AN ONTOLOGY-BASED ARCHITECTURE FOR MULTI-AGENT SYSTEM ENVIRONMENT

Roberto Paiano, Anna Lisa Guido and Enrico Pulimeno
Dipartimento Ingegneria dell'Innovazione, SalentoUniversity, Via per Arnesano, 73100 Lecce, Italy

Keywords: Multi agent systems, ontology reasoning, domain independent, domain dependent.

Abstract: The increasing interest towards e-business systems, and thus the need of the companies to communicate with other companies in an efficient and flexible way, brings to a new way of thinking about information systems that open itself towards distributed systems where the exchange of information may happen. The problem is in the way that this exchange can be made both from the technology point of view and from the conceptual point of view. If from a side a recent technology for information exchange (DDS- Data Distribution Service) can help us from the other one it seem interesting the use of ontology as representation tool of application domain that must open itself to the communication. In this paper we present an architecture that is oriented to a multi-agent communication that uses ontologies and DDS as information exchange protocol. The architecture here presented must be repeat without difficulties on an e-business system assuming that each agent of the proposed architecture is a company in an e-business system.

1 INTRODUCTION

An e-business system is a system where involved companies communicate among them exchanging data in as much as possible efficient and flexible way with the goal to satisfy the final user. Two are the main problems that rise in this area:

- Conceptual problem related to the definition of a semantics of interchange of the data in order to assure the communication of the companies in heterogeneous environments;
- Technological problem related to the use of a suitable data interchange system useful to assure the communication between heterogeneous systems.

Both the problems are difficult to be faced and surely can be of great help the formal ontologies to solve the first problem (conceptual problem) and, relatively to the technological problem, it can come in help the DDS technology (Data Distribution Service) (DDS, 2001) standard OMG and surely useful not only for synchronous comminication among companies but also for the asyncronous one.

Very often, in fact, the companies don't need only information in real time: we can think, for example, to a business process that needs to have a data coming from another information system. The data requested from the business processes in order to go on with the execution are not immediately available. It is necessary a mechanism that allows to ask for the data and wait until the data is made available by some other information system that can provide it.

A base architecture useful to think about an e-business system based on a multi-agent has been presented in a research project lead by SSI (Space Software Italia) and the Department of Engineering Innovation of the University of Salento. The project was founded by Apulia region. Starting from a multi-agent system already made up from SSI in a demining system where each agent is represented by a robot, the goal of the project has been to enable the communication between agents through a level of intelligence made up by semantic web technologies, in order to define a run-time all the parameters to publish/subscribe depending on the operational context where agents work.

Naturally, when in the multi-agent system the parameters to publish/subscribe has been identify , it is necessary to avoid that they are manually published/subscribed in the DDS; it would be, therefore, useful that the classes (the parameters are published/subscribed in the DDS through classes) that materially define these parameters inside the DDS, are automatically produced.

Paiano R., Lisa Guido A. and Pulimeno E. (2008).
AN ONTOLOGY-BASED ARCHITECTURE FOR MULTI-AGENT SYSTEM ENVIRONMENT.
In *Proceedings of the International Conference on e-Business*, pages 101-106
DOI: 10.5220/0001913301010106
Copyright © SciTePress

The "intelligent" level, that it is important to add, it has to operate so that to allow "to understand", depending on the context, the parameters to publish/subscribe. To add this level, the formal ontologies are particularly useful. Not only the ontology but also the technologies of reasoning that allow from a side to define the context where the system operates and from the other to individualize the rules of action that each agent can take in order to answer to a well defined event.

The comparison with this kind of system and the e-business systems is immediate: every agent represents every information system involved in the information interchange. The "intelligent" level allows subsequently the different information systems to publish/subscribe in the DDS the data of interest to an appropriate system of reasoning that allows to identify the correct information.

