OWL DL as a FIPA ACL content language

Bernhard Schiemann, Ulf Schreiber
Artificial Intelligence Division, Department of Computer Science,
University of Erlangen–Nuremberg
{schiemann|ulf.schreiber}@informatik.uni-erlangen.de

Abstract Agent-Agent-Communication using FIPA ACL speech acts is a well-known implementation of ontology based communication. The content of an ACL Message is formulated in FIPA SL content language or one of its sublanguages (SL1, SL2). Yet, validating the semantics of incoming SL encoded messages is a difficult task because SL is undecidable. Therefore, in order to enable a semantic validation in such cases, we use W3C's OWL DL for the ontological encoding of the content field and the tableaux reasoner RACER for validation. Additionally, we explain why this approach enables a more flexible use of ontologies in machine interpretable communication. We draw up a straight forward solution based on the combination of OWL DL and FIPA ACL Messages.

1 Introduction

In recent years, software agents (abbr. agents) have become a well studied and frequently applied implementation technique for distributed systems. From a users's perspective, agents enable innovative capabilities like proactivity, reactivity, social ability and autonomy. For research work in AI, such agents buid systems that are multi agent systems (MAS). In this paper, we focus on agent-agent-communication with respect to its social ability in terms of, for example, cooperativity, and autonomy. For communication, formal ontologies are used to provide concepts required to interpret the "content words" of the messages.

There is more than one definition of the term "ontology". We follow the classification of [1], in which a formal ontology is defined as terminological component of a Knowledge Base (KB). In addition, we use the definition of a knowledge based agent given by [2], according to which an agent uses its knowledge base containing theorems to reason about the application domain. Combining these two approaches, each agent of an MAS holds a KB based on the domain ontology (application ontology).

For agents satisfying the previous constraints, the domain ontology can be formulated in OWL DL [3]. The syntactic representation of an OWL DL ontology in RDF-XML can be translated into the description logic $\mathcal{SHOIN}(\mathcal{D})$. The KB of an agent consists of two parts: a *T-Box*, which holds the domain specification, and an *A-Box* where assertions are stored. Existing reasoners for OWL DL like RACER [4] provide *A-Box-T-Box*-reasoning which can among other possibilities be accessed by the http-DIG interface.

In order to model an ontology-based communication the following four FIPA[1] specifications were defined:

1. The FIPA message structure specification [5].

[1] Foundation for Intelligent Physical Agents

2. The FIPA Ontology Service specification [6], which models ontology based communication in general.

3. The FIPA SL Content Language Specification [7], that proposes a language for the content field of a message.

4. The FIPA Communicative Act Library Specification [8] and its substandards, which define the speech act details (i.e. performatives).

The fourth specification defines speech acts with the help of a formal model and examples. Consequently it provides agent communication framework in terms of schemata for subdialogs. One example for such a scheme is the *query-ref–inform* speech act pair. Unfortunately, these speech acts already affect the level of expressivity that has to be used in the content field, e.g.

a) the "request" speech act in the fourth FIPA specification determines that the content field should contain an action, whereas

b) the "request-when" speech act requires another speech act in its content field.

Using the semantics of such speech acts in combination with FIPA's second and third specifications determines the minimum expressivity of the content field, which is first order logic with modal operators.

A closer look at the FIPA SL content language (CL) shows that it is in general undecidable. However, the SL profiles *SL1*, *SL2*[2], have to be decidable by definition. Using these SLs as content languages has two major drawbacks:

First, the way in which SL is applied determines whether one of the profiles is satisfied or not. Only when one of the profiles is satisfied, the content language is decidable. Therefore it is the application designer's responsibility to model the content in a way that satisfies one of the profiles in order to make it decidable. Hence, building reasoners for message contents is a very complex task, taking into account that in the ideal case messages types are modelled by an application designer who is usually not an expert in reasoning systems.

Second, as already mentioned, the speech act "request-when" must have another speech act as its content. Therefore a separation of speech act semantics and content semantics of the speech acts in SL is difficult. Moreover, there is no tool or methodology that helps MAS developers building ACL protocols without knowing the content semantics of the speech acts.

There are more than one FIPA content language proposals, listed at FIPA's website[3]. We concentrate on SL because:

- The status for other FIPA content languages is experimental.
- The RDF CL specification does not provide a formal (logic) semantic model.
- The Constraint Choice Language (CCL) focuses on the communication of choice problems. Using this CL forces application programmers to use the FIPA Constraint Choice Language Ontology, which restricts flexibility of the content.
- The KIF CL specification again introduces predicate logic to the content field with the already discussed expressivity drawbacks. Even the KIF based Ontolingua has a problem with its expressive power [9], such that no general reasoner support for KIF exists.

