A Metamodel for Agents, Roles, and Groups

James Odell[1], Marian Nodine[2], Renato Levy[3]

[1]Agentis Software, Inc., 3646 West Huron River Drive,
Ann Arbor, MI 48103-9489 USA
email@jamesodell.com
http://www.jamesodell.com

[2] Telcordia Technologies, Inc., 106 East 6th Street, Suite 415,
Austin, TX 78733 USA
nodine@research.telcordia.com

[3] Intelligent Automation Inc., 7519 Standish Place, Suite 200
Maryland, MD 20855 USA
rlevy@i-a-i.com

Abstract. Societies need patterned behavior to exist. Large-scale agent societies may contain a diversity of agents, each with differing abilities and functionalities. When such an agent system is given a task, it must dynamically muster together a group of agents that collectively have the capability to accomplish the task. To do this, the agent society needs to be able to understand its agents and their potential interactions.

This paper contains a proposed superstructure specification that defines the user-level constructs required to model agents, their roles and their groups. These modeling constructs provide the basic foundational elements required in multi-agent systems to foster dynamic group formation and operation. As agent systems scale beyond the point where an individual organization can track and control their behavior, the use of these concepts within the society will facilitate dynamic, controlled, task-oriented group formation. This in turn will enhance the predictability, reliability and stability of the agent system as a whole, as well as facilitating the analysis of both group and system behavior.

1 Introduction

We simply have hardly any real experience building truly heterogeneous, realistically coordinated multi-agent systems that work together, and … almost no basis for systematic reflection and analysis of that experience [Gasser, 2001].

Societies need to employ patterned behavior to exist. The behavior of each individual is determined to a great extent by the requirements of these patterns [Katz, 1978]. However, the current practice of Multi-Agent System (MAS) design tends to be limited to individual agents and small face-to-face groups of agents that operate as closed systems. We have little principled understanding for:

- organizing sophisticated, interactive, heterogeneous agent-based systems.

- grouping the agents in such systems into very large-scale aggregates that exhibit predictable, stable, and reliable behavior.
- achieving economies of scale and scope within a large MAS.
- building and operating such systems in situ.

From a scientific standpoint, the foundations for constructing large multi-agent systems have a long history. Although researchers have been explicitly thinking about MAS/DAI (Multi-Agent System/Distributed Artificial Intelligence) organizations and attempting to link organization theory with MAS/DAI models for decades, the idea of organization, *per se*, has been only a peripheral theme. MAS/DAI researchers have focused on specific coordination techniques, rather than the central issues involved in MAS organization. Yet, without considering organizational issues for MAS, MAS designers will not be able to leverage benefits that can be gained from such social constructs and patterns, such as emergence and scalability. Therefore, any discussion of agent classification metamodels must also address organizational elements. At a minimum, this includes agent classifiers, roles and groups—and the structural and behavioral patterns defined by such constructs.

This paper contains a proposed superstructure specification[1] for modeling agents, agent roles, and agent groups. This architecture addresses simple homogeneous agent systems as well as those that require complex and heterogeneous social interaction. Furthermore, this specification is based on—and extends—the Unified Modeling Language (UML) superstructure [OMG, 2003]. It also contains a few suggested notations for some of these structures. Developing a notation to express these structures more fully is an on-going effort in several standards organizations including the Foundation for Intelligent Physical Agents (FIPA) and the Object Management Group's (OMG's) Agent SIG.

2 The Essential Class Model

Figure 1 illustrates the essential class model being proposed in this document. UML defines *class diagram* as follows: "A diagram that shows a collection of declarative (static) model elements such as classes, types, and their contents and relationships" [Booch, 1999]. Due to the relative complexity of agent design and the many differences that exist between an agent and an object [Odell, 2002], the agent class structure must extend UML class specifications to accomplish this. The complete notation for the extensions described in this paper may be found in [FIPA, 2004].

Starting from the left of Fig. 1, the Agent Classifier is a classifier specifically for classes of agents. An *Agent Classifier* defines the various ways in which agents will be classified and is subclassed in two ways: *Agent Physical Classifier* and *Agent Role Classifier*. *Agent Physical Classifier* defines the primitive, or basic, classes that define the core requirements of an agent. In general, an agent is implemented using some specific physical platform, such as JADE [Bellifemine, 2001] or Cybele [Cybele, 2000][Aronson, 2003]. The physical agent platform itself engenders certain

[1] A *modeling superstructure* specifies user-level constructs for modeling. In general, a *modeling infrastructure* provides the foundational constructs for the modeling language.

