Extending the UML Sequence Diagram to Model the Dynamic Aspects of Multi-Agent Systems

Viviane Torres da Silva Carlos José Pereira de Lucena

PUC-Rio, Computer Science Department, SoC+Agent Group,
Rua Marques de São Vicente, 225 - 22453-900, Rio de Janeiro, RJ, Brazil
{viviane, lucena}@inf.puc-rio.br

PUC-RioInf.MCC 15/03 May, 2003

Abstract. In this paper we present the dynamic aspects of multi-agent systems (MASs) based on the definition of their elements and relationships as proposed in the TAO metamodel. TAO is a conceptual framework based on agent and object abstractions that elicits an ontology connecting distinct abstractions such as objects, agents, organizations, roles and environments, and their relationships. The dynamic aspects of a MAS are particularly different from the dynamic aspects of an object-oriented system, since a MAS involves different entities and different relationships between those entities. The dynamic aspects characterize interactions between the elements of multi-agent systems that are related. In order to model the dynamic aspects of MAS we propose extending the UML sequence diagram, introducing new features.

Keywords. Modeling languages, agents, multi-agent systems, dynamic aspects.

Resumo. Neste artigo nós apresentamos os aspectos dinâmicos de sistemas multi-agentes (SMAs) nos baseando na definição dos seus elementos e relacionamentos proposto no metamodelo TAO. TAO é um framework conceitual baseado nas abstrações agentes e objetos que elícita uma ontologia conectando abstrações distintas como objetos, agentes, organizações, papéis e ambientes e seus relacionamentos. Os aspectos dinâmicos de um SMA são particularmente diferentes dos aspectos dinâmicos de sistemas orientados a objetos pois um SMA envolve entidades diferentes e relacionamentos diferentes entre as entidades. Os aspectos dinâmicos identificam as características das interações entre elementos de SMAs. Com o objetivo de modelar os aspectos dinâmicos de SMAs nós propomos estender o diagrama de seqüência proposto por UML introduzindo novas características.

Palavras-chave. Linguagem de Modelagem, agentes, sistema multi-agentes, aspectos dinâmicos.

1 Introduction

Multi-agent systems and object-oriented systems (OOSs) are particularly different systems [27][15]. MASs are composed of different elements that have different properties and characteristics and are related using different relationships. MASs are not composed only of objects. The heterogeneity of the elements' properties and characteristics make these systems fundamentally different from the OOSs. The behavior of each element differs from the behavior of objects; thus, the dynamic aspects of MAS are different than the dynamic aspects of OOSs.

In order to understand the difference between MASs and OOSs and to provide foundations to better understand and define the elements of a MAS we have proposed the TAO (Taming Agents and Objects) metamodel [22]. TAO is an evolving innovative conceptual framework centered on the agent and object abstractions, which are the foundations for modeling large-scale distributed software systems. The main role of the TAO framework is to provide a unified conceptual framework to understand distinct abstractions and their relationships in order to support the development of large-scale MASs. The proposed framework is based on an ontology that comprises well-known abstractions (such as objects and classes) and relates them to other abstractions (such as agents, roles, organizations and environments), which are the foundations for agent and object-based software engineering.

In a previous paper [23] we proposed a multi-agent system modeling language (MAS-ML) that extends the UML (Unified Modeling Language) metamodel [3] based on the TAO metamodel. The TAO metamodel has been used to produce a conservative extension [25] of the UML metamodel that includes the agent related notions that are part of the TAO conceptual framework (metamodel) while preserving all object-related concepts which constitute the UML metamodel. We have re-stated the UML class diagram and proposed two additional structural diagrams (organization and role diagrams). Moreover, in our previous work we based our proposal for the extension of the UML sequence diagram on a preliminary set of features that expresses some dynamic aspects of the MAS abstractions defined in TAO.

