Representing Interaction Protocols in DAML

Santtu Toivonen
VTT Information Technology
P.O.Box 1203, FIN-02044 VTT
Finland
Santtu.Toivonen@vtt.fi

Heikki Helin
Sonera Corporation
P.O.Box 970, FIN-00051 Sonera
Finland
Heikki.j.Helin@sonera.com

Abstract

We present an extension to DAML-S for representing interaction protocols. An interaction protocol defines the messaging patterns between communicating entities such as software agents. Serializing interaction protocols in a suitable form for reuse supports creating software agents capable of adapting to various environments. Serialized interaction protocols can be utilized, for example, when specifying details of interaction between a contractor and a subcontractor operating in the Internet.

Introduction

The importance of web service automatization is going to increase in the future Internet. Initiatives such as Web Services (World Wide Web Consortium 2002) and the Semantic Web (World Wide Web Consortium 2001) aim at shifting the burden of doing certain mechanical and monotonic things from humans to computers. This paper describes a specific interaction protocol based framework contributing to reaching that goal. We believe interaction protocols (IPs) function well as process descriptions for software agents. That is because IPs consist of speech acts or communicative acts (CAs) in certain order. CAs, in turn, are often utilized when modeled software agent communication.

We use interaction between a contractor and a subcontractor operating in the Internet as an example displaying a more general approach to software agent programming. The underlying idea of that approach is that software agents' information and knowledge should be externalized when possible. That idea is based on distributing cognition (see e.g., (Hutchins 1996)). In this paper we put forward our theories presented in earlier work (Toivonen & Helin 2002b; 2002a). We try to create agents that are relatively simple as such, but capable of externalizing their knowledge into their surroundings and also internalizing knowledge from their surroundings.

Traditionally the theory of distributed cognition is applied to human agents, but the approach can be extended to software agents as well. Agents have knowledge of facts and tasks. We have concentrated mainly on task-related knowledge or "know-how". More specifically, in (Toivonen & Helin 2002b) we divided the tasks of agents into general tasks and subtasks. We proposed that descriptions of domain-specific subtasks could be distributed in external repositories and the agents could download those descriptions and modify their behavior accordingly.

In (Toivonen & Helin 2002a), we focused our approach to agent conversations. Conversations are natural candidates for domain-specific subtasks of agents. We made a categorization of agent conversation elements into type- and instance-level elements as well as primitive and composite elements, as depicted in Table 1. Composite elements differ from primitive elements in that unlike composite elements, primitive elements cover only one state transition. The difference between instance- and type-level elements corresponds to the difference between objects and classes.

In this paper we further focus our work to serializing interaction protocols (see Table 1). We start by describing our proposal for an interaction protocol ontology that extends concepts defined in DAML-S (Ankolekar & others 2002; Martin & others 2002). Then, we apply our interaction protocol ontology to web service automatization scenario between a contractor and a subcontractor. We then present some related work. Finally, we draw conclusions and discuss ongoing and future work.

Table 1: Division into type- and instance-level elements

	Primitive	Composite
Type-level	Communicative Act	Interaction Protocol
Instance-level	Message	Conversation

Interaction Protocol Ontology

For defining the common aspects of various IP instances, we have defined an interaction protocol ontology that specializes concepts of DAML-S. More specifically, the concepts utilized here are from DAML-S version 0.7 (Martin & others 2002). The interaction protocol ontology includes definitions for the following components (see Figure 1):

Interaction protocol `InteractionProtocol` is the topmost concept of the interaction protocol ontology specification.

Progress An IP has one `Progress` component. `Progress` defines the flow of the IP at the upper-

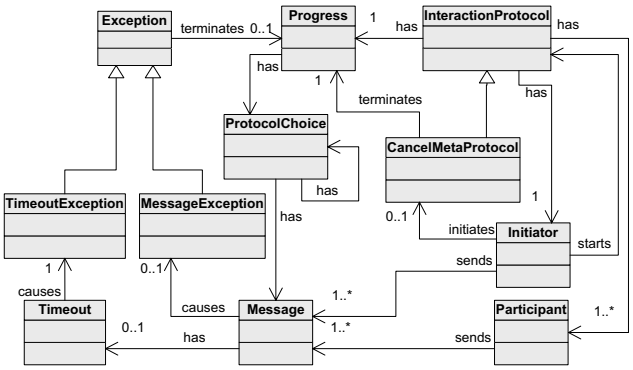

Figure 1: Components of interaction protocol ontology

most level. `Progress` is a subclass of DAML-S `Sequence`.

Initiator An IP has one `Initiator` agent that starts the IP by sending the first `Message` of the IP.

Participant An IP has at least one `Participant` agent that sends at least one `Message` during the interaction.