[2] SL Profiles *SL1* and *SL2* are concrete SL based content languages

[3] http://www.fipa.org/repository/cls.php3

Our approach here is to use contents that are exclusively formulated in OWL DL for FIPA ACL compliant messages. Using OWL DL as a content language for FIPA ACL should facilitate the building of simple interaction protocols that are based on information dialogues. The separation of speech act semantics from content semantics can be easily achieved by using of OWL DL as a content language. Such separation allows to avoid the problems with SL mentioned above. In our approach we clean the contents of a OWL DL FIPA message from speech act semantics defined by the FIPA Communicative Act Library Specification. Therefore the speech act semantics only defines the (sub)dialog semantics for example that a *inform* speech act follows a *query-ref* speech act.

2 Content Language

Figure 1: Agent A informs agent B (both KB agents) about an assertion E.

A common domain-related agent task within a MAS is to query and inform each other about domain knowledge.

Figure 1 shows an *inform* speech act in which agent A provides agent B with information about an assertion E from agent A's *A-Box*. The content field of this message contains a (positive) proposition encoded in OWL DL, which satisfies expressivity, especially that positive propositions can be easily stated as assertions in OWL DL.

For a *query-ref* speech act, the content field can only be filled with referential expressions. If, for example, agent B wants to know what agent A knows about e it sends a *query-ref* ACL message. The referential expression in the content field is formulated as OWL DL class expression which describes individuals with certain properties[4]. This fixes the granularity of query expressions on the individual level, which means that queries are rather high level, asking for everything known about the individuals matching the specified pattern, instead of asking for a specific kind of low level assertion.

The main advantage of referencing using class descriptions is that they directly plug into the classification capability of a reasoner without needing a dedicated query language. An-

[4]similar to "query-by-example"

other benefit is that this mechanism allows for a great variety of *A-Box* queries, since complex class descriptions can be composed from named concepts, restrictions (which are itself class descriptions in OWL DL), enumerations and the boolean set operators.

We are aware that a dedicated query language would still be desirable because the mechanism described above does not work for *T-Box* queries[5]. Nevertheless, a standard query language for OWL DL knowledge bases is a current research topic (cf. OWL QL [10]).

3 Application ontologies in OWL DL

In this section we discuss the advantages of combining the content field with application ontologies formulated in OWL DL.

Application ontologies can ideally be formulated irrespective of any application architecture (or MAS), the speech act ontology for MASs is applicable to multiple different application domains. This domain-independence aspect of agent-agent communication enables a clear separation between the speech act ontology and the application ontology, which in turn makes the MAS applicable to different domains.

OWL is a specification grew out of the design of languages for semantic web. Many domain experts already formulate their ontologies in OWL DL. Consequently, various application (domain) ontologies[6] are available so that the choice of OWL DL shows a great potential for reusing already formulated ontologies.

Agent-agent communication of independently developed agents faces problems with the respect to the integration of ontologies of different domains or even of different ontologies within the same domain [11]. Similar import problems exist in the development process of the semantic web. Collaboration between agent and semantic web researchers may help in solving these integration problems. Additionally, on the the side of the semantic web developers, the W3C suggests the use of OWL within agents (cf. [12]). Our approach provides a clear mechanism for combining agents with OWL DL. At the same time, from the agent developer perspective the semantic web is a well suited environment for agents, especially if their KBs are formulated with OWL DL.

4 Implementation Details

The JADE agent framework[7] is a well known open source middleware for MAS [13] that implements the FIPA specifications in Java. Current ontology support in JADE performs only limited type verification of incoming SL encoded content. Assertions from this content are interpreted as instances of prebuilt Java Beans. These Java Bean classes are generated e.g. from OWL by the Beangenerator[8] plugin for the Protégé[9] ontology editor. Using Java Beans as knowledge representation language seems to be unsatisfactory if changes in the *T-Box* are to occur at runtime and it also does not even support assigning multiple inheritance to one individual.

In order to integrate OWL DL as content language we implement a content codec for JADE that replaces the JADE ontology support described above. This codec is supplemented by a knowledge base infrastructure for managing an internal agent knowledge base in OWL DL. Validation of message content and inference on the knowledge base are achieved by connecting to a RACER server over the DIG interface. The *A-Box* representation of this

[5] this requires the use of classes of classes which is only available in OWL Full

[6] See http://protege.stanford.edu/plugins/owl/owl-library/

[7] http://jade.tilab.com/

[8] http://acklin.nl/beangenerator

[9] http://protege.stanford.edu/

knowledge base consists of instances of a domain independent Java class. Each of these instances holds the concept membership and property value assertions for one individual.

