, the agents extend their ontologies in order to solve the problem.

7. REFERENCES

[1] http://e-tourism.deri.at/documents/harmoniseontology.

[2] http://protege.cim3.net/file/pub/ontologies/shuttle/shuttle-crew-ont.owl, 2002.

[3] F. Baader, D. Calvanese, D. McGuiness, D. Nardi, and P. Patel-Schneider, editors. *The description logic handbook: Theory, implementation and applications*. Cambridge University Press, 2003.

[4] S. Bailin and W. Truszkowski. Ontology negotiation between intelligent information agents. *Knowledge Engineering Review*, 17(1):7–19, 2002.

[5] A. Borgida and L. Serafini. Distributed description logics: Assimilating information from peer sources. In S. Spaccapietra, editor, *Journal of Data Semantics*, volume 1 of *LNCS*, pages 153–184. Springer Verlag, 2003.

[6] B. Chandrasekaran, J. R. Josephson, and V. R. Benjamins. What are ontologies, and why do we need them? *IEEE Intelligent Systems*, 14(1):20–26, 1999.

[7] M. Ciocoiu, M. Gruniger, and D. Nau. Ontologies for integrating engineering applications. *Journal of Computing and Information Science in Engineering*, 1(1):12–22, 2001.

[8] A. Farquhar, R. Fikes, and J. Rice. The ontolingua server: a tool for collaborative ontology construction. *Int. J. Hum.-Comput. Stud.*, 46(6):707–727, 1997.

[9] C. Fellbaum. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

[10] FIPA. FIPA Ontology Service Specification, 2000.

[11] P. Grenon, B. Smith, and L. Goldberg. Biodynamic ontology: Applying BFO in the biomedical domain. In D. M. Pisanelli, editor, *Ontologies in Medicine*, pages 20–38, Amsterdam, 2004. IOS Press.

[12] H. Grice. Logic and conversation. In P. Cole and J. Morgan, editors, *Speech Acts*, pages 41–58, New York, 1975. Academic Press.

[13] T. R. Gruber. Towards Principles for the Design of Ontologies Used for Knowledge Sharing. In N. Guarino and R. Poli, editors, *Formal Ontology in Conceptual Analysis and Knowledge Representation*, Deventer, The Netherlands, 1993. Kluwer Academic Publishers.

[14] KSE. Knowledge sharing effort public library, 1994.

[15] D. B. Lenat and R. V. Guha. *Building Large Knowledge-Based Systems: Representation and Inference in the Cyc Project*. Addison-Wesley Publishers, 1990.

[16] M. Luck, P. McBurney, and C. Preist. Agent technology: Enabling next generation computing. *Agent link community*, pages 74–75, 2003.

[17] C. Matuszek, J. Cabral, M. Witbrock, and J. DeOliveira. An introduction to the syntax and content of Cyc. In *the 2006 AAAI Spring Symposium on Formalizing and Compiling Background Knowledge and Its Applications to Knowledge Representation and Question Answering*, 2006.

[18] D. Maynard, G. Stamou, I. Zaihrayeu, J. Barrasa, J. Euzenat, M. Hauswirth, M. Ehrig, P. Bouquet, R. Dieng-Kuntz, R. L. Hernández, S. Tessaris, S. V. Acker, and T.-L. Bach. D2.2.3: State of the art on current alignment techniques. Knowledge Web, 2004.

[19] D. L. McGuinness, R. Fikes, J. Rice, and S. Wilder. The chimaera ontology environment. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI 2000)*, Austin, Texas, 2000.

[20] E. Mena, A. Illarramendi, V. Kashyap, and A. Sheth. OBSERVER: An approach for query processing in global information systems based on interoperation across pre-existing ontologies. *International journal on Distributed And Parallel Databases (DAPD)*, 8(2):223–272, April 2000.

[21] M. M. Montez, J. L. Bas, S. Bellido, O. Corcho, S. Losada, R. Benjamins, and J. Contreras. WP10: Case study eBanking D10.3 Financial Ontology. April 2005.

[22] R. Neches, R. Fikes, T. Finin, T. Gruber, R. Patil, T. Senator, and W. R. Swartout. Enabling technology for knowledge sharing. *AI Magazine*, 12(3):36–56, 1991.

[23] I. Niles and A. Pease. Towards a standard upper ontology. In *FOIS '01: Proceedings of the international conference on Formal Ontology in Information Systems*, pages 2–9, New York, NY, USA, 2001. ACM Press.

[24] N. F. Noy and M. Klein. Ontology evolution: Not the same as schema evolution. *Knowledge Information Systems*, 6(4):428–440, 2004.

[25] N. F. Noy and M. A. Musen. Prompt: Algorithm and tool for automated ontology merging and alignment. *In Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2000.

[26] A. D. Preece, K. ying Hui, W. A. Gray, P. Marti, T. J. M. Bench-Capon, D. M. Jones, and Z. Cui. The KRAFT architecture for knowledge fusion and transformation. *Knowledge Based Systems*, 13(2-3):113–120, 2000.

[27] G. Schreiber, H. Akkermans, A. Anjewierden, R. de Hoog, N. Shadbolt, W. V. de Velde, and B. Wielinga. *Knowledge Engineering and Management, The CommonKADS Methodology*. MIT Press, 2000.

[28] J. F. Sowa. *Knowledge Representation: Logical, Philosophical and Computational Foundations*. Brooks/Cole Publishing Co., Pacific Grove, CA, USA, 2000.

[29] H. Stuckenschmidt and I. J. Timm. Adapting communication vocabularies using shared ontologies. *Proceedings of the Second International Workshop on Ontologies in Agent Systems (OAS)*, July 2002.

[30] R. Studer, V. R. Benjamins, and D. Fensel. Knowledge engineering: Principles and methods. *Data Knowledge Engineering*, 25(1-2):161–197, 1998.

[31] G. Stumme and A. Maedche. FCA-MERGE: Bottom-up merging of ontologies. In *IJCAI*, pages 225–234, 2001.

[32] J. van Diggelen, R. Beun, F. Dignum, R. van Eijk, and J.-J. Meyer. Optimal communication vocabularies and heterogeneous ontologies. In *Agent Communication, International Workshop on Agent Communication, AC 2004*, volume 3396 of *LNAI*, pages 76–90. Springer Verlag, 2004.

[33] J. van Diggelen, R. Beun, F. Dignum, R. van Eijk, and J.-J. Meyer. ANEMONE: An effective minimal ontology negotiation environment. In P. Stone and G. Weiss, editors, *Proceedings of the Fifth International Conference on Autonomous Agents and Multi-agent Systems (AAMAS06)*, pages 899–906. ACM, 2006.

[34] J. van Diggelen, R. Beun, F. Dignum, R. van Eijk, and J.-J. Meyer. Ontology negotiation: Goals, requirements and implementation. *International Journal of Agent-Oriented Software Engineering (IJAOSE)*, 1(1):63–90, 2007. Inderscience publishers.

[35] G. Wiederhold and M. Genesereth. The conceptual basis for mediation services. *IEEE Expert: Intelligent Systems and Their Applications*, 12(5):38–47, 1997.