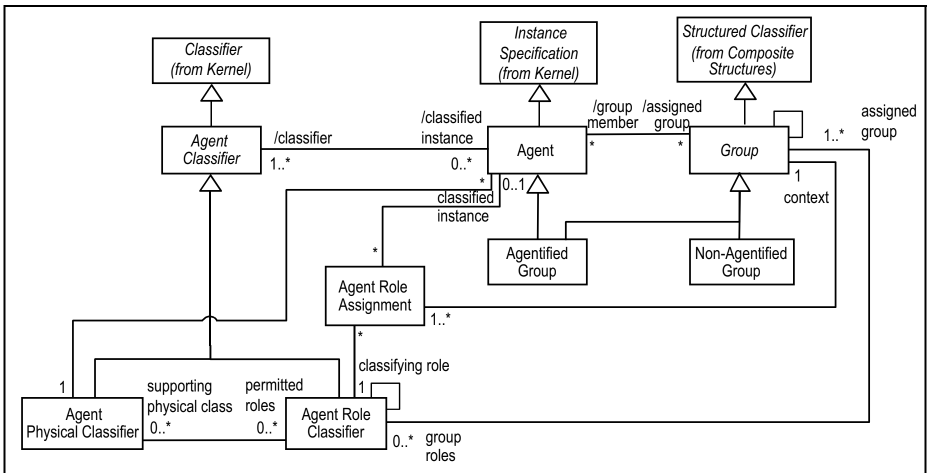


Figure 1. Proposed abstract syntax.

properties and capabilities on its agents, regardless of any other agents that they are collaborating with. These properties remain throughout the lifetime of the agent. *Agent Role Classifier* classifies agents by the various kinds of roles agents may "play". These in turn relate to the agent's capabilities as well as the activities in which it may become involved. The set of roles that an agent is playing may vary over time. Section 3 provides a more detailed explanation of the agent classifiers.

In the middle of Fig. 1 are the Agent instances. The class called *Agent* defines the set of all agents that populate a system. Each instance of an Agent is associated with one or more Agent Classifiers that define its necessary features. Agents are discussed further in Section 3.

To the right in Fig. 1 are the *Groups*, or sets of agents that have been collected together for some reason. Within a group, its member agents interact according to the roles that they play. Thus, each instance of a Group is defined by a set of roles and, by transitivity, its collection of agents. Groups are partitioned into Agentified Groups and Non-Agentified Groups according to whether or not they are addressable as an agent and can act as an agent in their own right. Groups are discussed further in Section 4.

Lastly we consider the associations between Agents and their Agent Role Classifiers and Agent Physical Classifiers (Section 5). The association of an Agent to an Agent Physical Classifier gives the agent its primitive, core capabilities and requirements. Every Agent must be associated with one Agent Physical Classifier. The association between an Agent and the Agent Role Classifiers determine what sorts of activities that the Agent is participating in. An Agent may have no associated Agent Role Classifiers; however, such an agent is not involved in any *Group*.

Key to understanding this figure is a crisp notion of the distinction between a UML Classifier and a UML Class. The two UML notions *Classifier* and *Class* are different elements in UML. A UML Classifier is:

- a *Namespace* whose members can include features.
- a *Type*, thereby making it possible to define generalization relationships to other Agent Classifiers.

- a *RedefinableElement*, meaning that it can be redefined more specifically or differently in the context of another classifier that specializes (directly or indirectly) the context classifier.

UML Classifiers do not have attributes, interfaces, inheritance, or any of the basic features that are associated with an object-oriented class. In contrast, the UML class called *Class* has these features. Class is a specialization of Classifier and possesses those additional features that are required for objects. (For more information on the differences between Classifier and Class, see [OMG, 2003]). This is important because agent classification will be based on an extension of Classifier, not Class. The reason for this is that we do not wish to develop a superstructure that is based on object orientation. To do so would mean that agents would necessarily support object-oriented-based messaging and polymorphism. Instead, by extending Classifier, we can add in those features that object-oriented classes possess that are useful for agents (e.g., attributes), while omitting features that are problematic for agents.

3 Agents and Agent Classifiers

Agent Classifier is a UML Classifier that specifically provides a way to classify *Agent* instances by a set of features that they have in common. Its associated instances are the Agents that it classifies. Classification is important because it enables the definition of a set of entities that share one or more capabilities and/or features in common. For example, Agent Classifiers named "Buyer" or "Seller" could be defined to represent the collection of agents that possess capabilities for buying and selling resources. Furthermore, Agent Classifiers facilitate the definition of those features that entities of a particular classification can have. For example, the "Buyer" Agent Classifier could define properties relating to what a Buyer can buy, and what its spending limit is.

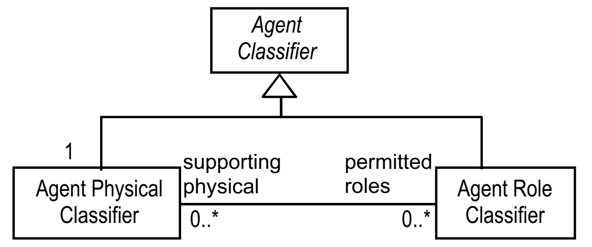


Figure 2. Abstract syntax for Agent Classifier, Agent Physical Classifier and Agent Role Classifier.

Figure 2 shows Agent Classifier and its two specializations: Agent Physical Classifier and Agent Role Classifier. The default notation for an Agent Classifier is a solid-outline rectangle containing the classifier's name. Abstract Agent Classifiers are shown in italics.