In the present work we propose a structured formulation of the domain-independent dynamic aspects of the elements described in the TAO metamodel extending the set of features discovered in our previous work [23]. The dynamic aspects of the elements of a MAS are based on the interactions between its elements. Interactions between elements occur because they are inherently related. As we will discuss later, some relationships characterize domain-independent behavior. In this work we classify the dynamic aspects of MASs into primitive domain-independent dynamic processes and high-level domain-independent dynamic processes. Some

dynamic processes are called primitive since every interaction depends on and is supported by these processes. Other dynamic processes support more complex domain-independent behavior that are specified in terms of primitive dynamic processes and other high-level dynamic processes. The domain-independent dynamic processes have been derived from the elementary behavior of all the elements of a MAS such as its creation and destruction and from the characteristics of the relationships between the entities of a MAS that define domain-independent interactions such as inhabiting an environment.

In order to model the dynamic aspects of MASs we propose to extend the UML sequence diagram. We do this by using an innovative approach that substantially differs from other proposals discussed in Section 5 (on related work). The analysis of primitive and high-level domain-independent behavioral processes of MASs allows us to detect features that need to be supported by UML's original sequence diagram so that it is able to model MASs. Our goal was to encompass all features related to the domain-independent MAS dynamic aspects in the UML sequence diagram.

The paper is organized as follows. Section 2 summarizes the structural aspects of the TAO metamodel and Section 3 defines the dynamic aspects of TAO. Section 4 describes and illustrates the extension of the UML sequence diagram according to dynamic aspects described in Section 3. Section 5 reviews some related work. Finally, Section 6 discusses some future research directions and presents the paper's conclusions.

2 An Overview of the Structural Aspects of TAO

The TAO metamodel [22] defines the entities and the relationships between the entities that are used in a MAS. The entities defined in TAO are object, agent, organization, role (agent role and object role), environment and event. Entities such as objects, agents, organizations, roles and environments are called elements and are defined based on their properties and relationships. Event is the only entity in TAO that is not defined based on properties and relationships. The properties of an element describe its state and behavior characteristics. The state of an element defines information about other elements of the system and the behavior of an element defines the actions or operations that the element can perform. An element must be related to another element; i.e., a relationship must exist between two elements, so that they can interact. The relationships link two elements and describe how these elements are related to each other. An element class defines properties and relationships that are common to all its instances. An element instance is a concrete manifestation of an abstraction to which a set of properties and relationships are applied [3]. An element instance of a class fulfills the description of its class.

2.1 Object

The state of an object stores information about itself, about the environment and about other objects and does not have any predefined structure. The behavior of an object defines the operations to either examine or change its state [9]. An object has control of its state. It performs operations that can modify its state during its lifetime. On the other hand, an object cannot modify its behavior and has no control of it; i.e., an object is not autonomous in the sense that it does everything that another element requests. Thus, objects are passive elements that do whatever anyone asks them to do and only when they are asked. The object relationships describe how objects are linked to system elements, such as other objects, agents, and roles. Moreover, objects can play roles defined by the organization that uses the object.

2.2 Agent

The state of an agent is expressed through mental components such as beliefs, goals, plans and actions [20][21][10]. An agent has beliefs or knowledge about the world, about itself and about other agents. The agent's goals consist of future states, or desires, which agents would like to reach or satisfy. An agent achieves a goal by executing a plan, which can be selected from a list of plans. Plans define a sequence of actions that is executed by an agent to achieve goals. Actions have a set of pre- and post-conditions. The behavior of an agent is expressed through its plans and actions that are based on its agent characteristics, e.g., interaction, autonomy and adaptation [20][11][8]. The relationships describe how an agent is linked to another element. Agents are elements that extend objects with structured state and agent behavioral properties.

2.3 Organization

Organizations group together the agents and sub-organizations of MASs [21][4]. At least one organization must inhabit the environment [22]. We call this organization the *main-organization*. As organizations specialize the definition of agents, an organization extends the properties and relationships defined by agents. The state of an organization is represented by its goals, beliefs and axioms. The axioms characterize the global constraints of the organization that agents and sub-organizations must obey. The goals and beliefs may be distributed among the agents. An organization's behavior is characterized by its actions and its plans and by the behavior of the agents that play roles in this organization. The relationships describe how an organization is linked to another element. An organization also defines roles that must be played by the agents and sub-organizations and other roles that can be played by objects within it. Every agent or MAS sub-organization plays at least one role in an organization.

2.4 Role

The two most important properties of roles are (i) its definition in the context of an organization and (ii) the fact that a role must be played by an agent, by an object or by a sub-organization. A role guides and also restricts the behavior of the instances that play the role. A role is an element, since it defines a set of properties and relationships.

