Policy Infrastructure as an Extension to the FIPA Abstract Architecture for Open Agent Platform Design[1]

Javier Soriano, Genoveva López, Fernando Alonso

Technical University of Madrid, Campus de Montegancedo S/N,
28660 Boadilla del Monte, Madrid, Spain. Fax +34 913367393
{jsoriano, glopez, falonso}@fi.upm.es

Abstract. A growing interest in the practical application of the autonomous agent paradigm in open and distributed environments has been observed in recent years. A host of standards are being developed in this respect, aiming at easing and promoting interoperability between heterogeneous agents and multi-agent systems. The work of the FIPA is prominent in this field. The effort of this organization centres on defining an abstract architecture that identifies all the elements involved in achieving the above-mentioned interoperability and establishes the relationships between these elements. Nevertheless, this model does not completely address the inherent social-ability of any multi-agent system and which is especially important for achieving a higher level of interoperability. This paper lays the foundations for an extended role model and an organizational model that will serve as a framework for establishing and specifying relationships between heterogeneous agents interacting in an open environment. The agent and agent information models proposed by the FIPA are adapted and extended to accommodate this social viewpoint and social requirements on agent conversations and communication languages are discussed. We then propose a policy framework that constitutes the basis for agent social infrastructure management. Finally, the most relevant platform service infrastructure needed to support the policy framework is presented and defined. Our aim is to be able to model how agents are going to interoperate with one another within an open environment by means of normative policies, and how this environment will be organized and supported.

1 Introduction

The already widely used autonomous agent paradigm is attracting even more interest nowadays, as it promises to be a suitable mechanism for modelling complex problems arising in heterogeneous and distributed environments. Agent autonomy is a key modelling element in these environments.

Despite their intrinsic autonomy, agents tend not to be self-sufficient, which means that generally they will not be able to act exclusively on the basis of their needs and individual capabilities, but will have to relate to and cooperate with the other agents in their environment to achieve their goals. This, together with the fact that multi-agent

[1] This work is being supported by the Spanish Ministry of Science and Technology, Project. TIC2001-3451

environments tend to be open and distributed, leads to the need to develop standards aimed at easing and promoting agent and heterogeneous agent systems interoperability.

The work of the FIPA is prominent in this field. Its specifications constitute an interoperability model covering everything from the agent infrastructure to the application domains. This model focuses on the communicativeness of agents and assumes the interoperability among agents that communicate with each other using a common set of semantics. This approach can be used to abstract the internal architecture of each particular agent, thus providing the aforesaid interoperability.

Agent systems interoperability, then, is based on the use of a common agent communication language (ACL) [1], characterized by a set of potential illocutions, called performatives. The assumption is that there is an agreement among agents to use (i) a language of specific content to contextualize the performatives, (ii) ontologies to reach agreement on the content of the messages expressed in the above language and (iii) a pre-established interaction protocol that directs the dialogue among the agents involved. All these elements provide for effective knowledge exchange among agents in heterogeneous environments [2].The abstract FIPA architecture [3] provides mechanisms that can be used to enact the communication process among heterogeneous agent systems with a view to achieving interoperability in message representation and transportation. However, the technological support provided by FIPA alone is not sufficient to achieve interoperability, agents also need to have the knowledge to contextualize their acts of communication within the multi-agent environment in which they take place (i.e. the agent society in which they participate). This knowledge can be formalized by a social model by means of which an agent can recognize, identify and relate to the other agents in the society of which it is a member, as well as make the decision to communicate with other agents and enact this communication in a heterogeneous environment.

This paper presents an extended role model and an organizational model that lay the foundations for a social conception of multi-agent systems and will serve as a framework for establishing and specifying relationships between heterogeneous agents interacting in an open environment. The agent and agent information models proposed by the FIPA are adapted and extended in sections 2 and 3 to accommodate this social viewpoint and, social requirements on agent conversations and communication languages are also discussed. We then propose a policy framework in section 4 that constitutes the basis for agent society management. Finally, the most important platform service infrastructure needed to support the policy framework is presented and defined in section 5.