In this paper we present a high-level architecture designed in the research project: the architecture is useful in order to allow the communication among the involved agents. As it will be clearer subsequently, the base idea will be that to make the application domain independent in comparison to the definition of the key elements that make possible the communication (DDS). Besides particularly interesting is the sharing of an only one knowledge base among several agents of the system.

After having introduced in the section 2 the state of the art to the multi-agent systems, in the section 3 will be introduced three alternatives considered as it regards the positioning of the knowledge base on the various agents that constitute the system. In the section 4 we present the architecture of the multi-agent system, finally, in the section 5, we present the conclusions of the paper.

2 BACKGROUND

The e-business systems have not been thought and developed, until now, according to a multi-agent logic. It is useful, however, to present the state of the art related to the multi-agent systems in order to understand the problem list related to the development of a system of this type.

In a multi-agent system we may speak about 4 main aspects:

- *decisional aspects related to the agent*: what actions an agent undertakes depending on the external environment;
- *outsourcing of the execution*: possibility that more agents collaborate together, performing elementary actions to complete a complex task;
- *interactions among agents:* information interchange among agents in distributed environment;
- *Evolution toward component more and more endowed with autonomy:* agents that in full autonomy reach their own goals.

The multi-agent systems need a study of the nature of the interactions. It results therefore important the notions of collaboration and cooperation.

Interesting, in this paper, to underline that for the realization of a multi-agent architecture it is necessary to have a layer of communication and a layer of conceptual modelling.

- **Communication Layer**: currently the most qualified is standard seems to be **FIPA-ACL** (FIPA, 2002) (Agent Communication Language) created from the Foundation for Intelligent Physical Agents (FIPA). This standard is founded on the linguistic action theory, elaborated by *John Searle* (Searle, 1969). An important implementation of the FIPA standard is the framework JADE (http://jade.cselt.it) an open source platform for peer-to-peer agent based communication developed by Telecom Italia Lab.
- **Conceptual Model Layer**: it is very important to identify the domain where agents operate and it is important the dynamics of interactions between agents.

Currently, particularly interesting within the modelling of the multi agent systems it results:

- The **BDI Model** (Beliefs-Desires-Intentions) (Chang-Hyun, Guobin et. Al, 1969): it considers the agent environment belief, which is the result of its knowledge and perceptions, and a set of Desires. Intersecting these two sets, we obtain a new set of intentions, which can become actions.
- **Tropos** (Bresciani, Perini et. Al, 2004) a software development methodology founded on concepts used to model early requirements. In particular, the proposal adopts Eric Yu's modelling framework, which offers the notions of actor, goal and dependency, and uses these as a foundation to model early and late requirements, architectural and detailed design. The language used in Tropos for the conceptual modelling is formalized in a meta-model described with a set of UML class diagram.

In literature there are several papers that examine the software engineering paradigm applied to the multi-agents system, among them an article of Pratik K. Biswas (Patrik, Biswas, 2007) describes

extensively the multi-agent system elements and their correlations, confronting them with UML paradigm that it reuses for their modelling.

Very interesting in a multi-agent system is the ontological approach very useful to provide an explicit and formal representation of the domain. This representation is simple to realize and it is simple to exchange between agents thanks the ontological languages such OWL (W3C, 2004).

3 USE OF KNOWLEDGE BASE IN MULTI-AGENT SYSTEM

An e-business system thought in the multi-agent terms it will have, for the complexity of the scenarios in which it will operate, an elevated degree of hardiness and modularity. To reach this goal it is important to make more independent possible the various agents (nodes) of the system, that they need a continuous interchange of data in order to reach their own goal.

The first problem to face is that to add to a multi-agent system a layer of "intelligence" that is able to provide a good level of autonomy and facility of updating the system, besides it provides the possibility to describe the several events that are verified inside the system and to which the agents must answer. This level of intelligence is constituted by an ontology (opportunely supported by a system of rules) that it results particularly useful to guarantee a *complete understanding of the domain* to all the interested agents, and accordingly one swifter change of his in case of changes or the insertion and management of unexpected events.