2.4.1 Object Role

An object role guides and restricts the behavior of an object since it describes a set of features that are viewed by other elements. From the point of view of the element that is related to the object that is playing a role, the role identifies the properties that the element can see and identifies the available relationships. The state of an object role keeps information and the behavior of an object role keeps operations. An object role may restrict access to the state and behavior of an object but may also add information, behavior and relationships to the object that plays the role [13]. Since an object is not an autonomous element, its behavior is directly associated with the features viewed (or accessed) by other elements. The relationships of object roles describe additional relationships and types of relationships that were not previously available to objects.

2.4.2 Agent Role

An agent role guides and restricts the behavior of an agent because it describes the goals of the agent, additional beliefs, the actions that an agent must perform and the actions that an agent may perform while playing the role. An agent role is an element since it has state, behavior and relationships with other elements. The state of an agent role defines beliefs and goals that are added to the set of goals and beliefs associated with an agent. Duties, rights and protocols define the behavior of an agent role. The duties define actions assigned to the agent playing the role — i.e., its responsibilities [28] — and the rights describe actions that the agent can execute when playing the role — i.e., they describe permissions to actions. The protocols define the interactions between roles and other elements. The definition of the relationships of an agent role is based on the protocols associated with the role. In this way, the agent role may add a set of relations to the agent that plays the role.

Unlike an object role, an agent role does not add behavior to the agent; i.e., it does not add actions or plans to the agent. An agent must be capable of performing the actions and plans described by the role.

2.5 Environment

An environment extends the definition of an element since it further defines its properties, i.e., its state and its behavior and its relationships. The state of an environment stores the lists of resources and services and associated access permissions. The behavior of an environment is defined based on its characteristics. An environment can be a passive element, such as an object, or can be an active element, such as an agent having agency characteristics such as autonomy, adaptation and interaction. The relationships of an environment describe the elements it contains and other associated environments.

Agents, organizations, and objects inhabit (or are immersed in) environments [10][14]. We use objects as an abstraction for modeling resources [4]. Objects, agents or organizations can use resources when playing roles. The environment defines axioms associated with resources that restrict the access of objects and agents to them. Services are the public facilities, or public functions, provided and used by the entities of the system [12]. An entity such as an agent, an object, an organization or the environment can make some of its functions (services) public, which allows other entities to have knowledge and to resort to them.

2.6 Event

Events are generated in different ways. They can be generated by objects through the execution of their operations, by agents through the execution of their actions and by the environment when the environment is an active element [26]. An event generated by the environment, by an agent or by an object can trigger the execution of actions associated with agents or operations associated with objects that perceive the event. As a consequence, events are related to actions and operations that generate them and to actions and operations that perceive them.

2.7 Relationships

TAO defines eight relationships among its elements:

Inhabit:

The inhabit relationship specifies that the element that inhabits – the citizen – is created and destroyed in the habitat and may leave and enter habitats, respecting the habitat permissions. This relationship is applied to environments and agents, environments and objects and environments and organizations.

Ownership:

The ownership specifies that an element – the member – is defined in the scope of another element – the owner – and that a member must obey a set of global constraints defined by its owner. The member does not exist outside of the scope of its owner. This relationship is applied to roles – the members – and to organizations – the owners.

Play:

The play relationship specifies that the element that plays the role assumes properties and relationships defined by the role. The behavior of the element is guided by and restricted to the scope of the role. Every agent or sub-organization plays at least one role in an organization. Objects also can play roles.

Specialization/Inheritance:

The specialization relationship defines that the sub-element that specializes the super-element inherits all state and behavior associated with the super-element and also may add and redefine the properties and behavior associated with the super-element. This relationship may be used between objects, between agents, between organizations, between object roles and between agent roles.

Control:

The control relationship defines that the controlled element must do anything that the controller element requests. An agent role can control another agent role or an object role. Object roles only can control another object role.

Dependency:

This relationship defines that an element — the client — may be defined to be dependent upon another one — the supplier — to do its job. The dependency relationship specifies that the client cannot completely do its job unless it asks the supplier. An agent role can depend on another agent role and an object roles can depend on another object role.