3.1 Agent Role Classifier

The Agent Role Classifier is an Agent Classifier that classifies according to the kinds of *roles* the agent is capable of playing at a given time. Within a MAS, roles define normative repertoires of behavior and other features, contextualized according to the group in which the role is being played. Agents can be associated with more than one Agent Role Classifier at the same point in time (multiple classification) and can change roles over time (dynamic classification).

The notion of *role* is fundamentally a thespian concept, and attention to how it functions in the theater can reinforce our intuitions and provide useful metaphors for application to multi-agent systems.

> All the world's a stage,
> And all the men and women merely players:
> They all have their exits and entrances;
> And one man in his time plays many parts.
> –W. Shakespeare, *As You Like It*, Act II, Scene 7.

The similarities between the Shakespeare's characterization and our present-day usage of *role* in role theory [Biddle, 1966] and organization psychology [Katz, 1978] are noteworthy. The *role perspective* [Odell, 2003] consists of those factors presumed to be influential in governing human behavior. It assumes that performance results from social proscriptions and individual behavior and that the individual variations in performance are expressed within the framework created by these factors. An individual's behavior is shaped by the demands and roles of others, the individual's own understanding of appropriate behavior, and the individual's competence in the performance.

Roles provide both the building blocks for agent social systems and the requirements by which agents interact. Each agent is linked to other agents by the roles it plays by virtue of the application's functional requirements—which are based on the expectations that the application has of the agent. Also, all Agent Role Classifiers must be associated with one or more groups, because the role is qualified by and given meaning by the group context. For example, the role of President for the United States is different that the role of President for IBM. The group context, therefore, provide a namespace for terms such as "President" so that unique role features can be specified for each context.

Agent Role Classifiers form a generalization hierarchy. Figure 3 illustrates a small hierarchy of Agent Role Classifiers, where the Broker and Matchmaker roles are sub-classifiers of the Facilitator role. In the UML extension for agents, Agent Role Classifiers are indicated using the stereotype «agent role»; non-agent classes which represent object classes will not have a stereotype designation. Please note that using a generalization relationship between classifiers does *not* necessarily imply inheritance. Generalization specifies inclusion which

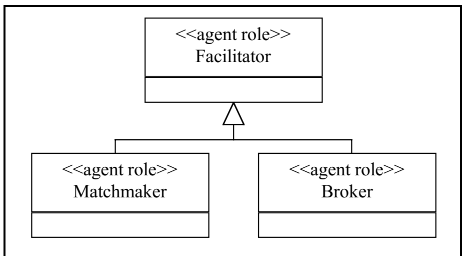


Figure 3: Example of a generalization hierarchy of Agent Role Classifiers**.**

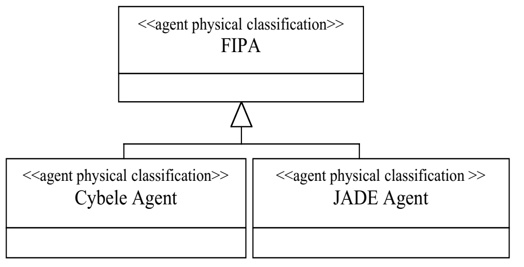
implies that whatever can be said of a classifier can also be said of its sub-classifiers. Generalization, therefore, can be implemented using various techniques—inheritance is only one. For agent-based systems, current wisdom suggests that using inheritance is not considered to be a good practice.

3.2 Agent Physical Classifier

The purpose of an Agent Physical Classifier is to define those sets of core, or primitive, features that all agents possess—*independent* of any role they may play. Every agent must be classified according to some Agent Physical Classifier. Also, an agent always will remain in the same basic Agent Physical Classifier that created it and bestowed its basic features.

An Agent Physical Classifier describes the set of basic (i.e., primitive, core) features that all agents possess. Certain features are basic to all agents, such as the possession of a unique name (from the namespace in the UML Classifier). Others are dependent on the implementation of the agent, such as how an agent sends and receives messages, maintains its attributes (i.e., its state, or beliefs), interacts with its particular environment or software package, and so on. In this way, Agent Physical Classifier could also be called "Agent Primitive Classifier" or "Agent Core Classifier" because its purpose is to define those classes that describe the set of basic features that all agents of a particular kind possess.

Figure 4 depicts several instances of Agent Physical Classifier (Cybele Agent Classifier, JADE Agent Classifier, and FIPA Agent Classifier) in a hierarchy. Any agents associated with the Cybele Agent classifier would possess all the features required and provided by the Cybele software environment (for example, the ability to possess state variables and to receive and send communications, create and refer to clocks and timers of various kinds).



<<agent physical classification>>
FIPA
<<agent physical classification>>
Cybele Agent
<<agent physical classification >>
JADE Agent

Figure 4: Example of a generalization hierarchy of Agent Role Classifiers**.**

As stated earlier, agents can be associated with more than one role at the same point in time and can change roles over time. However, an agent is quite likely considered to remain in the same basic Agent Physical Classifier that created it and bestowed its basic features. Therefore, the Agent Physical Classifier provides the required features for all agents, whereas, roles supplement these basic features by providing additional sets of features on an "as needed" basis.

