Supporting Agents in Intelligent Environments with Protocol Information *

Carolina Felicíssimo[1], José Viterbo[1], Luis Valente[1], Markus Endler[1], Jean-Pierre Briot[2], Carlos Lucena[1], Bruno Feijó[1]

[1] Department of Informatics
Pontifícia Universidade Católica do Rio de Janeiro, Brazil
{carol,viterbo,lvalente,endler,carlos,bruno}@inf.puc-rio.br

[2] Laboratoire d'Informatique de Paris 6 (LIP6)
Université Paris 6 - CNRS, France
Jean-Pierre.Briot@lip6.fr

Keywords: Open multi-agent systems, contextual protocol information

Abstract

Open Multi-Agent Systems (MAS) can be extremely dynamic due to heterogeneous agents that migrate among them in order to obtain resources or services not found locally. In this scenario, it is not reasonable to expect that foreign agents know in advance all the information of the MAS in which they will execute. Thus, this paper proposes DynaCIP, our approach for continuously supporting mobile agents with contextual information and presents a prototype application.

1 Introduction

Openness has led to software systems that are formed by autonomous entities [9] which may migrate among different systems in order to obtain resources or services not found locally. Thus, open systems can be extremely dynamic. In this work, we assume that a multi-agent system is an open system that puts together sets of heterogeneous, self-interested agents whose actions may deviate from the expected behavior in a given context.

An agent — human or artificial — originally has some previous knowledge about how to act in its own and, maybe, it also can have some learning skills for acquiring knowledge during its execution. When migrating among others systems, it is assumed that agents know how to act. For this, agents should have the knowledge a priori (i.e., implemented inside it) or they can obtain it while executing. However, it is not reasonable to assume that an agent will have the knowledge a priori about all systems in which it will play due to their varieties. This knowlege about how to act, may be comprised in specific protocols, i. e. sets of directives, which are action prescriptions that should be followed in order to fulfill the conditions for achieving a goal.

*Work partially funded by projects CNPq 55.2068/2002-0 (ESSMA), CNPq 47.9824/04-5 (Ed. Universal), CAPES/Cofecub 482/05/07 (EMACA) and CNPq individual grants

This paper proposes DynaCIP (meaning *dynamic contextual information provision in open multi-agent systems*), our approach for continuously providing the precise information about protocols, so that, context-aware agents can perform efficiently and coherently in MASs. We work with an action and effect approach, i.e. agents aware of the valid protocols in a given context are more likely to perform correctly and so, can achieve their goals faster.

The remainder of this paper is organized as follows. Section 2 describes a typical scenario for our system. Section 3 presents the DynaCIP solution, including how to classify, represent and compose contextual protocols. Section 4 describes an example explaining how effectively using DynaCIP. Section 5 describes a prototype application implemented with the support of DynaCIP. Section 6 presents a comparison with some related works. Finally, we draw our conclusions and outline future work in section 7.

2 Scenario

As a typical scenario to exemplify our DynaCIP approach, we consider two universities in two different countries, for instance, PUC-Rio in Brazil and Paris VI in France. We assume that both organizations have a location service capable of determining the position of agents inside them (such as the MoCA's Location Inference Service [17]). This information is given in terms of symbolic location, which associates a name with each of the smallest areas distinguishable by the location service. Those atomic spaces may be organizations, laboratories, seminar rooms, offices, corridors, etc., and they are populated with agents that have some association with the organizations (e.g., professors, students, researchers, administrative staff members). We further assume that the two organizations have some sort of cooperation, so that a member of one institution may be a temporary visitor at the other institution.

We propose a motivating situation in which agents interact for scheduling a meeting: Caroline, a student from Paris VI, arrives in PUC-Rio for a internship of one year. Caroline wants to schedule a meeting with Walter, a professor from PUC-Rio, that will be her advisor during the internship. Caroline knows that if she were in LIP6 she would