Protocol choice `Progress` is constituted of one or more `ProtocolChoices`. Options of a `ProtocolChoice` are either further `ProtocolChoices` or `Messages`. A `ProtocolChoice` is a subclass of DAML-S `Choice`.

Message A `Message` is sent by one agent to another(s). It is a subclass of DAML-S `AtomicProcess`.

Timeout A `Message` can have a `Timeout`. `Timeout` is a subclass of DAML-S `Condition`.

Exception An `Exception` can terminate the `Progress` of the IP at any time. `Exception` is a subclass of DAML-S `ConditionalEffect`.

Timeout exception A `TimeoutException` occurs when a `Message` is not delivered by time. `TimeoutException` is a subclass of `Exception`.

Message exception A `MessageException` is triggered when an agent receives message that does not belong to normal flow of the IP. An example of such is a NOT-UNDERSTOOD `Message` (Foundation for Intelligent Physical Agents 2002a). `MessageException` is a subclass of `Exception`.

Cancel metaprotocol `CancelMetaProtocol` is initiated by the `Initiator`. `CancelMetaProtocol` terminates the `Progress` of the IP.

Serializations of individual IP instances, such as those defined by FIPA (Foundation for Intelligent Physical Agents 2001), are intended to specialize the concepts of the interaction protocol ontology specification. `Progress`-component of a FIPA Request (Foundation for Intelligent Physical Agents 2002b), for example, would consist of a REQUEST `Message` and two consecutive `ProtocolChoices`. First `ProtocolChoice` would consist of two `Messages`: REFUSE and AGREE. The second one would have three `Messages`: FAILURE, INFORM-DONE, and INFORM-RESULT. This second `ProtocolChoice` would be initiated if neither `Exceptions` nor `CancelMetaProtocol` initiations had emerged while processing the first `ProtocolChoice`.

Interaction Protocol Based B2B Interaction

Figure 2 depicts the essential steps of interaction between a contractor and a subcontractor. The subcontractor has published its service profiles and process descriptions in a generally accepted format somewhere available to the contractor(s). A contractor wishing to utilize services provided by the subcontractor has received a pointer to the subcontractor somewhere, for example, from a UDDI (UDDI 2000) registry (see step 0 in Figure 2).

The contractor first contacts the subcontractor and the subcontractor provides the contractor with a pointer to the IP repository (1). IPs found in the IP repository instantiate the interaction protocol ontology specification outlined in the previous section. The contractor can download the profiles and process descriptions (2). These descriptions contain also information whether the contractor should be the initiator or a participant of the IP. Next, the contractor's software agents may have to adapt to the descriptions in order for the interaction to take place (3). The contractor's agents act (4) based on the profiles and process descriptions downloaded in step (2).

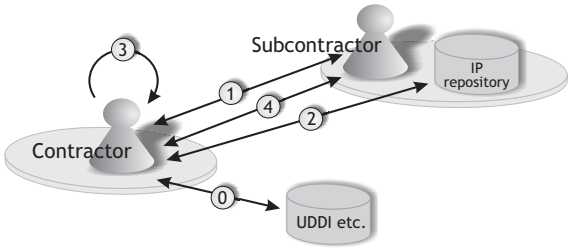

Figure 2: IP based B2B interaction

Related Work

Web Services initiative gathers a lot of industrial attention at the moment. Web Services are intended to facilitate businesses operating in the Internet to find each other and interoperate. WSDL (Chinnici & others 2002) is an important technology for implementing the Web Services framework. Service descriptions found in UDDI registries are often described in WSDL. However, WSDL is quite low-level language and incapable of describing semantic details of the services. In (Ankolekar & others 2002), the authors propose a way to map DAML-S to WSDL, thereby enriching Web Services with descriptions defined in the Semantic Web community.

Our work differs with the above in that we are not trying to merge agents and Web Services at the technology level. We are rather approaching the question at a higher level and

propose interaction protocols originally designed for software agents to be used to guide and restrict the communication between Web Services. We serialize IPs as DAML-S processes yet do not consider mapping them to WSDL.

Other related work includes (Freire & Botelho 2002), in which the authors consider representing FIPA interaction protocols in XML, and (Gaio & others 2002), which is about translating DAML-S into executable program code. Technologically speaking, our work is a combination of those two. We consider serializing IPs in DAML-S, and then we translate the protocols into executable program code. Our approach is however wider, and this interaction protocol serialization is just a small fraction of it, as mentioned earlier. Moreover, our outlining of an upper-level interaction protocol ontology is not visible in (Freire & Botelho 2002; Gaio & others 2002).