2 The Extended Role Model

The social metaphor and the role-based model are coming to be accepted as a natural means of conceiving multi-agent systems in the agent paradigm. Indeed, this metaphor has matured tremendously since it was first proposed, and was firmly established in [4]. In this paper, organizations are conceived as work frames that accommodate a topology based on the concept of role [5], which has been a revulsive

for the adoption of this metaphor. The reason is that such a model can be used to constrain the behaviour of the agents in an organization through a series of rules, from which the structures existing in the organization can be deduced and some behaviour patterns can even be extracted or exploited.

Our contribution takes a step further in this same direction, as it proposes a plausible way of characterising an agent society by identifying the social relationships between its actors and, particularly, between the different identified roles. This identification is specified in an extended role model and an organizational model.

In our extended role model, social relationships provide a conceptual framework in which each agent plays its role or occupies a well-defined position within the society. Roles motivate and structure all interactions between and assign abilities to agents. Fig. 1 shows the proposed role model.

As shown in Fig. 1, a role determines the following competence areas for the agent that assumes the role in question:

- *Skills*. Skills represent the abilities required for an agent to assume the role. Each agent is characterized by a set of abilities that represent the functionality or services provided by the agent. The skill constitutes the abstract representation of a low-level service associated with one or more specific agents. Skills represent reusability and encapsulation at a fully detailed level of role behaviour. Every role contains a number of specific skills, and one skill may belong to more than one role. An important skill-related issue is to define certain skills as a prerequisite for others. The concepts of role and skill should not be confused: a skill does not assume a given position in the society, whereas a role does.
- *Responsibilities*. The role assumed by an agent determines all the responsibilities assigned to the agent within the society. A distinction is made between individual and social responsibilities, although we are only concerned with the social responsibilities. Responsibilities represent the rights, duties and activities to be assumed by the agent that acquires the role.

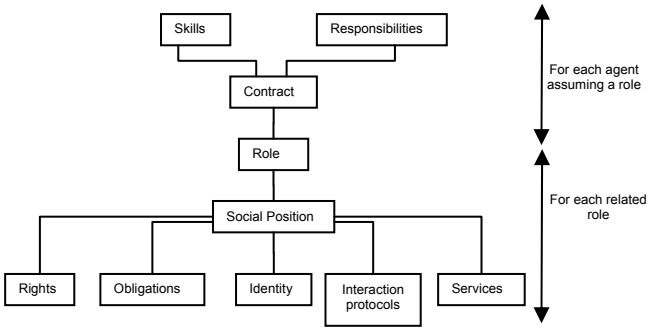


Fig. 1. Components of the Extended Role

Fig. 1 also shows that a role determines the following competence areas from the viewpoint of each related role.

- *Identity*. Each interaction model will be described much more easily if the interaction scheme is identified among roles, rather than among specific agents. This is an extension of the FIPA Agent Model
- *Services*. Identification of services at a role level also makes the service location scheme more flexible and, thus, enables dynamic changes in the functionality and location of services by delegating all roles related to these services to new agents. This is an extension of the FIPA Agent Information Model.
- *Patterns and protocols of interaction*. These are used to allow the agent to contact other agents and/or roles, get the information it needs from such agents and/or roles and provide agents with inferred information, as well as to coordinate agent activities. They are, therefore, based on role specification and information exchange. The interaction protocol is related to the role activity and is a part of the specification of a relationship between two or more roles.
- *Rights and duties*. These concepts are used to model the authorizations and obligations associated with the position represented by the role.
- *Information/knowledge shared* (requested and provided) for the agent assuming the role to fulfil its responsibilities.

So, not only does the role define and constrain the areas of competence of each agent within the organization, it also positions the agent within the organization with respect to its potential relationships with other agents.