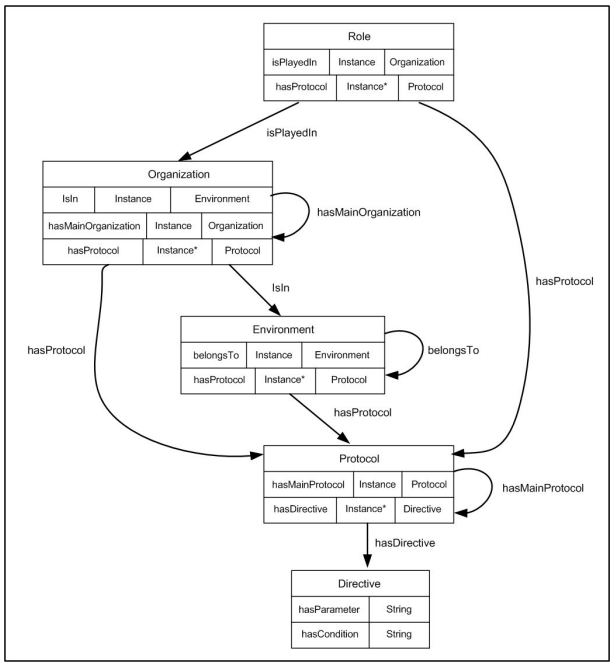

Figure 1: The DynaCIP ontology.

send an email to her advisor for scheduling the meeting. Walter does not fix the date of their meeting because he doesn't know his agenda for meetings. But in PUC-Rio, for scheduling the meeting with Walter, Caroline should send an e-mail to Vera, his secretary, to arrange everything.

3 Contextual Protocol Information Provision in MAS

DynaCIP aims to support agents in MASs with updated contextual protocol information. For this, developers should follow the DynaCIP approach to classify, represent and compose their contextual protocols. A system implemented under the DynaCIP approach results in a dynamic MAS in which information about the set of protocols that are valid in a given context is continuously provided to agents. Modular context refinements allow a more flexible system for developers while they are managing protocol information.

3.1 Protocol Classification

Basically, an MAS is constituted of environments, organizations and agents playing roles and interacting [10], suggesting different context scopes for information provision in MASs. Context [3] is any implicit information that can be used to characterize the situation of agents and to provide relevant information and/or services to them, where this relevancy depends on the agent's tasks.

DynaCIP follows directions taken by research into context-aware applications that suggest top-down architectures for classifying contextual information [8] [13]. In DynaCIP, protocols should be classified in at least the Environment, Organization, Role and Interaction context scopes. This set should be improved with additions and refinements of protocols for representing particular domain context scopes. Context scopes are differentiated by their boundaries. More precisely, Environment Protocols are applied to all agents of an environment; Organization Protocols are applied to all agents of an organization; Role Protocols are applied to all agents playing a role; and Interaction Protocols are applied to all agents involved in an interaction.

3.2 Protocol Representation

DynaCIP use a contextual ontology to explicitly represent protocol information, having the `Protocol` concept as a central asset. An ontology is a conceptual model that embodies shared conceptualizations of a given domain [7]; and a contextual ontology is an ontology that represents localized domain information [2]. The use of ontologies in MASs supports heterogeneous agents with a common understanding about well-defined system information relating abstract concepts, in which information are formulated, to their concrete application domain.

The DynaCIP ontology defines five related concepts: `Role`, `Organization`, `Environment`, `Protocol` and `Directive`, as illustrated in Fig. 1 by multi-lines linked boxes. In each concept, the first line contains the concept's name/identification and the other lines correspond to the concept's attributes. Each attribute's line can be divided into two or three parts. The first part contains the attribute's name/identification. The second part can have a datatype property (e.g., String, Integer, etc.) or, if the attribute corresponds to an object property, this part brings the attribute's cardinality (i.e., `Instance` for a unique value and `Instance*` for n-vary values) of a property, while the third part brings the another concept that is linked to that one by the described property. For example, the first line of the Role concept has `Role` as the concept's name; the second line has the object property `isPlayedIn`, which links the `Role` to a unique instance of the `Organization` concept; and the third line has the multi-value object property `hasProtocol`, which links the `Role` to several instances of the `Protocol` concept. As an another example, the first line of the Directive concept has `Directive` as the concept's name; the second line has the datatype property `hasParameter`, which is assigned to a String multi-value, and the third line has the datatype property `hasCondition`, which is also assigned to a String multi-value.