Other attempts to bring more agent-like communication facilities to Web Services architecture include Conversation Policy XML (cpXML) (Hanson & others 2002) and Web Services Conversation Language (WSCL) (Banerji & others 2002). These approaches utilize XML Schema as a language for describing the IP ontology. We believe DAML-S has some useful concepts already defined and thereby base our IP ontology to DAML-S. Further, cpXML starts with states as primitive elements whereas we take transitions between states as our primitive elements. WSCL is restricted to one-to-one conversations, has no notions of timeouts, and takes stand on the format (XML) of the documents sent.

Conclusions and Future Work

In this paper we presented a short outline of an interaction protocol ontology. It can be utilized as an upper-level ontology when designing interaction protocol instances for different purposes. An example application area would be interaction between a contractor and a subcontractor operating in the Internet, as depicted in Figure 2.

We proposed a serialization for the interaction protocol ontology in DAML-S. DAML-S is still under development. Among other things, DAML-S lacks specification of a conversation protocol, as mentioned the release status notes of (Martin & others 2002).

We pursue the goal of creating agents capable of externalizing and internalizing their knowledge step by step. In this paper we concentrated on interaction protocols (see Table 1) and supposed that the agents of our system are familiar with the CAs used in IPs. Some of the CAs are however based on others. For example, knowledge of INFORM-IF or PROPOSE presupposes knowledge of INFORM (Foundation for Intelligent Physical Agents 2002a). This allows the same treatment for certain CAs that we have in this paper proposed for IPs. We plan to investigate this in the future.

Moreover, even the simplest CAs such as INFORM could be included in DAML descriptions and taught to agents. This would require embedding the formal BDI-models of CAs in the descriptions. We see this as a faint possibility but not a very tempting one. Agents without any pre-knowledge of CAs whatsoever would be extremely stripped down and teaching them anything seems cumbersome at this point.

References

Ankolekar, A., et al. 2002. DAML-S: Web service description for the semantic web. In Horrocks, I., and Hendler, J., eds., *Proceedings of the First International Semantic Web Conference (ISWC2002)*, 348–363. Heidelberg, Germany: Springer-Verlag.

Banerji, A., et al. 2002. *WSCL 1.0 Specification*. http://www.w3.org/TR/wscl10/. Work in progress.

Chinnici, R., et al., eds. 2002. *WSDL 1.2 Specification*. http://www.w3.org/TR/2002/WD-wsdl12-20020709/. Work in progress.

Foundation for Intelligent Physical Agents. 2001. *FIPA Interaction Protocol Library Specification*. Geneva, Switzerland. Specification number XC00025.

Foundation for Intelligent Physical Agents. 2002a. *FIPA Communicative Act Library Specification*. Geneva, Switzerland. Specification number SC00037.

Foundation for Intelligent Physical Agents. 2002b. *FIPA Request Interaction Protocol Specification*. Geneva, Switzerland. Specification number SC00026.

Freire, J., and Botelho, L. 2002. Executing explicitly represented protocols. In *Proceedings of the 1st International Workshop on Challenges in Open Agent Systems, AAMAS'02*, 37–40.

Gaio, S., et al. 2002. From DAML-S to executable code. In *Proceedings of the 1st International Workshop on Challenges in Open Agent Systems, AAMAS'02*, 15–19.

Hanson, J., et al. 2002. Conversation-enabled web services for agents and e-business. In Arabnia, H., and Mun, Y., eds., *Proceedings of the International Conference on Internet Computing (IC'2002)*, 791–796. Las Vegas, Nevada, USA: CSREA Press.

Hutchins, E. 1996. *Cognition in the Wild*. Cambridge, MA: MIT Press.

Martin, D., et al. 2002. *DAML-S 0.7 Draft Release*. http://www.daml.org/services/daml-s/0.7/.

Toivonen, S., and Helin, H. 2002a. Options for reusing agent conversations. In Karmouch, A.; Magedanz, T.; and Delgado, J., eds., *Mobile Agents for Telecommunication Applications, 4th International Workshop, MATA 2002 Barcelona, Spain, October 23-24, 2002, Proceedings*, volume 2521 of *Lecture Notes in Computer Science*, 1–10. Heidelberg, Germany: Springer.

Toivonen, S., and Helin, H. 2002b. Task-sensitive adaptability for software agents. In Zemliak, A., and Mastorakis, N., eds., *Advances in Information Science and Soft Computing*, 50–55. Cancun, Mexico: WSEAS Press.

UDDI. 2000. The UDDI technical white paper. http://www.uddi.org/.

World Wide Web Consortium. 2001. Semantic web activity. http://www.w3.org/2001/sw/.

World Wide Web Consortium. 2002. Web services activity. http://www.w3.org/2002/ws/.