According to the extended role model, when an agent enters into a contract in order to acquire a role, its position and behaviour is constrained according to the skills and responsibilities agreed in the contract. This means that the agent's behaviour is automatically aligned with that of the organization, so that the organization can, convergently, achieve its social goals. So, a new type of behaviour, termed socially responsible, is added to the classical taxonomy (benevolent, individualistic, social-cooperative). Additionally, this behaviour is not established during agent design, but is defined dynamically, depending on the role requirements in the organization and restricted to the lifetime of the contract.

3 The Organizational Model

As stated above, the general lack of agent self-sufficiency means that they have to relate to other agents and agent organizations. They are, therefore, thought of as members of a society whose model must be taken into consideration. As shown in Fig. 2, this model defines an agent society whose key concepts are the relationships between roles, the social laws that govern these relationships and the social structure that arises as a result. As the society is open, these relationships and social laws are dynamic. Therefore, the social structure will be determined by the roles, the relationships identified between these roles and the social laws that govern these relationships.

Each agent in a society assumes full responsibility for autonomously carrying out its individual tasks. This is not the case with its social responsibilities, however. The need to ensure that these social responsibilities are fulfilled means that both the social laws required for such actions to be completed and the social laws governing such

relationships have to be identified. These laws have to be respected and/or enforced by the society, through agency mechanisms and services, if it is to work properly according to the generally expected behaviour. It is these rules that govern social relationships and ensure a socially responsible behaviour in individualistically designed agents. The idea of ensuring socially responsible behaviour during agent design is evidently not appropriate in a dynamic, heterogeneous environment like a society, where, just to give an example, interaction patterns are established dynamically. Coordination and negotiation models [6], as well as the actual social structure, make up the conceptual abstractions for establishing the laws that will govern both agent performance and their mutual relationships with the aim of attaining socially responsible behaviour. The coordination and negotiation resources provided by the agency support such models and constitute the broker mechanism employed by the society to this end.

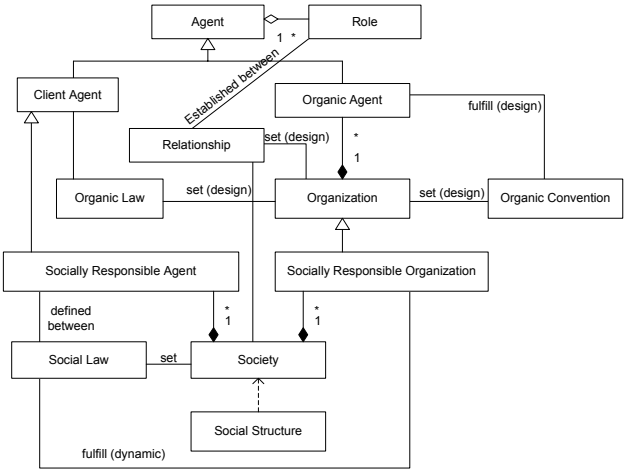


Fig. 2. Concepts in the Organizational Model

For a further description of this model, see [7].

4 A Policy Framework for Agent Society Management

Policies are a multidisciplinary research area intended to constrain the behaviour of software entities [8][9]. FIPA specifications include a brief policy-based approach to agent management [10]. It covers only the constraints on agent platform services utilization. Policies related to social organizations and relationships among agents in the society fall outside of its scope.

Our proposal is to use a policy framework to specify and automatically generate infrastructures for open agent society management. We consider that this is what is needed to assure the expected social behaviour and as the foundation for supporting the role and organizational models described above. For example, consider classes of

policies that express requirements concerning the use of a particular conversation policy when interacting with a particular role.

We are concerned with policies that provide statements of objectives that are intended to govern agent behaviour within an agent society. Policies are intended to capture constraints and across behaviour defined (possibly nested) scopes, such as "all agents requesting a service from agent x must have role y".