Among the five concepts in the DynaCIP ontology, Role, Organization and Environment represent context information that characterizes an agent's situation. Each Role instance has associations with its organization (`isPlayedIn` property); each Organization instance has associations with its main organization (`hasMainOrganization` property) and Environment (`isIn` property); and each Environment instance has associations with its owner environment (`belongsTo` property). The Protocol concept encompasses the instances of all protocols to be followed. Each Protocol instance has associations with its main protocol (`hasMainProtocol` property) and directives (`hasDirective` property). Pro-

tocol instances are related with instances of Role, Organization and Environment (`hasProtocol` property), expressing that these protocols are only valid in a given scope (i.e., contextual protocols). The Directive concept encompasses the instances of all directives to be followed. Each directive instance has a respective String field for its parameters and conditions.

Interaction protocols may be described in the DynaCIP ontology defining a new Protocol concept which links two interacting Role concepts. This solution follows the representation pattern presented in [16] which defines that the relation object itself must be represented by a created concept linking the other concepts from the relation (i.e. reification of relationship).

3.3 Protocol Composition

After classifying protocols in precise levels of abstractions and representing them in an ontology instance, their information can be composed during system's execution since, at any given moment, an agent may be subjected to protocols defined at one or more context scopes. Compositions of contextual protocols result in sets of independent information, in which the semantic of one protocol can influence the semantic of the others. Updates in a DynaCIP domain ontology instance and different compositions of related contextual protocols, both at run-time, provide the dynamism and flexibility necessary for information management regarding social changes characteristic of MASs.

Code 1 : Rules to hierarch DynaCIP contextual protocols

```
(1)  Rule1 [ruleForEnvWithOEnvProtocols:
(2)    hasProtocol(?Env,?OEnvProtocols)
(3)  ← hasProtocol(?OEnv,?OEnvProtocols),
(4)    belongsTo(?Env,?OEnv)]

(5)  Rule2 [ruleForOrgWithMOrgProtocols:
(6)    hasProtocol(?Org,?MOrgProtocols)
(7)  ← hasProtocol(?MOrg,?MOrgProtocols),
(8)    hasMainOrganization(?Org,?MOrg)]

(9)  Rule3 [ruleForOrgWithEnvProtocols:
(10)   hasProtocol(?Org,?OrgEnvProtocols)
(11) ← hasProtocol(?OrgEnv,?OrgEnvProtocols),
(12)   isIn(?Org,?OrgEnv)]

(13) Rule4 [ruleForRoleWithOrgProtocols:
(14)   hasProtocol(?Role,?OrgProtocols)
(15) ← hasProtocol(?Org,?OrgProtocols),
(16)   isPlayedIn(?Role,?Org)]

(17) Rule5 [ruleForProtocWithMProtocDirectives:
(18)   hasDirective(?Protoc,?MProtocDirect)
(19) ← hasDirective(?MProtoc,?MProtocDirect),
(20)   hasMainProtocol(?Protoc,?MProtoc)]
```

DynaCIP uses rules to describe the composition of its contextual protocols and is able to execute the following task, by using an inference rule engine: (i) read an ontology instance to get data (i.e., concept instances and relationships), (ii) read active rules to get how concepts must be composed, and (iii) infer an ontology instance based on the previous readings. Thus, resulting information is always updated due to the ontology instance and active rules.

DynaCIP rules are ontology-driven rules, i.e. they are created according to the ontology structure and limited by the number of related concepts to which each concept is linked. All DynaCIP predefined rules are presented in Code 1. Inputs for these rules are domain instances of the Environment, Organization, Role and Protocol concepts and outputs are compositions of their related contextual protocols. For instance, Rule1 (line 1 to 4) from Code 1 states that each environment will have its protocols composed with the protocols of its owner environment. More precisely, the following process is executed: in (4), for a given environment `?Env`, its owner environment `?OEnv` is discovered; in (3), the protocols `?OEnvProtocols` of the owner environment `?OEnv` are discovered; and in (2), the discovered `?OEnvProtocols` are established with new properties `hasProtocol` linking the `?Env`, meaning that those protocols are composed with the protocols of the `?Env`. During the inference process this rule is applied to each environment instance.

Following the same composition process, Rule2 (line 5 to 8) states that each organization will have its protocols composed with the protocols of its main organization; Rule3 (line 9 to 12) states that each organization will have its protocols composed with the protocols of its environment; Rule4 (line 13 to 16) states that each role will have its protocols composed with the protocols of its organization; and Rule5 (line 17 to 20) states that each protocol will have its directives composed with the directives of its main protocol.