In addition to the generic interface provided by this class there is also a tool to generate optional wrapper classes, which provide a more convenient interface tailored to specific *T-Box* concepts. From the programmer's point of view these accessor classes[10] ease the transition from FIPA SL to OWL DL, since they resemble the Java Beans from the established JADE ontology support.

Currently the knowledge base implemention contains methods for the validation of incoming messages, automatic processing of query messages and coherencec̄hecked absorption of *inform* content into the agent's *A-Box*. This means that while the knowledge base is protected from contradictions introduced by incoming messages, there is still no domain-independent update mechanism that could relieve agent programmers of the duty to implement domain specific methods in order to accomodate for change.

All in all our use of the JADE toolkit allows us to rely on its well-elaborated implementation of FIPA ACL message envelope handling and to focus exclusively on the changes resulting from converting to OWL DL as a content language. At the same time this gives agent programmers the opportunity to leverage existing experience with JADE for writing OWL DL enabled agents. Finally, the new codec allows for changes of the *T-Box* at runtime.

5 Application Example

In this section we show an example of an information dialog between agents using *query-ref* and *inform* performative speech acts. The application ontology used here is the pizza ontology[11] that comes as an example with Protégé. Both agents initialize their *T-Box* with this ontology.

A ChefAgent (CA) stores representations of the pizzas that are ready for pickup in his A-Box and a WaiterAgent (WA) queries available pizzas of certain types. When the WaiterAgent wants to know if there are any pizzas from Germany ready for delivery then it sends the ChefAgent a message with the *query-ref* performative:

```
(QUERY-REF
 :sender  ( agent-identifier
  :name WaiterAgent@example.com
  :addresses (sequence http://example.com:7780/jade ) )
 :receiver  (set ( agent-identifier
   :name ChefAgent@example.com
   :addresses (sequence http://example.com:7781/jade ) ) )
 :content  "<rdf:RDF
   xml:base=\"http://example.com/WaiterAgent/msg1234\"
   xmlns:rdf=\"http://www.w3.org/1999/02/22-rdf-syntax-ns#\"
   xmlns:owl=\"http://www.w3.org/2002/07/owl#\">
  <owl:ontology rdf:about=\"http://example.com/WaiterAgent/msg1234\"
   owl:imports=\"http://example.com/WaiterAgent/kb\"/>
  <owl:Class rdf:about=\"#ResultSet\">
  <owl:equivalentClass>
   <owl:Class>
    <owl:intersectionOf rdf:parseType=\"Collection\">
     <owl:Class
      rdf:about=\"http://www.co-ode.org/ontologies/
                          pizza/pizza_20041007.owl#Pizza\"/>
     <owl:Restriction>
      <owl:onProperty
```

[10] comparable to those from Kazuki (http://projects.semwebcentral.org/projects/kazuki/)

[11] http://www.co-ode.org/ontologies/pizza/pizza_20041007.owl

```
       rdf:resource=\"http://www.co-ode.org/ontologies/pizza/
                              pizza_20041007.owl#hasCountryOfOrigin\"/>
      <owl:hasValue
       rdf:resource=\"http://www.co-ode.org/ontologies/
                              pizza/pizza_20041007.owl#Germany\"/>
     </owl:Restriction>
    </owl:intersectionOf>
   </owl:Class>
  </owl:equivalentClass>
 </owl:Class>
</rdf:RDF>"
 :language  OWL-DL
 :ontology  http://example.com/WaiterAgent/kb
 :conversation-id  WaiterAgent@example.com1234567890 )
```

Note that this message content is a valid, self contained OWL document, formally importing the knowledge base of the sending agent, which in turn contains the concept definitions of the pizza ontology used. The sender (WA), by sending such a message, assumes that the receiver (CA) will understand the terms used in the content field. In this example (and for many other MAS) one can assume that a shared application ontology is available to implement a common grounding between all agents of a MAS. Without this common grounding assumption an ontology-merge-align-process would have to clarify if a belief-revision (update) process at the receiver agent (CA) should incorporate concepts of the sender agent (WA). Note that such a process is already prepared by the import statement.