In the use of ontology in the e-business systems based in multi-agents it is had to analyze with attention as it must be defined and above all if this ontology has to be positioned only on a node inside the system or must be distributes on different nodes. We describe 3 hypotheses of work individualized defining among them the most suitable to the context in which we work.

3.1 Centralized Knowledge Base

The centralized system foresees that all the agents that cooperate make reference to a knowledge base centralized on only one agent. This system involves that, all the information push through the central node which stores them in the knowledge base and elaborates them.

The advantages of this approach are:

- the system is simple to manage;
- presence of a supervision node from which it is possible to access all the information and to provide precise commands to the other agents;

The disadvantages are:

- The supervisor node results a critical Point of Failure, in fact if the supervisor node had to come less for some reason the knowledge base would result unreachable from the other nodes and, therefore, the overall system go down.
- Agents have little decision autonomy.

3.2 Knowledge Base Total Distribute

In a structure that introduces an high degree of distribution, all the agents work in independent way and the exchange of data it has the goal to make possible the coordination among the agents. All the agents are in communication exploiting a system of connection reliable (the DDS) that limit the Point of failure. At ontology level, for this scenario two different solutions can be identified, one that foresees to repeat the whole knowledge base of domain on every agent and the other is to foresees the distribution of the domain ontology so that to put on every agent only the part of knowledge base that interests the specific agent for the carrying out of his/her own role in the system.

3.2.1 Knowledge Base Replied on the Agents

In this structure, in which knowledge base is repeat on every agent every node of the system, knows the whole domain.

The advantages of this approach are:

- Non-existence of a Point of Failure;
- The autonomy degree of the agent enhance
- All the agents have peer decisional ability exploiting the knowledge base;

The disadvantages are:

- Every change on the knowledge base has to be repeat on every agent;
- The system needs a good structure of coordination;
- The complexity of each agent enhances.

3.2.2 Knowledge Base Distribute on the Agents

Another solution foresees the distribution on the several agents of the description of the domain that must be decomposed in modules depending on the specific criterions of competence. In other words, it deals with decomposing in several functionalities the ontology and to implement them on the several agents based on the demands of these.

The advantages of this approach are:

- Least redundancies of the data, every agent has only the information of which it has need;
- Absence of a single Point of Failure, and therefore greater independence of the robots;
- Possible changes to be brought to the ontological structure of the data base would be alone on the single agent and not on everybody.

The disadvantages are:

- Planning is complex because each agent is complex.
- Limited decisional ability for the agents.

3.3 The Selected Solution

The selected solution is to have a distributed system with a knowledge base repeat on every agent of the operational context. In this way the several nodes that operate in the system will be fully autonomous and able to develop his/her own task without depending in some way on the other participants to the mission, also communicating and exchanging data, in continuous way, with the other agents.

4 LOGICAL ARCHITECTURE OVERVIEW

Before introducing the logical architecture conceived it is fundamental to clarify shortly the operation of the middleware of communication selected.

4.1 Use of the DDS as Tool for the Communication among Agents

The DDS is a useful tool proposed by the OMG that enable the communication within data-centric systems. The system of communication is asynchronous and it is based on the publishing/subscribing protocol: when agent that operates in the system has the necessity of a data, it makes a subscribing of it pointing out the data of which it has need; when some other agent of the system makes the data available, it effects a publishing of it. The agent that has previously effected the subscribing is able, to this point, to get the data of interest. The data publishing/subscribing is encapsulated in a class and send to the DDS. Since information may change depending on the particular operational context, also the relative publishing/subscribing class can vary depending on the particular data to send to the DDS.

4.2 Goal in the Realization of the Architecture

The idea that is at the base of the conceived architecture is born from two fundamental requisite:

- *Decouple* the decisional aspect related to the publishing/subscribing of well defined information, from the technological aspect tied up to the necessity to produce on the fly the useful classes to publishing/subscribing the parameters in the DDS
- To provide an high flexibility level to the system in this way to allow an *adaptation* of the same to the different operational scenarios where the architecture could operate, also without denying the consequential potentialities from the existence of a middleware able to guarantee the communication among the various agents.