3.3 Agents

For agent-based design, the primary fundamental modeling constructs, or *elements*, are Agent Classifier and Agent (Fig. 5). These elements are considered fundamental because they enable agent-based systems to define both the instances of agent for a system and the enabling classifications for those agents. The instances of Agent specify the autonomous, interactive entities known as an "agent" in a modeled agent-based system. That is, they are important because those are the *actual* functioning entities thought of as agents within a system; the classifications are necessary as they provide the underlying features for each and every agent. In other words, each instance of Agent Classifier specifies those features that its associated Agents may possess (such as state and provided services).

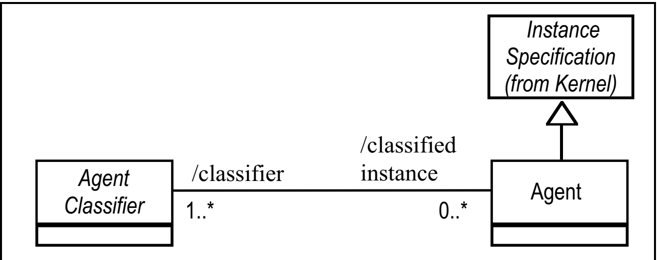


Figure 5: Association of Agents with Agent Classifiers.

Agent is a concrete class and its description may include:

- Classification of the agent by one or more Agent Classifiers of which the agent is a classified instance (multiple classification).
- Specification of features of the agent, independently of those specified by associated Agent Classifiers.
- Specification of how to compute, derive or construct the agent (optional).

An Agent instance specification describes the agent. These details can be incomplete. The purpose of an Agent instance specification is to show what is of interest about an agent in the modeled system. The agent conforms to the specification of each Agent Classifier that classifies it, and has features with values indicated by each slot of the Agent instance specification.

Figure 6 illustrates two linked Agent instances. Agent instances represent the agent at a specific point in time (a snapshot). Each Agent is depicted using the same notation as its classifier, but in place of the classifier name appears an underlined concatenation of the instance name (if any), a colon (':') and an optional comma-separated list containing the classifier name or names. Slots may be shown textually as a feature name followed by an equal sign ('=') and a value specification. Other properties of the feature, such as its type, also may be shown. Agents may be linked by the roles they play. In this situation, Agent4 is a Broker, and is brokering something for Seller Agent2.

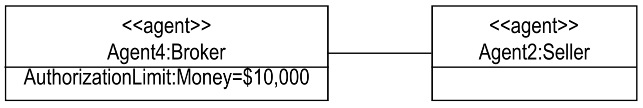


Figure 6. Agents with slots and associations

3.4 Associations between Agent Physical Classifiers and Agent Role Classifiers

The association between an Agent Physical Classifier and an Agent Role Classifier specifies those Agent Role Classifiers that are permitted for any given Agent Physical Classifier. Figure 7 depicts instances of Agent Physical Classifier ("JADE".

"Cybele") and Agent Role Classifier ("Manager", "Buyer", "Trust Manager"). For any of the roles of "Buyer", "Broker" and "Trust Manager" can be taken on by Agents with the Agent Physical Classifier "Cybele". Similarly, either of the roles "Manager" or "Broker" can be taken on by Agents with the Agent Physical Classifier "JADE".

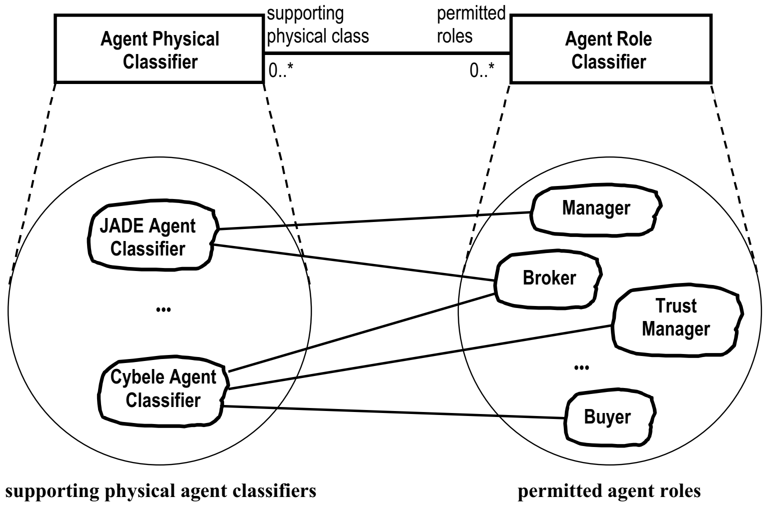


Figure 7. Example associations between Agent Physical Classifiers and Agent Role Classifiers.