4 DynaCIP At Work

In this section we dicuss all the steps needed to instantiate the DynaCIP ontology for a specific application. As a case study we discuss the ontology that describes the scenario presented in Section 2.

4.1 Setting the Stage

First of all, we observe that the institutions, spaces and roles in our scenario must be represented using the DynaCIP ontology. Accordingly, in this example, PUC-Rio and Paris VI are instances of the Organization concept; Brazil, France, South America and Europe are instances of the Environment concept; and, Professor, Student, and Secretary are instances of the Role concept. So, the agents that populate and interact in this system can have their contextual situation completely described by the ontology, that is, the agent plays a role (represented by an instance of Role), which `is played in` an organization (an instance of Organization), which `is in` an environment (an instance of Environment). For example, we can think of Caroline as a student (instance of Role) at Paris VI (instance of Organization) in France (instance of Environment), and Walter as a professor (instance of Role) at PUC-Rio (instance of Organization) in Brazil (instance of Environment).

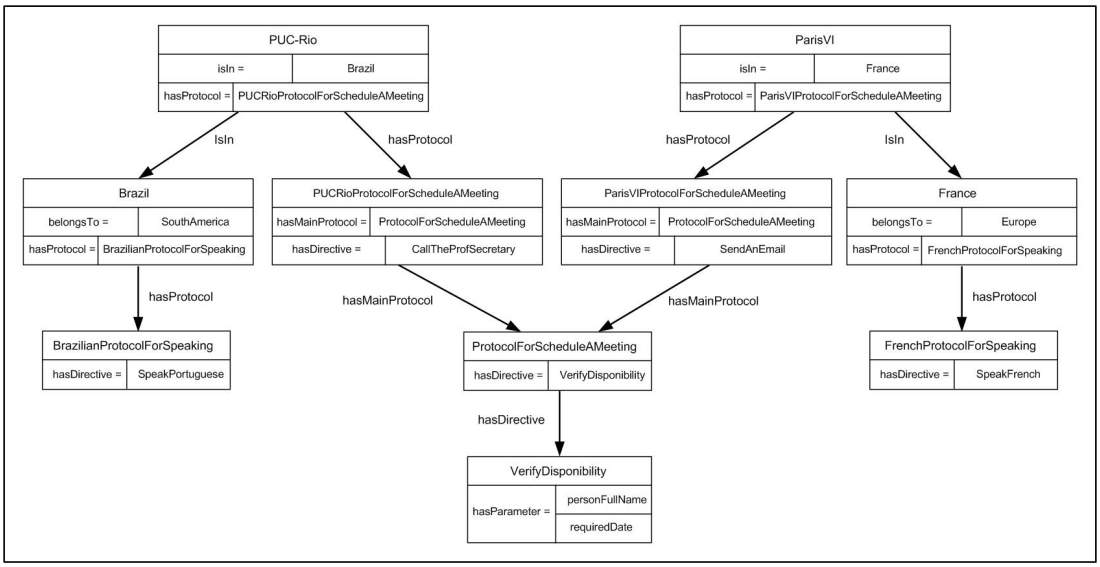

Figure 2: Contextual protocols for scheduling meetings.

4.2 Classifying Domain Protocols

Usually, organizations do not make their protocols public, thus, we created contextual protocols for our example and organized them in the contexts in which they apply.

4.2.1 Environment Protocols

Environment Protocol for Speaking. In all countries, a national idiom is spoken by their habitants. Contextual Environment Protocol for Speaking: (a) In Brazil, Portuguese is the national idiom spoken by Brazilians; (b) In France, French is the national idiom spoken by French.

4.2.2 Organization Protocols

Organization Protocol for Scheduling a Meeting. All attendees' agenda should be verified before fixing a date for a meeting. Contextual Organization Protocol for Scheduling a Meeting: (a) In PUC-Rio, some professors have secretaries responsible for scheduling meetings in their agenda; (b) In Paris VI, students should send an email to verify date availability for scheduling a meeting.

4.2.3 Role Protocols

Role Protocol for Answering Emails. Professors should answer their academic emails in less than one week. Contextual Role Protocol for Answering Emails: (a) PUC-Rio professors should answer their academic emails in less then four days.