Having decided to use a semantic base replied on every agent of the environment, we decided to decouple the component of the knowledge base of domain from that related to the management and representation of the concepts that describe the middleware of communication (DDS).

The proposed decoupling makes the system independent from the particular operational context, providing a flexible and easily adaptable structure.

The proposed architecture is in fig. 1.

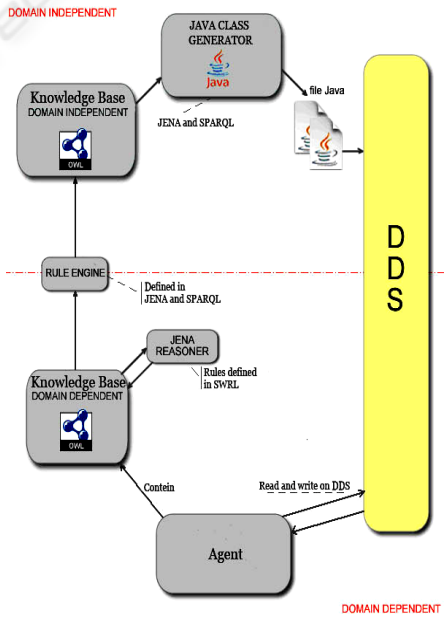

Figure 1: Logical architecture.

First of all we observe the presence of two layers one called "domain independent" and one called "domain dependent". Within these two layers,

interconnected through a system of rules as will be detailed after, the presence of two knowledge bases is observed:

- **Knowledge Base *Domain Dependent***: this module represents the KB specific of a special operational context.
- **Knowledge Base *Domain Independent***: this module represents several elements that constitute the DDS.

We can observes, within the layer "domain dependent" the presence of an "agent". It represents a generic agent that will contain both the knowledge base domain dependent and the knowledge base domain independent and that will make operation of publishing/subscribing of the information on the DDS.

4.3 Ontology Domain Dependent Module

The knowledge of the context, in which the agents work, and of its dynamics, it allows to be able to manage in the best way the events that can be verified in it. The module has the goal to represent the application domain in which the agents operate and the possible semantics relationships existing among the information that the various agents could exchange. In this module it will be present, therefore, this domain ontology that must have realized from the expert of domain that, better of everybody, it is able to define every aspect of it. In the knowledge base it is possible to find the specific information about the domain and their semantic relationships as an example if the domain is an information system that manage data about environment, the concept in the knowledge base will be the *sensor* used to obtain data or the *area* where data will be obtained and these two concepts will be related each other in order to provide the semantic link. For each sensor it will be possible to define the *period of time* when the data will be obtained and the threshold values defined in the specific context. These information will constitute a small part of the overall domain ontology that describe the specific information system.

4.4 Ontology Domain Independent Module

In this module there is the ontology that represents the specifications of the substratum of communication chosen for this architecture. This knowledge base reproduces, of fact, the OMG specification for the DDS: this knowledge base will be completed with the individuals, depending on the specific necessities, from the information obtained following inference, from the KB domain dependent. For example, if it is important, for the information system to publish/subscribe a well specific data obtained by the sensor, for example the *temperature* data, the information useful to make a topic for the temperature will be moved from the knowledge base domain dependent to the knowledge base domain dependent.

4.5 Jena Reasoner Module

The Jena reasoner module have the task to realize, through the application of ad-hoc SWRL rules written by the business expert, the deductions on the knowledge base that allow to intercept events that must happen and that they require, eventually, of a publishing/subscribing of information on the DDS with the goal to complete one determined activity. With an opportune analogy, every information system within an e-business system will deduce from the domain ontology what information to publish/subscribe in order to communicate with the other systems.