Indeed, we are primarily interested in policies that will make it possible to formally define roles, specify the requirements established in the agent-role contracts, express relationships between agents and, therefore, formally specify the organizational structure and dynamics of agent societies.

4.1 Contract Commitment

The concept of *contract commitment* is associated with policies and the mechanisms required to support policy application. A contract commitment is used both as an agreement entered into by an agent when assuming a role and as an agreement entered into by roles (i.e. agents) participating in a relationship, to be constrained by one or more sets of policy constraints.

For an agent to "honour" a request against a contract commitment, an inference procedure must be used to verify that the applicability requirements of the request can be satisfied in line with the policies in force. It will require a kind of policy service that should be called upon to (1) validate something against a specified policy, (2) determine what policy applies to a given condition and (3) determine whether there is a conflict between applicable policies.

4.2 Policy Schemas and Domains

Role-based models are insufficient for identifying agent groups covered by a given policy or between which there is a normative relationship, as the same policy or normative relationship tends to affect more than one role. Therefore, we propose extending the role-based model to a domain-based model. The domain concept extends the role concept and can define the organisational structure of the agent society on the basis of roles. Accordingly, a domain can identify both functional and structural agent groups and, consequently, express concepts as important as (1) job, which can involve one or more roles between which there is a given functional relationship, (2) the organisational unit in which these jobs occur and (3) the organisation itself.

The benefits of a domain-based approach include (1) the possibility of policy application to a domain being propagated to all its subdomains with a view to improved scalability and (2) not having to change the set of policies when an agent migrates from one domain to another.

A domain represents a set of agents characterized by a common policy schema. Policy domains are aids for efficiently applying policies. The infrastructure needed to support policy domains includes constructs such as domain controllers. Policy schemas and policy domains assure agent uniformity and interoperability in policy-

governed interactions across multiple heterogeneous platforms, as long as semantically equivalent repository, monitoring and enforcement mechanisms are available across those platforms. Policies can be structured in independent domains, related domains and hierarchically nested sub-domains.

For the purpose of combining domains and expressing the set of roles to which a policy applies, we introduce a declarative language ℓ_{dom} to build domain expressions (any domain expression is also a domain). This language should be semantically rich enough to be able to express the fact that a sub-domain is not necessarily a subset of the parent domain, which means that the agents within this sub-domain could be indirect, not necessarily direct, members of the parent domain. This is a key distinction for determining the scope of a given policy (a single sub-domain or all sub-domains, etc.). The definition of the language is:

$$\ell_{dom} = r \in \Re \mid d^{\circ} \mid d^{\bullet} \mid d^{n} \mid \delta_1 + \delta_2 \mid \delta_1 \oplus \delta_2 \mid \delta_1 \times \delta_2 \mid \delta_1 \otimes \delta_2 \mid \delta_1 - \delta_2 \quad \textbf{(1)}$$

where *r* denotes a role, d° represents the set of all the members of a domain that are not themselves domains (domain entries representing subdomains), whereas $d^{\bullet}$ also includes the subdomain members recursively nested in *d* and d^{n} delimits nesting to depth n. $\delta_1 + \delta_2$ is the set that contains all the different members of the union of the sets resulting from evaluating the domain expressions δ_1 and δ_2. Similarly, $\delta_1 \times \delta_2$ represents the intersection, $\delta_1 \oplus \delta_2$ represents the disjoint union ($\delta_1 \oplus \delta_2 \equiv (\delta_1 + \delta_2) - (\delta_1 \times \delta_2)$), and $\delta_1 - \delta_2$ represents the difference ($\delta_1 - \delta_2 \equiv \delta_1 - (\delta_1 \times \delta_2)$).

4.3 Policy Parameterization and Type Specialization

Reuse is provided by supporting parameterization in policy definition, to which any policy type can be passed as a formal parameter. Multiple instances can then be created and tailored for the specific environment by passing the actual parameters.