4.2.4 Interaction Protocols

Interaction Protocol for Scheduling a Meeting. PhD students should ask their advisors or advisors' secretaries for scheduling a meeting. Contextual Interaction Protocol for Scheduling a Meeting: (a) In PUC-Rio, Prof. Walter's PhD students should call Vera, his secretary, for scheduling a meeting.

4.3 Representing and Composing Domain Protocols

DynaCIP explicitly represents domain contextual protocols in an ontology instance and uses rules to compose those. For example, Figure 2 illustrates part of a DynaCIP domain ontology extended and instantiated focusing on the contextual organization protocols for scheduling meetings of our example.

The contextual organization protocols of PUC-Rio and Paris VI are represented, respectively, by the `PUCRioProtForScheduleAMeeting` and `ParisVIProtForScheduleAMeeting` protocols instances, which have the `ProtocolForScheduleAMeeting` as their main protocol instance. The directives of the protocol instances are composed with the directives of their main protocol according to Rule5 from Code 1. More precisely, the following process is executed: in (20), the `ProtocolForScheduleAMeeting` is discovered as the main protocol of the `PUCRioProtForScheduleAMeeting`; in (19), its `VerifyDisponibility` directive is discovered; and in (18), this directive is composed with the `CallTheProfSecretary` directive from the `PUCRioProtForScheduleAMeeting` protocol.

The `PUCRioProtForScheduleAMeeting` organization protocol is also composed with its related environment protocol `BrazilianProtForSpeaking` (inherited from Brazil), according to Rule3 from Code 1. More precisely, the following process is executed: in (12), Brazil is discovered as the environment of PUC-Rio; in (11), its `BrazilianProtForSpeaking` is discovered; and in (10), this protocols is composed with the PUC-Rio protocol.

The Caroline student, playing a PhD student role from PUC-Rio, receives the contextual protocols from PUC-Rio according to Rule4 from Code 1. More precisely, the fol-

lowing process is executed: in (16), PUC-Rio is discovered as the new organization of Caroline; in (15), its protocols `PUCRioProtForScheduleAMeeting` (inherited from PUC-Rio) and `BrazilianProtForSpeaking` (inherited from Brazil) are discovered; and in (14), these protocols are composed with the Caroline's protocol. The result is that now she knows that she should call Vera, speaking in Portuguese, for scheduling a meeting with Prof. Walter.

5 Prototype Application

As a prototype application we implemented a simple location-aware quiz game to run on a Nokia N95 Smartphone. In this game, a (*human*) player must answer some questions that are posed depending on both his location and the level he has achieved in the game. The purpose of the player is to go through a number of rooms, answering some questions in each, while upgrading his level in the game, until he gets the last level and becomes a winner. At this moment he gains the permission to create and post one new question per game. In a certain institution, each room will be associated with a subject. Starting from *beginner*, the player may sequentially assume the other four roles: *intermediate*, *advanced*, *expert* and *winner*. Hence, starting from a given room in the *beginner* level, he will have to correctly answer an amount of questions about the associated subject to go to the *intermediate level*, when he changes rooms and go on answering questions to get to the next level. There are two kinds of questions, the 'easy' ones, that are proposed for *beginner* and *intermediate* players and the 'difficult' ones, proposed to *advanced* and *expert* players. After correctly answering the due questions as an *expert*, the player becomes a *winner* and may post his own question.

Figure 3 shows some screen shots from the UbiQuiz for a player in the *expert* level. In the screen shot at left, the question "How many frequency bands has a GSM Network?" (translated from Portuguese) is shown with three possible answers. The screen shot in the middle shows the message indicating that the answer of the player was correct, and, acknowledging that the player now is a *winner*, and so proposes that he sends his own question. The last screen shot shows the interface for submitting an answer.

The game application was implemented as an UbiQuiz Agent (UQA) that communicates with a client application in the smartphone to send the questions and to collect the answers. UQA uses DynaCIP to get the appropriate protocol depending on the location of the user and on his level in the game. Instances of the Environment class represent the possible locations in an institution, and instances of the Role class represent the level of the game. For example, an organization protocol for PUC-Rio would pose questions about technology; an environment protocol for Room 512 would pose questions about telecommunications, and a role protocol for *expert* players would pose questions evaluated as difficult. Accordingly, DynaCIP would indicate "Answer difficult questions about telecommunications" when a player with level *expert* interacts with UQA in the Room 512 of PUC-Rio, for example.