4.6 Rule Engine Module

The rule engine module will provide, through the opportune rules defined in the SPARQL language, to extract from the KB domain dependent the information to publish/subscribe and will add individuals to the ontological classes that represent, instead, the DDS (KB domain independent).

4.7 Java Class Generator Module

Since the system of publish/subscribe of the DDS founds him, of fact, on a system of publish/subscribe of classes, the presence of this module is fundamental because it able to produce, beginning from the individuals of the KB domain independent, the useful classes to make the operation of publish/subscribe. The class draws the information that the domain expert associates to every topics (“topics” are the elements that able the communication in the DDS) this information, through the rule engine module, is repeat in the KB domain dependent and from here used for the creation of the relative classes.

4.8 Description of the Information Flow

The information flow in the architecture can be so defined:

- The agent captures the information stored in the knowledge base domain dependent;
- The Jena reasoner module makes the appropriate deduction through the SWRL rule engine;
- The RULE ENGINE module defines the mapping rules between the two knowledge base (domain dependent and domain independent).
- The java classes generator generates the .java file and the corresponding .class file;
- The .class file will be used to send information through the DDS.

It is important to observe that the separation between knowledge base domain dependent and knowledge base domain dependent increases the flexibility level of the system. Changing the context of business in which the system operates mean to define the knowledge base domain dependent and the system of reasoning (module rule engine and Jena reasoner) leaving the dynamics of communication made up through the DDS.

5 CONCLUSIONS AND FUTURE WORKS

In this paper we propose an architecture conceived for the communication of a data centric multi-agent systems that they use as middleware of communication the DDS recently proposed by OMG.

Particularly interesting is the analogy among the data-centric systems and the multi-agent systems applied to an e-business context. The analogy can be done thinking about every information system involved in the e-business environment as a single agent of the architecture here presented. In this way it will be possible to bring on a e-business system the whole efficiency and the flexibility that the proposed architecture introduces: the great advantage obtainable is the separation between the KB domain independent and the knowledge base domain dependent, another advantage is the possibility to produce in automatic the useful classes to make the data publishing/subscribing of the data.

Naturally, repeating the architecture proposed on an e-business system there are many aspects to clarify that brings to several research ideas.

The first problem is surely tied to the semantic interoperability: it is necessary to build a knowledge base that describes the whole domain in which every information system participates but as it is possible to effect the transfer of the information proper of every informative system (which those present in the database) in format compatible with those described in the ontology domain dependent? And what mean, practically, to describe a domain?

Parallels to these problems, the problem of reasoning is still open: few has been done in international scientific community in this sense so much that doesn't exist, until now, a standard language universally recognized for realizing the reasoning. In this architecture SWRL is used but this doesn't exclude, in a next future, the use of a language more efficient.

In every case, is interesting the idea at the base of the present paper that consists of using in an e-business system the formal ontologies and the mechanisms of reasoning.

ACKNOWLEDGEMENTS

We would thank SSI (Space Software Italia) company and Antonella Falzone for the tangible support.

REFERENCES

Bresciani, P., Perini, A., Giorgini P., Giunchiglia, F., Mylopoulos, J. 2004 Tropos: An Agent-Oriented Software Development Methodology. *In Autonomous Agents and Multi-Agent Sytems, 8, 203–236, 2004*

Chang-Hyun, J., Guobin, C., James, C. (2004) A New Approach to the BDI Agent-Based Modelling *In 2004 ACM Symposium on Applied Computing*

Data Distribution Service for Real-time Systems Version 1.2 OMG Available Specification formal/07-01-01

FIPA ACL Message Structure Specification 1996-2002

Pratik K. Biswas, Towards Agent-Oriented Conceptualization and Implementation. 2007, IGI Global,

Searle, J. 1969 Speech Acts: An Essay in the Philosophy of Language, Cambridge University Press, 1969 ISBN 052109626X

W3C (February, 10 2004) “OWL Web Ontology language Reference” W3C