Type hierarchies allow policy type specialization, through the mechanism of inheritance and polymorphism. In this respect, it is important to note that policies can contain other policies in an inheritance model, which is crucial for efficient and scalable use.

4.4 Policy Constraints

An important element of each policy is the set of constraints that represents the conditions under which the policy is valid. This information must be explicitly stated in the specification of the policy. Constraints can be classified in two different ways:

With regard to their structure:

- *Basic policy constraints* limit the applicability of a basic policy and are expressed in terms of a predicate. Different types of constraints are based on subject/target state, action/event parameters in obligations, action parameter values in authorizations or refrains, or time constraints.

- *High-order policy constraints* constitute another category of constraints. A high-order constraint is parameterized by other constraints. This is a form of dependency among constraints; however, it differs from normal conjunction (which is implied by policy inheritance, for example) in that a high-order policy refers explicitly to a policy variable.

With regard to the kind of policy they specify:

- *Operational constraints* specify policies about agent behaviour that should not be violated. Constraints on the knowledge to be shared and constraints on the interaction protocol structure are examples of operational constraints.
- *Structural constraints* specify policies about which relationships should be authorized or obligatory among roles (i.e. among agents assuming those roles).

The agents involved in a specific relationship can behave differently (benevolently, individualistically or social-cooperatively), which leads to different kinds of organizational structures (control-driven, peer-to-peer, benevolent, dependence or property relationships). Operational and structural constraints can be used to restrict agent behaviour and organization within the agent society.

4.5 Policies for Specifying the Structure and Dynamics of Agent Societies

The purpose of our extended role and organizational models described above was to specify the structure and dynamics of an agent society. The concepts in those models, such as right, duty, relationship, interaction protocol, etc., can be further modelled and managed by means of the proposed policy framework. For this purpose, our policy framework provides for a number of, often related, policy types. The main ones are related to permissions, obligations, organizational structure issues, and conversation management. Permission and obligation policies are basic policy types, whereas organizational and conversational policies are sophisticated policy types constructed from the basic policies and recursively from themselves.

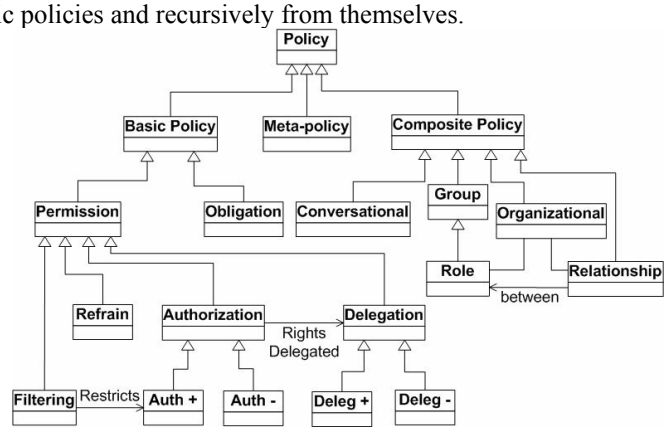


Fig. 3. Policy Classification

Fig. 3 shows a more detailed classification of these policy types, including the concept of meta-policies.

4.5.1 Basic Policies

Permission policies and obligation policies are basic policies. *Permission policies* represent constraints on service and resource accessibility based on agent credentials and requirements. The concept of right in our role model can be modelled as a permission policy. Permission policies can be further classified as:

- *Authorization policies* limiting the activity of legitimate agents who have been successfully authenticated. Authorization policies may be positive (permissive) or negative (prohibitive).
- *Filtering policies* needed to transform the information input or output parameters into an action and to introduce the concept of 'agent views'. Filters are often related to positive authorization policies.
- *Delegation policies* often used to cater for the temporary transfer of access rights between agents. They specify the authority (right) to delegate. The ability of an agent to delegate access rights to another must be tightly controlled by security policies. This requirement is critical in agent systems permitting cascaded delegation of access rights. A delegation policy is always associated with an authorization policy, which specifies which access rights can be delegated.
- *Refrain policies* are subjective, self-decided authorizations. They are similar to negative authorization policies, but are enforced by subjects rather than target access controllers.