Association:

If an element is associated with another element, it knows that the other element exists. The association relationship must define how one element interacts with another. Associations may be

used between roles (object roles or agent roles), environments, objects, agents and objects and organization and objects.

Aggregation:
If an element is aggregated with another element, we say that it is part of an aggregator. The aggregator may use the functionality available in its parts. This relationship may be applied between object roles, agent roles, objects, agents and organizations.

3 Dynamic Aspects of TAO

The dynamic processes that are to be described in this paper are directly related to the relationships between the elements of a MAS. Those dynamic processes are domain-independent behavior associated with the interaction between the elements. The dynamic processes of a MAS are classified as primitive (or elementary) dynamic processes and high-level dynamic processes.

3.1 Primitive dynamic processes

The basic or elementary interactions between the elements of a MAS are described as primitive dynamic processes. The creation and destruction processes are characterized as basic interactions between the creator or destructor and the associated element. These two dynamic processes are primitive since every interaction process depends on and is a function of the creation and destruction of the elements that will interact. An interaction occurs between two entities that have been created and have not yet been destroyed. Section 3.1.1 presents the dynamic processes that describe the creation of all elements of a MAS and Section 3.1.3 presents the corresponding destruction dynamic processes for those elements.

3.1.1 The elements' creation

The creation dynamic processes are presented with emphasis on four characteristics. First, the element that can be the creator of the other element is described. Next, pre-conditions to the creation of the element are also defined. Subsequently, the creation behavior is described and, finally, the post-conditions that must be guaranteed after the creation of the element are specified.

3.1.1.1 Agent Role creation

Creator: Only agents and organizations can create an agent role. The creator is the one that will play the role.

Pre-conditions: The creation of a role is conditional upon the existence of an agent or an organization to play the role; i.e., agents and organizations must have been created before the creation of a role. It is not possible to create a role instance without an agent or sub-organization to play the role.

The creation: A role instance is created based on a role instance class. The creation occurs when an agent or sub-organization commits to the role. Agents and sub-organizations commit to a role when they are created or when they change their roles, while agents commit to roles when they need to enter an organization to play another role. The creation of a role instance is based on an agent role class associated with an organization.

Post-conditions: A role must be associated with an agent or an organization after its creation.

3.1.1.2 Agent creation

Creator: An agent can be created by another agent, by an organization or by an environment.

Pre-conditions: The creation of agents is conditional upon the existence of the main-organization. Agents are created to play roles in organizations.

The creation: An agent instance must be created based on one of the agent classes available in the system. An agent is created in an environment to play a role in an organization that is registered in the same environment. When created, the agent instance is registered in the environment and a role instance is associated with it. The registration of the agent indicates the services made available by the agent to other elements. The element that creates the agent informs the initial role of the agent. The agent creates its role instance based on this information.

Post-conditions: Since every agent has to play at least one role in an organization, a role must be created and associated with the agent.

3.1.1.3 Organization creation

Creator: An organization can be created by an agent, by another organization or by the environment. The main-organization can only be created by the environment since there is no other element in the MAS.

Pre-conditions: The creation of organizations is conditional upon the existence of an environment.

The creation: Each environment must have exactly one main-organization. The unique organization that does not play roles is the main-organization since it is not a sub-organization of any other organization. Each organization instance is created based on an organization class. If the organization being created is a sub-organization, a role instance must be created and associated with the sub-organization since every sub-organization plays at least one role in the organization that owns it. The organization that owns the sub-organization defines a set of role classes that can be instantiated at the moment of the creation of sub-organizations. The organization creator defines the initial role of the sub-organization.

Every organization must be registered in the environment. The registration allows agents to find organizations to play roles and allows the environment to catalog the services that the organization makes available to other elements. Instances of the roles defined by the organization are not created at the moment of the organization creation. They are created only when there is an agent or sub-organization to play the role in the organization.

Post-conditions: A role must be created and associated with the organization if it is not the main-organization.

3.1.1.4 Environment creation

Creator: An environment can be created by another environment and by agents and organizations that inhabit another environment.