The associations between Agent Physical Classifiers and Agent Role Classes restrict the roles that specific agents may take on, independent of the capabilities of the individual Agents themselves. In the following section, we discuss Agents and their associations with Agent Classifiers, based on both these restrictions and the capabilities of the individual Agents.

3.5 Association of Agents with Agent Classifiers

The association of an Agent with its Agent Classifiers establishes the features and behavior for each agent. Each Agent Classifier classifies of agent instances according to a common set of physical or role-based features that they have in common. Figure 8 depicts instances of Agent Physical Classifier ("Cybele" and "JADE"), instances of Agent Role Classifier "Buyer", "Seller", "Broker", "Trust Manager") as well as a set of instances of Agent ("Agent1", "Agent2", "Agent3", and "Agent4"). The links indicate classification; e.g., Agent1 is a classified instance of both the "JADE" Agent Physical Classifier and the "Seller" Agent Role Classifier.

Note that a given agent can be classified with more than one role at the same point in time (e.g., Agent2). Also, Agents can change roles over time, as the needs of the applications change.

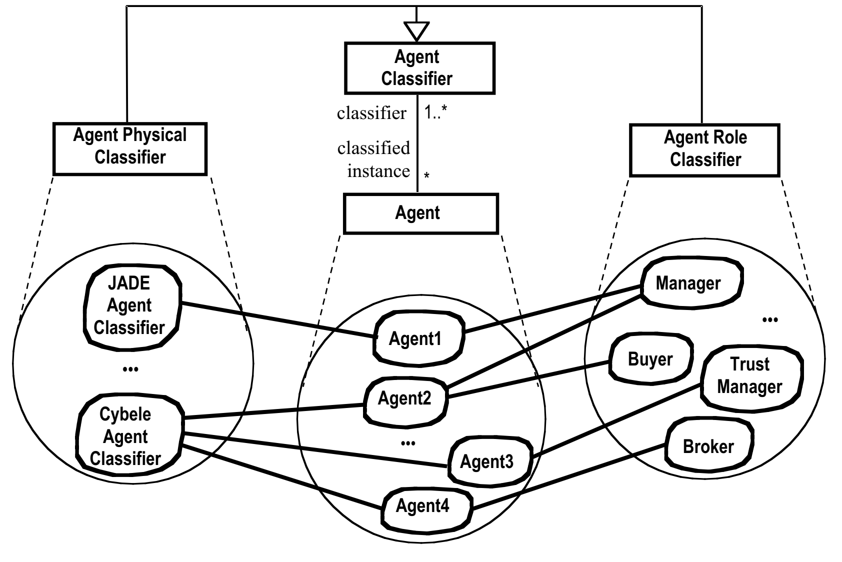


Figure 8. Examples of Agent Classifier and Agent instances and classification links.

4 Group, Agentified Group, and Non-Agentified Group

A *group* is a set of agents that are related via their roles, where these links must form a connected graph within the group. Another way to look at this is that a group is a composite structure consisting of interrelated roles, where each of the group's roles has any number of agent instances. This definition implies not only that a group is a function of the roles contained within it, but also that roles have no meaning without their group referent. Hence, our ability to understand roles is limited by our ability to understand the groups of which they are a part.

A group can be formed to take advantage of the synergies of its members, resulting in an entity that enables products and processes that are not possible from any single individual. As with roles, groups may be deliberately established (i.e., by a system designer) or they may be emergent. In human organization terms, a deliberately established group could be a department or other workgroup that has been defined by some organizational authority. In contrast, an emergent group might be a social group that forms when several individuals decide to go out for a beer after work. Over time, they define themselves as a group ("My Friday Afternoon Drinking Buddies").

Groups are commonly formed to regulate, foster, or support the interaction of those agents *within* the group; so the group provides a place for a limited number of agents to interact among themselves via roles. In this way, intra-group associations encourage resource sharing, promote internal coordination. establish common supervision, and provide a degree of safety in numbers.

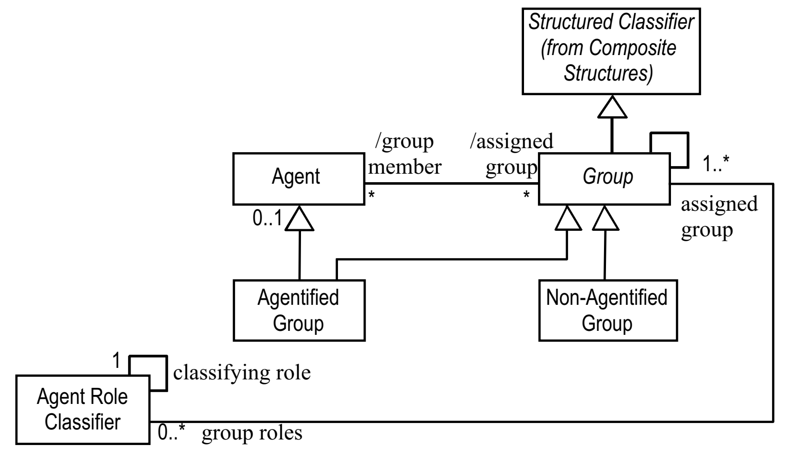


Figure 9. Abstract Syntax for Groups, their Agents, and their Roles.