Assuming that the CA has one German pizza ready, which happens to be a MeatyPizza named `#bratwurstpizza3456` (defined by the CA, therefore using the URI of its knowledge base as a base URI) and topped with Bratwurst (fried sausage), it sends an *inform* message like this one:

```
(INFORM
 :sender  ( agent-identifier
  :name ChefAgent@example.com
  :addresses (sequence http://example.com:7781/jade ) )
 :receiver  (set ( agent-identifier
   :name WaiterAgent@example.com
   :addresses (sequence http://example.com:7780/jade ) ) )
 :content  "<rdf:RDF
  xml:base=\"http://example.com/ChefAgent/msg5678\"
  xmlns:rdf=\"http://www.w3.org/1999/02/22-rdf-syntax-ns#\"
  xmlns:owl=\"http://www.w3.org/2002/07/owl#\"
  xmlns:wcoopp=\"http://www.co-ode.org/ontologies/pizza/
                 pizza_20041007.owl#\">
 <owl:ontology rdf:about=\"http://example.com/ChefAgent/msg5678\">
  <owl:imports rdf:resource=\"http://example.com/ChefAgent/kb\"/>
  <owl:imports rdf:resource=\"http://example.com/WaiterAgent/msg1234\"/>
 </owl:ontology>
 <wcoopp:MeatyPizza
  rdf:about=\"http://example.com/ChefAgent/kb#bratwurstpizza3456\">
  <rdf:type
   rdf:resource=\"http://example.com/WaiterAgent/msg1234#ResultSet\"/>
  <wcoopp:hasTopping>
   <wcoopp:MeatTopping
    rdf:about=\"http://example.com/ChefAgent/kb#bratwursttopping4567\"/>
  </wcoopp:hasTopping>
  <wcoopp:hasBase
   rdf:resource=\"http://example.com/ChefAgent/kb#base5678\"/>
  <wcoopp:hasCountryOfOrigin
   rdf:resource=\"http://www.co-ode.org/
```

```
                          ontologies/pizza/pizza_20041007.owl#Germany\"/>
 </wcoopp:MeatyPizza>
</rdf:RDF>"
 :language  OWL-DL
 :ontology  http://example.com/ChefAgent/kb
 :conversation-id  WaiterAgent@example.com1234567890 )
```

This inform message is a concrete example for the *inform* speech act shown in Figure 1. An interesting detail in this message is that the individual `#bratwurstpizza3456` is explicitly asserted to be a member of `http://example.com/WaiterAgent/-msg1234#ResultSet`. Without that the receiver would only know that the message contains assertions stating the existence of four individuals, namely `#bratwurstpizza3456`, `#bratwursttopping4567`, `#base5678` and `#Germany`, with a varying level of detail regarding property fillers and types of the individuals.

Instantiating the `#ResultSet` class in the content field of the *inform* message provides an explicit reference to the referential expression in the *query-ref* message, which allows to distinguish between those individuals which are members of the class description in the query and those which are just mentioned in the message because they are somehow related to a member by means of an `owl:ObjectProperty`. In this example, this distinction could also be accomplished with inference done by the reasoner, but this approach would generally force the sending agent to include all information necessary for this inference, which, depending on the class description used as referential expression, might be very difficult to determine.

The conversation-id in the FIPA ACL envelope surrounding the OWL document also provides contextual information that ties it to the referential expression of the *query-ref* message, but this applies to the whole message content and therefore does not offer the necessary granularity.

Lastly, making this distinction based on the syntactic structure of the message would be a violation of OWL (and RDF) rules, since the specific way in which a given set of triples is serialized is not supposed to carry any meaning.

So far, for the ease of understanding, the example given above does not address the issue of doing meaningful agent communication without using domain related actions. In order to achieve this, the domain ontology can often be extended so that, for example, the whole order/make/deliver/pay life-cycle of a pizza can be handled with nothing but the two performatives *query-ref* and *inform*.