Pre-conditions: none

The creation: Every MAS must have one environment. The creation of every MAS element is conditional upon the creation of an environment instance. The environment must be created before other elements because the elements inhabit the environment. An environment is created based on an environment class. An environment is populated through the creation of objects, agents and sub-organizations to inhabit the environment and through the migration of agent and sub-organizations from another environment.

Post-conditions: A main-organization must be created.

3.1.1.5 Object Role creation

Creator: Object roles are created by agents and organizations through the roles that they play.

Pre-conditions: The creation of an object role is conditional upon the existence of an object to play the role; i.e., an object must have been created before the creation of a role. It is not possible to create an object role instance without an object to play the role.

The creation: An object role is created when an element, the creator, wants to access an object in an organization that restricts the view to the object. The creator must create an object role and associate it with the object that it wants to access. The role may add properties and relationships to the object and may restrict the access to the properties of the object. An object role is created based on an object role class.

Post-conditions: The creator must associate the object role with an object.

3.1.1.6 Object creation

Creator: Every element of the system can create an object.

Pre-conditions: An environment must have been created before the creation of an object.

The creation: The creation of an object takes place through the instantiation of a class. The object must be registered in the environment. The element that instantiates the object has a reference to the new object.

Post-conditions: none

3.1.2Creation Pattern

The creation pattern defines the order of the creation of MASs elements as illustrated in Figure 1. The steps illustrate the sequence of creations, i.e., the dependence between the creation of an element and the creation of another element. Specifically, the elements presented in the third step depend upon the creation of elements presented in the second step and elements presented in the second step depend upon on the creation of the element presented in the first step.

The first element that needs to be created in a MAS is the environment since agents, objects and organizations inhabit the environment. Then, the next elements to be created can be

objects and the main-organization. Since objects inhabit the environment, the creation of an object depends upon the creation of an environment; however, it does not depend upon the creation of organizations. The dependence between an object and an organization is associated with the role that the object plays in the organization. The role of an object represents the view of other elements (other objects, agents and sub-organizations in the organization) to the object.

Agents and sub-organizations cannot be created before the main-organization since they must play roles in organizations. For every agent or sub-organization created, a role must also be created and associated with the element. Moreover, object roles cannot be created before the main-organization and before the creation of objects because object roles are defined in organizations and must be played by objects.

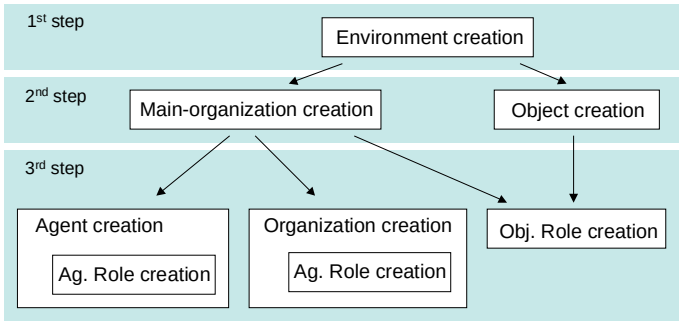

Figure 1 - Creation Pattern

3.1.3 The elements' destruction

The destruction dynamic processes are presented emphasizing the destroyer, the pre-conditions to the destruction of the element, the destruction process and the post-condition for the destruction.

3.1.3.1 Agent Role destruction

Destroyer: An agent role can only be destroyed by an agent or by an organization. The destroyer is the one that is playing the role.

Pre-conditions: none

The destruction: A role played by an agent or sub-organization is destroyed when the agent or sub-organization cancel it. If an agent or sub-organization has all of its roles destroyed it means that it will also be destroyed since it must play at least one role or that it is changing from an organization to another or from an environment to another and that it will choose another role to play.

Post-conditions: If the role that has been destroyed is the last role of the element, the element must also be destroyed.

3.1.3.2 Agent destruction

Destroyer: An agent can be destroyed by itself, by another agent, by an organization or by the environment.

Pre-conditions: All the agent roles must have been destroyed before the destruction of an agent.

The destruction: The agent deletes its registration in the environment, i.e., the description of the services that the agent had made available. All system elements lose the identification of the agent.

Post-conditions: none

3.1.3.3 Organization destruction

Destroyer: Organizations can be destroyed by themselves, by agents, by other organizations and by the environment. Note that the main-organization can only be destroyed by itself or by the environment.