Obligation policies are used by socially responsible agents when entering into agreements that commit them to a particular future behaviour. If the autonomy of an organic agent is to be preserved, situations in which the obligatory normative constraints can be violated should be accounted for. The fulfilment of an obligation is then subject to a deliberative decision-making process driven by a weighting of the consequences derived from violating this obligation, that is, the associated cost of the penalty incurred by breaching the obligation. On the other hand, the interpretation of authorisations (positive and negative) is not subject to any deliberative decision-making process, as they can never be violated. Obligations can be enforced following a priori or sanction-based models. In sanction-based models, obligation policies are used to design reputation services that provide a means for agents to complain about the failure of other agents to meet obligations and for agents to verify the reliability of other agents before entering into agreements. Obligation policies provide support for negotiation in agent coordination schemas. The concepts of agent responsibility and role duty can be modelled as obligation policies.

4.5.2 Composite Policies

Basic policies are insufficient for fully specifying relationships between agents in an organization, and they are also insufficient for specifying agent conversations in terms of message content and permitted message sequences (e.g. for contract negotiation). Organizational policies and conversation management policies provide support for these tasks.

Composite policies represent clauses increasing requirements, directly conjunctive constraint clauses, high-level clauses changing the context of the policy and low-level clauses modifying an individual constraint by adding a conditional factor. They provide the ability to group and structure policies to reflect organizational structure, preserve the natural modus operandi of the system administrator or simply provide

reusability of common definitions. This makes policy management and enforcement in large, complex societies of agents easier.

We propose policy groups, subject groups and relationships as mechanisms for policy composition. Organizational policies and conversation management policies use policy composition mechanisms to become composite policies.

Policy groups are a packaging construct for grouping related policies together for the purposes of policy organization and reusability. The criteria for grouping policies together may reflect social organization, as we will see with social management policies. They can contain zero or more basic policies, nested groups and/or meta-policies. A meta-policy specifies constraints on the policies within the scope of the group. Reusability can be achieved by specifying groups as types, parameterized with any policy element.

Subject groups provide a semantic grouping of policies with a common role as the subject, generally pertaining to a position within an agent society. So a subject group is a special case of a group, in which all the policies have the same subject (i.e. agent role). The criteria for grouping policies together as roles may reflect the agent role model.

Relationships. Agents acting in organizational positions (roles) interact with each other. A relationship groups the policies defining the rights and duties of one participating agent as regards another. It can also include policies related to resources that are shared by the roles within the relationship. It thus provides an abstraction for defining policies that do not pertain to the actual roles (so they are not subject group policies), but are part of the interaction between those roles. A relationship descriptor includes definitions of the participating roles. Relationships can have nested role definitions. Participating roles can also be defined as parameters within a relationship type definition.

Organizational policies model social management structures. Agent societies can be structured as organizations that have a similar configuration of roles and policies. Social management structures define a configuration in terms of instances of roles, relationships and nested management structures relating to organizational units. A social management structure is thus a composite policy containing the definition of subject groups for roles, relationships and other nested management structures, as well as instances of these composite policies. For example, instances can be created for branches within the organization that exhibit the same role-relationship requirements. These social grouping policies represent a set of artificial constraints that articulate agent organizational relationships. They represent constraints on the permissible communication, visibility, etc., between agents based on their position in a social structure or organization. Thus, social grouping policies capture relationships between roles, such as incompatibilities, hierarchy, etc., which leads to different kinds of organizational structures (control-driven, peer-to-peer, benevolent, dependence or property relationships).