4.1 Metamodel for Group

Figure 9 presents the abstract syntax for Groups, their Roles and their Agents. The Group class extends the UML Structured Classifier. Therefore each Group is defined as a composite structure. In UML, structured classifiers can be thought of structured collection of classifiers. Groups, then, are structured collections of Agent Role Classifiers. Group is an abstract class.

Conceptually, a group consists of a set of Agents playing roles. The roles that the Agents may play within the group are represented by one or more Agent Role Classifiers that are associated with the Group. The set of Agents within the Group, according to the model, can be derived from the Group via the Agent Role Classifiers (which will be discussed in Section 5).

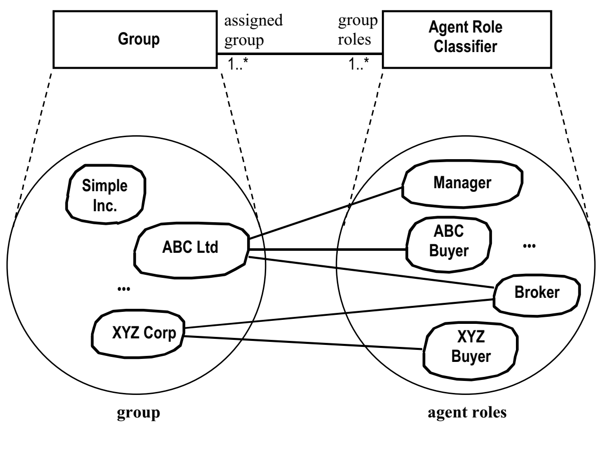


Figure 10. Example of Associations between Groups and Agent Role Classifiers

4.2 Relationships between Groups and the Roles that Agents play in them

Figure 10 illustrates the necessary association between groups and the roles that are played in groups (by Agents). Roles are only meaningful in a context; therefore, all roles must be assigned to a group. For example, the “Broker” role is used by the “ABC Ltd.” and “XYZ Corp” groups. Notice also that both groups have a “Buyer’ role. However, since the buyer role for “ABC Ltd.” has different features than for “XYZ Corp.”, two different roles are defined, “ABC Buyer” and “XYZ Buyer”.

4.3 Agentified and Non-Agentified Groups

A group can take on the qualities of being an agent in its own right, with its own interactive capability. Such groups can be thought of as sets of agents that interact with other agents or sets of agents. Inter-group associations are important and appropriate, because they encourage a basis for input and output standardization. This in turn facilitates interaction between groups, promotes patterns of interaction between groups, and establishes standard interaction points for each group.

An *Agentified Group* possesses all the features that any agent might possess. For example, it can send and receive messages directly and take on roles. Such a group is an agent in its own right, and therefore is a subclass not only of Group but also of Agent. In contrast, Non-Agentified Groups are still first-class entities; however, these entities do not possess agent properties. Thus, they are as objects, rather than agents.

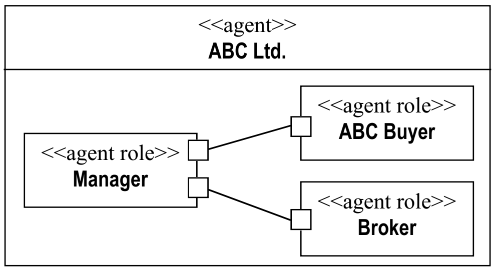


Figure 11. Example of the ABC Ltd. Agentified Group and its associated Roles.

Figure 11 represents the Group “ABC Ltd” as a composite structure with three associated roles, “Manager”, “Broker” and “ABC Buyer”. The “Manager” interacts directly with the “ABC Buyer” and the “Broker”. In this situation, it is possible to interact with the agent “ABC Ltd.” without knowing directly about any specific “Manager”, “Broker” or “ABC Buyer” within the department; thus, this group is Agentified. The stereotype “<<agent>>” indicates that the group is Agentified.

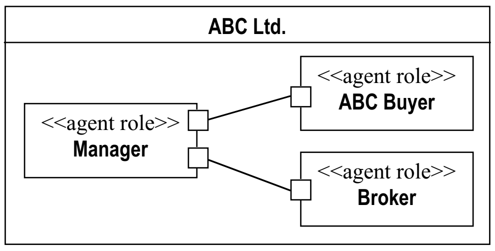


Figure 12. Example of the Non-Agentified ABC Ltd.

Groups can also be formed simply to establish a set of agents for purposes such as intra-group synergies or conceptual organization. A *Non-*

Agentified Group is a Group that is not a subclass of Agent. Figure 12 shows a Non-Agentified version of "ABC Customer Sales Dept". It has the same associated Roles; however, it does not have the "<<agent>>" stereotype. In order to interact with this Department, you must interact directly with one of its members -- a "Manager", an "ABC Buyer" or a "Broker".