Pre-conditions: All the roles played by agents, objects and sub-organizations must be destroyed before the destruction of the organization. The destruction of an organization is a recursive process since all its sub-organizations must also be destroyed or must leave the organization. Moreover, all the roles played by the organization must also be destroyed.

The destruction: The organization deletes its registration in the environment — i.e., the description of the services that the organization was making available. All system elements lose the identification of the organization.

Post-conditions: none

3.1.3.4 Environment destruction

Destroyer: An environment can be destroyed by itself, by an agent, by an organization or by another environment.

Pre-conditions: The main-organization must have been destroyed and all objects must also have been destroyed before the destruction of the environment takes place. Consequently there must be no agent or sub-organization in the system.

The destruction: The destruction of the environment characterizes that the elements that usually inhabit the environment have been destroyed or have been moved to another environment. All other environments lose the identification of the environment that was destroyed.

Post-conditions: none

3.1.3.5 Object Role destruction

Destroyer: An object role may be destroyed by an agent or by an organization that was using the object that was playing the role.

Pre-conditions: none

The destruction: All system elements that were using the same object and accessing the object by the same role lose the identification to the object. The destruction of the object role does not affect the object. The destruction of the object role affects the elements that were interacting with the object through its role.

Post-conditions: none

3.1.3.6 Object destruction

Destroyer: Any system element can destroy an object.

Pre-conditions: All object roles must have been destroyed.

The destruction: All system elements lose the identification of the object and the registration of the object in the environment is destroyed.

Post-conditions: none

3.1.4Destruction Pattern

The pattern bellow defines the order of the destruction of the elements as illustrated in Figure 2. The 1st step describes the elements that must be destroyed before the elements of the 2nd step. The elements of the 2nd step must be destroyed before the elements of the 3rd step. Since agents, objects and sub-organizations depend on the existence of the environment, the environment cannot be destroyed while it is the habitat of agents, objects and sub-organizations. The main-organization can only be destroyed when there are no more agents, sub-organization or objects playing roles. The main-organization must be destroyed before the environment since an organization inhabits an environment. The environment must be the last element to be destroyed.

The destruction of an agent, of a sub-organization and of an object is preceded by the destruction of the roles they have been playing. Agents and sub-organizations must be destroyed at the moment in which all their roles have been destroyed since they cannot exist without a role. The same does not happen with objects. Objects can exist without playing roles. However, an object can only be destroyed if all its roles have been destroyed. Moreover, the destruction of a sub-organization is preceded by the destruction of the roles that agents, objects and its sub-organizations are playing. A sub-organization can only be destroyed before there are no elements playing roles in it.