Conversation management policies extend the concept of interaction protocols. Agent communication is better modelled when conversation policies, rather than isolated messages or fixed interaction protocols, are taken as the primary unit of analysis. Conversation policies are declarative specifications that govern and constrain specific instances of communications between agents using an ACL.

4.5.3 Meta-Policies

The last category of policies is what we call meta-policies. Meta-policies specify policies about the policies within a composite policy or some other scope. They govern the nature of other policies. Meta-policies can be specified for policy groups to express constraints that limit the policies permitted in the system or prohibit the simultaneous execution of conflicting policies, e.g., "there should be no policy authorising a manager to retract policies for which he is the subject".

4.6 The Need for a Policy Definition Language

Policies can be expressed in a variety of languages [11][12]. We propose a declarative and ontology-based approach. Declarative and ontology-based policy definition languages enable consistency enforcement and inference processes (e.g. policies related to an agent role and agent roles related to a policy), apart from other properties such as expressivity, composability, computability and equivalency.

The basic building block of a policy is a policy rule, which is a simple declarative statement associating a policy object with a value. For example, a policy rule can define a destination, such as destination=role1. Policy rules define either conditions or actions. Each policy, then, includes one or more conditions and one or more actions. The conditions define when the policy rule is applicable. If the conditions in a policy are not met, then the actions comprising that policy are not enacted. Standard predicate logic can be used to describe policy rules. Logically, a policy rule takes the form of a conjunction of implications.

In addition to the use of standard predicate logic to specify policy rules, we envisage a policy definition language having built-in ontologies [13] for sharing and relating concepts such as action, permission and obligation. Omitting the relationship between permission and obligation will lead to unforeseeable conflicts among policies. Ontological axiomatization is an aid for formally defining and relating these concepts. In this respect, for example, the axiom $Pp = \neg O \neg p$ expresses that "p is permitted is equivalent to p is not under obligation", which will be accepted as a suitable definition of permission [14]. Policies described based on the logic of permissions and obligations call for a type of modal logic called *deontic logic*. Ontologies can also help in policy inference and conflict resolution.

At the higher levels in policy-based agent society management, policies may be represented at various levels of abstraction. Agents will have to make decisions based on the policies related to them.

5 Platform Infrastructure Needed to Support the Policy Framework

The following sections describe the platform service infrastructure needed to support the policy framework presented in the previous section.

5.1 Domain Controllers

An agent domain consists of a unique instance of a domain controller along with any agents that are registered to it. The function of a domain controller is to serve as a single point of administration for policy management, i.e., configure, modify, store, publish and enforce where possible the set of policies declared for that domain.

A major purpose of domain controllers is to ensure consistency of policy across a set of agents potentially running on different agent platforms and hosts. This can be accomplished as long as semantically equivalent monitoring and enforcement mechanisms are available across those platforms and hosts.

It is possible to define domains that explicitly require registration, as well as domains that require no registration, or subordinate registration to other elements of the FIPA environment, such as physical agent platforms.

5.2 Directory Service

Directories provide services for agent register directory-entries. Other agents can search the directory-entries to find agents with which they wish to interact. In other words, it provides services for registration, lookup, discovery, authentication, etc.

Directory service provides the basis for policy storage and distribution.

5.3 Reputation Service

Prohibitions and obligations require different kinds of enforcement mechanisms. As mentioned before, the former are to be supported by the target and can be enforced with help from domain controllers, but the latter are to be supported by the subject and requires a kind of reputation service because they cannot be prevented but only required.

A reputation service allows agents and services to monitor the public performance of agents and services in terms of their compliance to publicly entered-into policy agreements. A reputation service takes the role of a trusted third party that agents and service providers may use to monitor compliance with agreements.

A typical use of a reputation service is for all parties to an agreement to publish their agreement with the reputation service. If one of the parties determines that another party has defaulted on an obligation it may lodge a complaint with the reputation service. If an agent defaults on an obligation, other agents and services may become more reluctant to offer it facilities if they are able to query a reputation service.