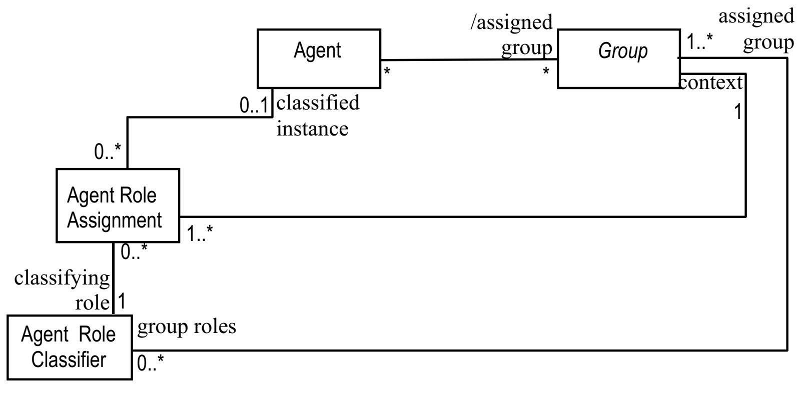


Figure 13. Abstract syntax for Agent Role Assignment.

5 Agent Role Assignment

Section 3 describes the association between Agents and their Agent Role Classifiers. However, the assignment of Agents to Roles is dynamic. This assignment is modeled by the Agent Role Assignment. Figure 13 shows the Agent Role Assignment and its associations. This section describes how Agent Role Assignment supports the dynamic association of Roles to Agents.

5.1 Agent Role Assignment as a Ternary Association

A direct association between Agent and Agent Role Classifier would represent that Agents play particular Roles, or Roles are played by specific Agents. However, this distinction is not sharp enough, because an Agent could play a given Role in one Group and not another. In Fig. 14, Agent2 plays the role of Broker in GroupB, but not in GroupC; furthermore, Agent3 is a Broker in GroupC, but not in GroupB. This situation illustrates that a role assignment between an agent and its role must be qualified by a group context. For example, the Broker role is used by GroupB and GroupC. The Broker for GroupB is Agent2, and the Broker for GroupC is Agent3.

Agent Role Assignments, then, are three-way, or ternary, associations. An Agent Role Assignment is a Class whose associated instances associate Roles and Groups and Agents. Each instance of the ternary Agent Role Assignment, associates a role, group, and an agent.

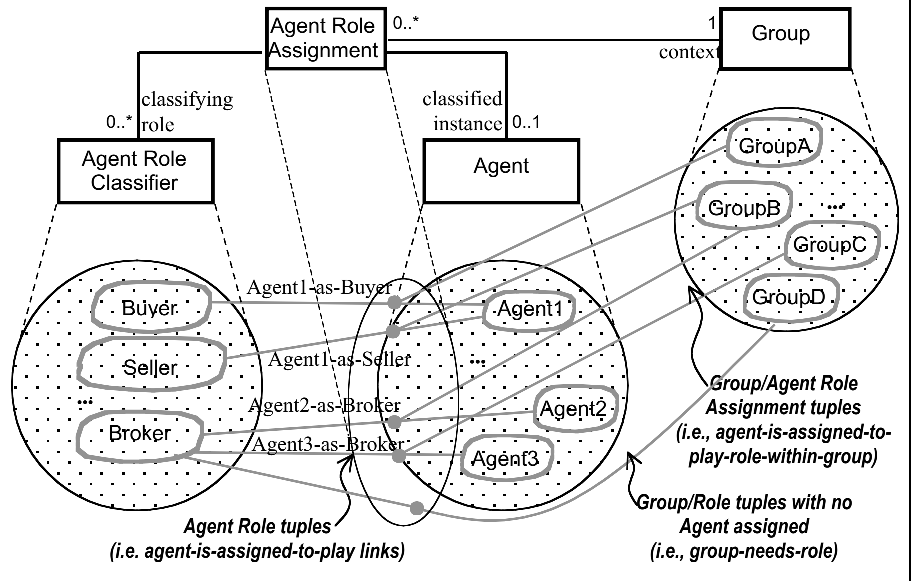


Figure 14. Example of Agent Role Assignments.

Contextualizing roles to groups has the additional advantage that it allows for a greater diversity of situations. For example, GroupB has two associated roles, Seller and Broker, played by Agent1 and Agent2, respectively. Agent1 is also a Buyer for Group A. This is allowed, even though Agent1 is now both a Buyer and a Seller, because Agent1 is a Buyer in one Group and a Seller in the other; thus, there is no conflict of interest.

5.2 Positions

While each Agent Role Assignment must have a Role and Group, it might not have an associated agent. Agent role assignments without agents can be called *positions*. For example in Fig. 14, there is an assignment that links the "Broker" role with "GroupD", but no Agent is assigned. This means that a slot, or *position*, has been assigned for some yet-to-be-defined agent to be empowered to play a "Broker" role in "GroupD". In other words, no agent has been assigned to "fill" the position. This approach is useful when "requisitioning" role assignments that must be filled at some point in the future to accomplish some task.