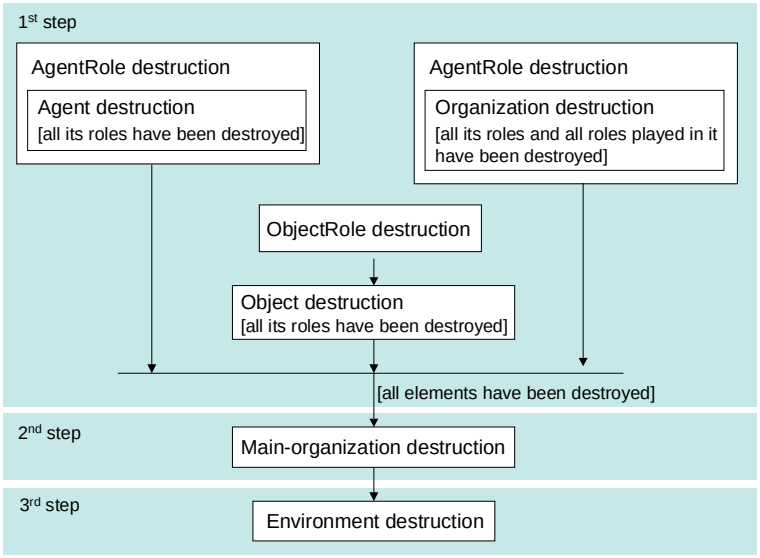

Figure 2 - Destruction Pattern

3.2 High-level dynamic processes

High-level dynamic processes are more complex domain-independent behavior that are described based on primitive dynamic processes and other high-level dynamic processes. The high-level dynamic processes describe patterns of behavior derived from the characteristics of the relationships between the entities of a MAS. From the set of relationships described in TAO three relationships are associated with domain-independent behavior: ownership, play and the inhabit relationships. Every agent, object and organization inhabit the environment. Every agent and organization play roles that are defined by organizations. The relationships inhabit, ownership and play characterize an interaction set that may occur in every MAS application. Relationships such as control, dependency, association and aggregation describe domain-dependent interaction. For instance, each application defines how a role controls another role, what kind of dependence links two roles and what the association between two elements means. The specialization relationship is a unique relationship that does not describe interactions between the related elements but rather is strictly a structural relationship.

States of an agent role

Before describing the high-level dynamic processes, it is important to introduce the definition of the states of agent roles. Agent roles have four states: created, active, inactive and destroyed as illustrated in Figure 3. A role is created when an agent or a sub-organization commits to play a role in an organization. The role created automatically changes its state and becomes an active role that is being played by the element. A role that is active can become inactive or can become destroyed. When the element playing the role cancels the commitment with the organization, the role is canceled; i.e., the role instance is destroyed. When the element leaves the organization where it is playing the role – not necessarily leaving its environment – without canceling the commitment with the organization, its roles become inactive. A role instance in the inactive state exists but is not being played. An inactive role keeps information about the element that was playing it and it can become active if the element returns to the organization – and consequently enters the environment – and reactivates its roles.

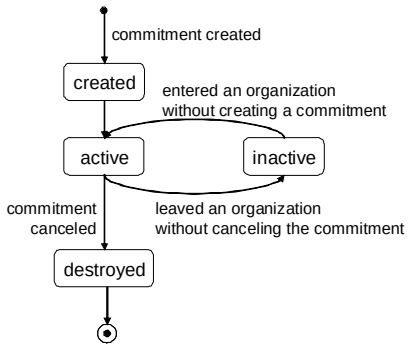

Figure 3 – States of an agent role

3.2.1 Ownership and play relationships

The ownership relationship specifies the members that the owner defines. Roles are created in an organization according to its specifications. The play relationship is directly related to the ownership relationship because the roles that are created must be played by agents, objects and sub-organizations. A role cannot be created if there is no element to play it.

The simple creation and destruction of roles have been described as primitive dynamic processes. However, the processes described by an agent and a sub-organization entering and leaving an organization are high-level dynamic processes. The process of an agent or a sub-organization entering an organization involves choosing an organization to play a role, choosing a role to be played and creating a role or reactivating a role. The process of leaving organizations involves the destruction of roles or the deactivation of roles that the agent or sub-organization has been playing. If the agent or sub-organization destroys or deactivates its unique role, the element should be destroyed or it should choose another organization to select and play another role. The dynamic process of leaving an organization may include the dynamic process of entering an organization. The processes of entering and leaving an organization are described as high-level domain-independent dynamic processes because they are based on primitive dynamic processes.

The behavior of agents, objects and sub-organizations playing roles in the organization are particular aspects described by the MAS applications. Since they are not domain-independent behaviors they will not be described in this paper.

3.2.1.1 Agent entering an organization

An agent can enter an organization where it has never been before or can enter an organization that it already knows. If an agent comes back to an organization where it has already been before, the agent can reactivate its inactive roles and can play new roles. If an agent wants to enter a new organization, it must look for the registration of organizations in the environment. The

registration of organizations may contain the identifier of organizations, their goals and axioms and the goals of its roles. The registration of an organization can be viewed as a notification of the roles that may be played by agents in that organization.

An agent may decide whether to enter an organization based on the goals of the organization and on the agent's own goals and based on the axioms described by the organization. The agent matches the organization goals to its own goals to see if they are compatible and the agent evaluates if it can follow the axioms. The match of goals may be as simple as comparing the elements of a set of goals or as complex as matching logical expressions. It is outside the scope of this paper to deal with the details of such matching. If the agent finds that the goals are compatible and that it can follow the axioms of the organization, the agent next should decide which role it should play.