6 Conclusions

The growing interest in the use of the autonomous agent paradigm for modelling complex problems arising in open and distributed environments raises a pressing need to develop standards aimed at easing and promoting interoperability among agents

and heterogeneous agent systems. Although the work of the FIPA is prominent in this respect, its proposals do not consider the inherent sociability of this paradigm to its full extent, despite the fact this is especially important for achieving higher levels of interoperability among heterogeneous agents.

On the basis of the underlying social metaphor of the autonomous agents paradigm, this paper has proposed an extended role model and organizational model as aids for modelling the authority, responsibility (rights and duties), services, relationships and interactions associated with the different positions within an agent society. It has also proposed a policy-based framework to be used to specify the concepts within the role and organizational models in terms of authorization/obligation, as well as control the structure and dynamics of agent societies. Finally, the most important platform service infrastructure needed to support the policy framework has been presented and defined.

We believe that this approach will make it easier to specify and generate the mechanisms required to assure the correct operation of heterogeneous agent societies, such as dynamic role and domain assignment to agents, directory services, domain controllers, reputation services, etc.

7 References

[1] R. Neches et al. "Enabling Technology for Knowledge Sharing". AI Magazine 12, 3, 1991, pp 36-56.

[2] M. R. Genesereth and S.P. Ketchpel, "Software Agents", Communication of the ACM, Vol. 37, No. 7, July 1994.

[3] Foundation for Intelligent Physical Agents, "FIPA Abstract Architecture Specification", Geneva, Switzerland, 2000. http://www.fipa.org/specs/.

[4] F. Zambonelli, N. R. Jennings and M. Wooldridge. "Organizational abstractions for the Analysis and Design of Multi-Agent Systems". In Agent Oriented Software Engineering, pp 185-193, 2000.

[5] B. J. Biddle and E. J. Thomas, "Role Theory: Concepts and Research", Robert E. Krieger Publishing Company, New York, 1979.

[6] P. Ciancarini, "Coordination models and languages as software integrators", ACM Computing Surveys, 28(2), June 1996.

[7] F. Alonso, F. Fernández, G. López, J. P. Rojas and F. J. Soriano, "Intelligent Virtual Agent Societies on the Internet", in Springer Lecture Notes in Computer Science and Lecture Notes in Artificial Intelligence Series (LNCS/LNAI), LNCS 2190. 2001.

[8] M. Sloman, "Policy Driven Management for Distributed Systems", Journal of Network and Systems Management, Plenum Press, Vol. 2, Nº 4, 1994.

[9] K. Ong and R. Lee, "A logic Model for Maintaining Consistency of Bureaucratic Policies", in proceedings of the 26th Annual Hawaii Conference on System Sciences, Vol. III, pp. 503-512, 1993.

[10] Foundation for Intelligent Physical Agents. FIPA Policies and Domains Specification. 2001. Geneva, Switzerland. http://www.fipa.org/specs/.

[11] J. Lobo, R. Bhatia and S. Naqvi, "A Policy Description Language", in proceedings of AAAI, Orlando, Florida, USA, July 1999.

[12] Jajodia, S., Samarati, P., and Subrahmanian V.S. A Logical Language for Expressing Authorisations. In Proc. of IEEE Symposium on Security and Privacy. 1997.

[13] T. R. Gruber, “Toward Principles for the Design of Ontologies Used for Knowledge Sharing”. Formal Ontology in Conceptual Analysis and Knowledge Representation, edited by Nicola Guarino and Roberto Poli, Kluwer Academic Publishers, 1993.

[14] R. Ortalo, “A Flexible Method for Information System Policy Specification”, in Proceedings of 5th European Symposium on Research in Computer Security (ESORICS <98), Louvain-la-Neuve, Belgium, Springer-Verlag, 1998.