One possible way to do this is to add the boolean properties `isOrdered`, `isMade`, `isDelivered`, `isPaid` and `isImpossibleToMake` to the `Pizza` concept (or to a concept `OrderableThing` that `Pizza` inherits from) that act like a checklist for the various states. Now, when a patron orders a pizza, the WaiterAgent would instantiate the appropriate subclass of `Pizza` with the `isOrdered` state set to true. The other state properties would have to be explicitly initialized to false in order to get clearly identifiable states under the open world assumption. Later it would send an *inform* message containing this individual to the ChefAgent. If the WaiterAgent is expected to poll the ChefAgent for meals that are ready for delivery (maybe because there are multiple waiters and broadcasting to all of them is not allowed) then the ChefAgent can not proactively inform the WaiterAgent when the pizza is ready, instead it will silently set the `isMade` property of that pizza individual to true in his knowledge base. Now whenever the WaiterAgent is ready to take pizza from the kitchen to the guest room it will send a *query-ref* message like the one above (but with additional restrictions about the states of `isMade` and `isDelivered`) to the ChefAgent in order to learn which pizzas are ready for delivery. It goes without question that a good WaiterAgent would be expected to proactively query the state of any pizza, that has not been delivered

a long time after being ordered, before the customer complains (maybe the ChefAgent has set `isImpossibleToMake` to true because anchovies are out). Single individuals can be referenced inside a *query-ref* message by using `owl:oneOf` together with the URI of the individual as the class description.

6 Discussion and Further Work

As mentioned in section 2 there is still no agreement on a standard query language (QL) for OWL DL KBs, and hence there are several QLs. The TAGA project [14] chose to support multiple QLs. This results in two drawbacks: First, the overhead and the complexity of the QL support in JADE will increase. Second, the foundation of a standard QL would make the support of multiple QLs unnecessary.

A possible alternative approach to the one presented in this paper, is to define a subset of FIPA SL that is reduced from FOL plus modal operators to $\mathcal{SHOIN}(\mathcal{D})$. However, the resulting language would be limited to the same expressiveness as OWL DL. At the same time it would complicate the reusability of existing application ontologies formulated in OWL DL by establishing the need for an additional syntactic filter between OWL DL and this subset of FIPA SL.

Our approach is to use content that is exclusively formulated in OWL DL for FIPA ACL compliant messages. That includes the separation of content semantics from speech act semantics. One limitation of this approach is that we only use propositions or referential expressions in the content field of speech acts. Since $\mathcal{SHOIN}(\mathcal{D})$ does not contain modal operators, some FIPA ACL speech acts cannot be expressed. Whether this is a problem or not depends on the structure of the MAS and the task for which it is built. A part of our ongoing work deals with adding OWL DL compatible semantics to further FIPA ACL speech acts.

However, our experience with MAS implementations shows that making the content decidable would in many cases compensate the limitation in expressiveness. In addition, having a decidable domain knowledge encoding is of particular importance for agents that are managing their own KBs by theory revision. The main topic of our future research work will be theory revision among agents communicating in FIPA ACL with OWL DL as content language for knowledge representation.

References

[1] Nicola Guarino, Laure Vieu, and Stefano Borgo. Formal Ontology for Semanticists, a course at ESSLLI, 2005.

[2] S. Russel and P. Norvig. *Artificial Intelligence*, chapter 7. logic based agents. Prentice Hall, 2003.

[3] Eric Miller et al. Web Ontology Language (OWL), 2004.

[4] RACER Systems GmbH. The features of racerpro version 1.9, 2005.

[5] Foundation for Intelligent Physical Agents FIPA. FIPA ACL Message Structure Specification, 2002.

[6] Foundation for Intelligent Physical Agents FIPA. FIPA Ontology Service Specification, 2001.

[7] Foundation for Intelligent Physical Agents FIPA. FIPA SL Content Language Specification, 2002.

[8] Foundation for Intelligent Physical Agents FIPA. FIPA Communicative Act Library Specification, 2002.

[9] Dieter Fensel, Ian Horrocks, Frank van Harmelen, Stefan Decker, Michael Erdmann, and Michel C. A. Klein. OIL in a nutshell. In *Knowledge Acquisition, Modeling and Management*, pages 1–16, 2000.

[10] Richard Fikes, Pat Hayes, and Ian Horrocks. OWL-QL: A Language for Deductive Query Answering on the Semantic Web. Technical Report KSL 03-14, Stanford University, Stanford, CA, 2003.

[11] Helena Sofia Pinto and João P Martins. A methodology for ontology integration. In *K-CAP '01: Proceedings of the 1st international conference on Knowledge capture*, pages 131–138, New York, NY, USA, 2001. ACM Press.

[12] Web Ontology Working Group. OWL Web Ontology Language: Use Cases and Requirements, 2004.

[13] Giovanni Caire. JADE Introduction AAMAS 2005, 2005.

[14] Youyong Zou, Tim Finin, Li Ding, Harry Chen, and Rong Pan. Using Semantic web technology in Multi-Agent systems: a case study in the TAGA Trading agent environment. In *Proceeding of the 5th International Conference on Electronic Commerce*, September 2003.