JADE [1] was used to implement UQA and J2ME to implement the client on the mobile device. The JENA API [14] was used for dealing with ontologies and ontology-driven rules. UQA was implemented extending the JADE Agent class with a specific Informative behavior which continuously informs each agent about its proper contextual protocols, representing effectively the DynaCIP behavior. This process continuously executes during the agent life-time because DynaCIP is implemented as an active behavior.

The location of a player is continuosly determined using MoCA/MAX (MoCA/Multi-Agent eXtension [21]), a multi-agent extension of MoCA architecture. MoCA [17] is a middleware that supports the implementation of context-aware applications. MoCA offers a location service capable of inferring the location of a mobile device from the information about RF signal patterns collected from 802.11 access points [15]. The service keeps track of the location of any registered device of the system. The location information is provided in terms of symbolic regions, which correspond to well-defined physical areas in a given organization (i.e. rooms) in which location-aware applications may be interested.

6 Related Work

Mechanisms to constrain the behavior of interacting agents in an open system are an increasingly popular approach to dynamic adjustability of applications. We highlight two well known systems with such a purpose, Kaos and Rei, that may be compared with our approach. Systems for modeling and coordinating organizations of agents, such as OMNI, have also a use similar to our system.

KAoS [19] consists in a set of platform-independent services that allow the definition of policies to ensure the adequate control over distributed systems. It is one of the first systems to use OWL [6], i.e., description logic, to describe and specify policies, and context conditions. Therefore, KAoS is able to classify and reason about both domain and policy specifications based on ontological subsumption, and to detect policy conflicts statically, i.e., at policy definition time. However, a pure OWL approach encounters some difficulties with regard to the definition of some kinds of policies — particularly those requiring the definition of variables. Relying purely on OWL, it is not possible to define policies that refer to statically unknown values [18].

Rei [12] is a flexible and expressive policy language that is based on deontic concepts, i.e. logic programs, and which can be used to describe several kinds of policies. It allows the description of individual policies as well as group and role based policies. Differently from KAoS, Rei's rule-based approach enables the definition of policies that refer to dynamically determined values, i.e. context variables, thus providing greater expressiveness and flexibility to policy specification [18]. On the other hand, the choice of expressing Rei rules similarly to declarative logic programs prevents it from exploiting the full potential of the OWL language.

OMNI (Organizational Model for Normative Institu-

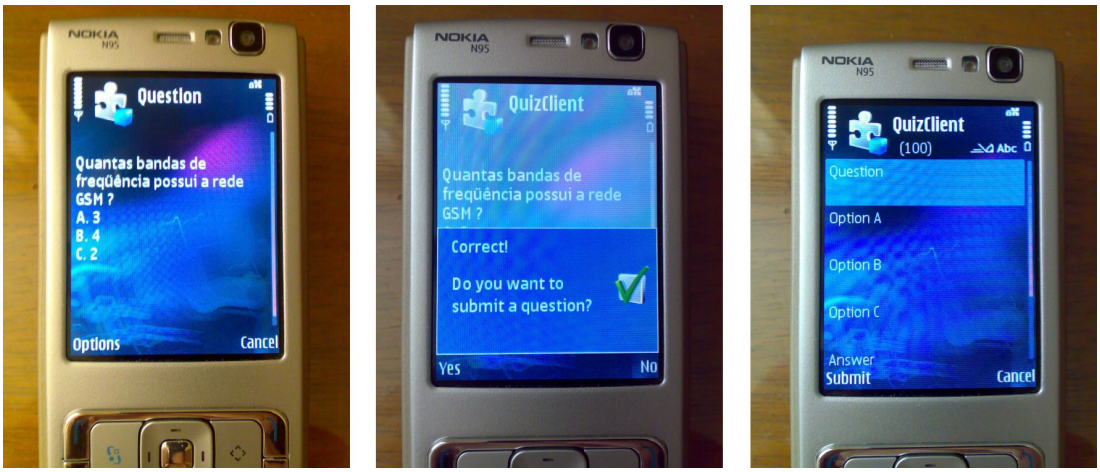

Figure 3: Screen shots showing three steps of UbiQuiz for a player in the Expert level.