The set of all Agent Role Assignments that have agent assignments can be expressed as an association in its own right. These links are considered as part of the derivation of the association between Agent and Agent Classifier, as expressed in Figs. 1 and 5.

6 Conclusions

Agent-based systems are increasing both in size and diversity. This growth is pushing agent-based systems beyond a size that is manageable by individual organizations. Thus, there is a growing need for agents to be able to organize themselves according to their assigned tasks. Since these tasks may be complex, and beyond the abilities or knowledge of individual agents, this capacity to self-organize must be based on a solid metamodel. This metamodel needs to take into account individual agents, how they can interact, and how they can and do fit into groups.

This paper presents general metamodeling constructs for large-scale multi-agent systems. These concepts are anchored in the modeling and classification of agents according to both the capabilities that they have from their physical implementation (Agent Physical Classifiers) and from their current activities (Agent Role Classifiers). Agent activities are done in the context of *groups*. Furthermore, within the group their behavior conforms to specific patterns, and these patterns of behavior are enacted by the agents via the *roles* that they play in the group. For example, a group involved in a purchase may include an agent taking on the role of "Buyer", an agent taking on the role of "Trust Manager", and an agent taking on the role of "Seller". Within this group, the role of "Buyer" specifies the capabilities and governs the operations that are allowed for the agent that is playing it.

Using this metamodel within an agent system as a basis for understanding, regularizing and controlling agent behavior has many advantages. While the agent itself may be both large and diverse, the scoping of tasks within groups increases the predictability, stability, and reliability of the entire agent system. It also facilitates the monitoring and analysis of operations within the multi-agent system. This in turn means that the multi-agent system itself can scale to a greater size while still retaining properties of stability and controllability.

Acknowledgement

We would like to acknowledge the FIPA Modeling Technical Committee members for their contribution to this work. Also, this work was funded in part by NASA contract # NAS2-02093.

References

[Bellifemine, 2001] Bellifemine, Fabio, Agnstino Poggi, and Giovanni Rimassa, "JADE: A FIPA2000 Compliant Agent Development Environment," *Proceedings of the International Conference on Autonomous Agent*, Montreal, Canada, ACM, 2001.

[Biddle, 1966] Biddle, Bruce J., and Edwin J. Thomas, *Role Theory: Concepts and Research*, John Wiley and Sons, New York, 1966.

[Booch, 1999] Booch, Grady, James Rumbaugh, Ivar Jocobson, *The Unified Modeling Language User Guide*, Addison-Wesley, Reading, MA, 1999.

[Cybele, 2000] *OpenCybele User's Manual*, Intelligent Automation, Inc, http:www.opencybele.org/docs/Users.pdf, 2000.

[FIPA, 2004] Odell, James, Renato Levy and Marian Nodine, *FIPA Modeling TC: Agent Class Superstructure Model*, http://www.auml.org/auml/documents/CD2-04-21.doc, 2004.

[Gasser, 2001] Gasser, Les, "Perspectives on Organizations in Multi-Agent Systems," *Multi-Agent Systems and Applications*, Michael Luck *et al.* eds., Springer-Verlag, Berlin, pp. 1-16, 2001.

[Aronson, 2003] Aronson, J., Manikonda V., Peng W., Levy R. and Roth K.. *An HLA Compliant Agent-based Fast-Time Simulation Architecture for Analysis of Civil Aviation Concepts*. Spring SISO Simulation Interoperability Workshop, Orlando, Florida, April 2003.

[Karageorgos, 2003] Karageorgos, A., *Using Role Modeling and Synthesis to Reduce Complexity in Agent-Based System Design*, in *Dept. of Computation*, doctorate thesis, University of Manchester Institute of Science and Technology, Manchester, 2003.

[Katz, 1978] Katz, Daniel, and Robert L. Kahn, *The Social Psychology of Organizations*, (2nd ed.), John Wiley and Sons, New York, 1978.

[Mintzberg, 1993] Mintzberg, Henry, *Structure in Fives: Designing Effective Organizations*, Prentice Hall, Englewood Cliffs, NJ, 1993.

[Moreno, 1960] Moreno, J.L. ed., *The Sociometry Reader*, The Free Press, Glencoe, IL, 1960.

[Odell, 2002] Odell, James, "Objects and Agents Compared," *Journal of Object Technology*, Vol 1, Number 1, May, 2002.

[Odell, 2003] Odell, J., H.V.D. Parunak, and M. Fleischer, *The Role of Roles in Designing Effective Agent Organizations*, in *Software Engineering for Large-Scale Multi-Agent Systems*, A.F. Garcia *et al.*, Eds., Springer-Verlag: Berlin. pp. 27-38, 2003.

[OED, 1992] Oxford English Dictionary, (2nd ed.), Oxford University Press, Oxford, 1992.

[OMG, 2003] OMG, *UML 2.0 Superstructure Specification*, OMG document ptc/03-08-02, September 2, 2003.