tions [4]) is a framework for modeling agent organizations composed of three dimensions: Normative, Organizational and Ontological. OMNI contains the three levels of abstractions with increasing implementation detail: the Abstract Level, which has the statutes of the organization to be modeled, the definitions of terms that are generic for any organization and the ontology of the model itself; the Concrete Level, which refines the meanings defined in the previous level, in terms of norms and rules, roles, landmarks and concrete ontological concepts; and, finally, the Implementation Level, which has the Normative and Organizational dimensions implemented in a given multi-agent architecture with the mechanisms for role enactment and for norm enforcement.

In comparison to both KAoS and Rei, our approach has the advantages of allowing the definition of a protocol composition process, based on ontology-driven rules, that makes it easy to update the system information. Besides we propose a top-down methodology to define the classification for contextual protocols, which facilitates the tasks of elucidation, organization and management of protocol information.

In DynaCIP, protocols can be modelled with norms. In this sense, comparing our work with OMNI, both define a meta-ontology with a taxonomy for regulations in open MAS and use norms to recommend right and wrong behavior. The use of norms can inspire trust in the regulated MAS. One difference is that, in OMNI, enforcement is carried out by any internal agents from the system, while in our work it can be carried out by agents or not. A second difference, and the most important, is that in OMNI the idea of regulatory contexts is not explicit, especially for the environment and role law contexts. Our approach is based on the environment, organization, role and interaction regulatory contexts to simplify the enforcement and evolution processes. For instance, the social structure of an organization in OMNI describes, in the same level of abstraction, norms for roles and groups of roles. Group of roles is used to specify norms that hold for all roles in the group. We use the organization regulatory context to specify organization (or sub-organizations) norms that hold for all roles from an organization and use the role regulatory context to specify role norms, both regulatory contexts from different levels of abstractions.

7 Conclusion

The motivating question of our research is how to easily implement open MASs permitting heterogeneous agents to perform efficiently and coherently. The thesis we held here is that the complexity of protocol management in MASs can be decreased by decoupling information in precise levels of abstractions (i.e., in contextual protocols).

Our ongoing work, named DynaCIP, intends to be a straightforward method for smoothly providing protocol information in MASs for their agents. We believe that it has already achieved to bring some important contributions for the domain of MASs. DynaCIP's main contributions are: (i) a definition of a top-down classification for contextual protocols, which facilitates the tasks of elucidation, organization and management of protocol information; (ii) a contextual ontology to explicitly represent the semantic of classified protocols in a meaningful way (i.e., with a common understanding) for heterogeneous agents; and (iii) a definition of a protocol composition process, based on ontology-driven rules, that makes it easy to update the system information by both evolving protocol data in a unique resource (an ontology) and by activating particular rules for acquiring different customized compositions of contextual protocols.

DynaCIP is not currently dealing with possible conflicts that exist among protocols from different contexts, leaving for the developers the responsibility for maintaining the coherence of systems' directives. However, contexts can be enhanced with a priority field for determining overriding in the event of conflicts. For instance, policies for resolution of conflicts can be protocols from lower levels (e.g., role protocols) that override others protocols (e.g., organization and environment protocols), as the solution pointed out in [11].

DynaCIP's main idea for continuously supporting agents with updated contextual information, based on both a do-

main ontology instance and compositions of rules, seems to be suitable for MASs from different domains. For the domain of normative MASs, instead of contextual protocol information, contextual norms are provided as precise inputs for enforcement mechanisms [5]. For the domain of pervasive mobile applications, policies are applied according to agents' contexts, which are determined based on agents' physical locations [20]. For the domain of next-generation wireless communications, contextual information permits to automatically change clients' benefits (e.g., price discounts) according to clients' locations (i.e. clients' contexts). The idea is to balance the use of network bandwidths by continuously distributing clients in particular networks. Clients are guided to not use overloaded networks by following given benefits. The result is that clients are dynamically distributed in regulated networks by only changing domain rules and data.

For future work, we are currently designing a framework for developing context-aware open MASs in which agents will be supported with updated information according to their contexts in specific domains.

References

[1] F. Bellifemine, A. Poggi, and G. Rimassa. JADE: a FIPA2000 compliant agent development environment. In *AGENTS '01: Proceedings of the fifth international conference on Autonomous agents*, pages 216–217, 2001.

[2] P. Bouquet, F. Giunchiglia, F. Harmelen, L. Serafini, and H. Stuckenschmidt. C-owl: Contextualizing ontologies. In *Proceedings of the ISWC-03*, pages 164–179, 2003.

[3] A. Dey. Understanding and using context. *Personal and Ubiquitous Computing*, 5(1):4–7, 2001.

[4] V. Dignum, J. Váquez-Salceda, and F. Dignum. Omni: Introducing social structure, norms and ontologies into agent organizations. *Programming Multi-Agent Systems: Second International Workshop (ProMAS 2004)*, pages 181–198, 2004.

[5] C. Felicíssimo, C. Lucena, J.-P. Briot, and R. Choren. Dynamic contextual regulations in open multi-agent systems. In *Proceedings of the ISWC-06*, 2006.

[6] A. Gómez-Pérez, M. Fernadéz-Peréz, and O. Corcho. *Ontological Engineering*. Springer Verlag, London, 2004.

[7] T. R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220, 1993.

[8] K. Henricksen and J. Indulska. Developing context-aware per-vasive computing applications: models and approach. *Pervasive and Mobile Computing*, 2005.

[9] C. Hewitt. Open information systems semantics for distributed artificial intelligence. *Artificial Intelligence*, 47(1–3):79–106, 1991.

[10] N. R. Jennings. On agent-based software engineering. *Artificial Intelligence*, 117(2):277–296, 2000.

[11] L. Kagal and T. Finin. Modeling conversation policies using permissions and obligations. *Journal of Agents and Multi-Agent Systems*, 14(4):187–206, 2007.

[12] L. Kagal, T. Finin, and A. Joshi. A policy language for a pervasive computing environment. In *Proceedings of the IEEE 4th International Workshop on Policies for Distributed Systems and Networks (POLICY 2003)*, pages 63–74, June 2003.

[13] M. Khedr and A. Karmouch. Acai: Agent-based context-aware infrastructure for spontaneous applications. *Journal of Network and Computer Applications*, 28(1):19–44, 1995.

[14] B. McBride. Jena: a semantic web toolkit. *IEEE Distributed Systems Online*, 6(6):55–59, Nov/Dec 2002.

[15] F. N. Nascimento, V. Sacramento, G. Baptista, H. K. Rubinsztejn, and M. Endler. Development and evaluation of a positining service based in ieee 802.11 (in Portuguese). In *Proceedings of the XXIV Brazilian Symposium on Computer Networks (SBRC 2006)*, 2006.

[16] N. Noy and A. Rector. Defining n-ary relations on the semantic web: Use with individuals, 2007. Link: http://www.w3.org/TR/swbp-n-aryRelations.

[17] H. K. Rubinsztejn, M. Endler, V. Sacramento, K. Gonçalves, and F. N. Nascimento. Support for context-aware collaboration. *First International Workshop on Mobility Aware Technologies and Applications (MATA 2004)*, 5(10):34–47, 2004.

[18] A. Toninelli, L. Kagal, J. M. Bradshaw, and R. Montanari. Rule-based and ontology-based policies: Toward a hybrid approach to control agents in pervasive environments. In *Proceedings of the Semantic Web and Policy Workshop (SWPW)*, November 2005.

[19] A. Uszok, J. M. Bradshaw, M. Johnson, R. Jeffers, A. Tate, J. Dalton, and S. Aitken. Kaos policy management for semantic web services. *IEEE Intelligent Systems*, pages 32–41, July/August 2004.

[20] J. Viterbo, C. Felicíssimo, J.-P. Briot, M. Endler, and C. Lucena. Applying regulation to ubiquitous computing environments. In *Proceedings of the 2nd Workshop on Software Engineering for Agent-oriented Systems (SEAS 06)*, pages 107–118, Outubro 2006.

[21] J. Viterbo, M. Malcher, and M. Endler. Supporting the development of context-aware agent-based systems for mobile networks. In *Proceedings of 23rd ACM Symposium on Applied Computing (SAC 2008), Mobile Agents and Systems (MAS) Track, Poster session (to appear)*, 2008.