An agent decides which role it will play based on the goals of the role and on its own goals. If the goals are compatible, the agent may ask the organization to play the role. Organizations can permit or deny the inclusion of agents. An organization decides about the necessity of having an agent play the chosen role based on its goals. Having the agent playing the chosen role may be contrary to the organization's goals, it may not influence the goals of the organization or it may prevent the organization from achieving its goals.

If the agent effectively enters an organization, the agent commits to playing the role and a role instance is created. The agent agrees to obey the duties and rights described in the role and the axioms described in the organization.

3.2.1.2 Agent leaving an organization

An agent may leave an organization to play roles in other organizations. When an agent leaves an organization it stops playing roles in the organization. The agent may leave the organization by canceling the commitment with the organization to play the role and destroying the role instance, or inactivating the role and not canceling the commitment.

As an agent must play at least one role, when the agent wants to leave the organization where it is playing a unique role the agent must choose another organization and choose another role to play in the organization. This process is described as an agent entering organizations. If the agent is playing more than one role, it is not obliged to enter another organization because it is already playing roles in other organizations. The process of leaving an organization may be followed by the process of entering another organization.

An agent may leave an organization for many reasons. Once an agent has achieved the goals of its roles it may want to leave the organization. The commitment between the agent and

the organization that specifies that the agent plays a role and obeys the axioms of the organization is automatically canceled. An agent may also want to leave an organization before it achieves the goals of the role. The agent breaks its commitment with the organization or deactivates its roles. The organization may not allow the agent to leave it before the goals of the role are achieved. The agent and the organization may negotiate.

3.2.1.3 Sub-organization entering an organization

The dynamic process of a sub-organization entering another organization is similar to the dynamic process of an agent entering an organization. A sub-organization can enter another organization where it has never been before or can enter another organization that it already knows. If a sub-organization comes back to an organization where it has previously been, the sub-organization can reactivate the roles that were inactive and can play new roles. On the other hand, if an organization wants to enter a new organization it must look for the registration of organizations in the environment. The sub-organization may decide whether to enter the other organization based on its own goals, on the goals of the other organization and on the axioms described by the other organization. If the goals are compatible and the sub-organization infers that it can follow the axioms, the sub-organization may choose a role to play in the other organization. The sub-organization decides which role it should play based on its own goals and on the goals of the role. As happens with agents, the organization that receives the foreign sub-organization can permit or deny the inclusion of the sub-organization. If the sub-organization effectively enters the other organization, it commits to play a role and a role instance is created.

When playing a new role, the sub-organization may adapt itself to follow the axioms described by the other organization and to follow the restrictions described by the role. The adaptation may influence the roles defined by the sub-organizations. The sub-organization may change the characteristics of the roles played by agents and its sub-organizations.

3.2.1.4 Sub-organization leaving an organization

This dynamic process is similar to the dynamic process of an agent leaving an organization. When an organization leaves another organization it may cancel the commitment with the organization to play the role and destroy the role instance, or it may change the state of the role from active to inactive. As an organization must play at least one role, if the organization stops playing its only role it must look for another role to play. The entering and leaving of a sub-organization for another organization is transparent to the agents that play roles.

3.2.2Inhabit relationship

As we have already seen, the inhabit relationship specifies that the element that inhabits – the citizen – is created and destroyed in the habitat and may leave and enter habitats, respecting the habitat permissions. Agents, objects and organizations inhabit environments. The inhabit relationships involve four different behaviors: creation and destruction of elements, the movement of elements from one environment to another and the actions of the elements in the environment. The creation and destruction of elements in the environment are described as primitive behavior and the actions of those elements in the environment are described as domain-dependent behavior that are out of the scope of this paper.

The elements that have the mobility characteristic can leave and enter the environment. Agents and sub-organizations can move from one environment to another. The movement of agents and sub-organizations characterizes domain-independent behavior since any application that allows agents and sub-organizations to move from one environment to another may be concerned with the same issues that involve these behaviors.

3.2.2.1 Agent moving from an environment to another

The dynamic process that characterizes the movement of an agent from one environment to another is composed of the dynamic process of leaving and entering environments. Moreover, the dynamic process of leaving an environment is composed of the dynamic process of leaving organizations and the dynamic process of entering an environment is composed of the dynamic process of entering organizations. The movement of an agent from one environment to another is a high-level dynamic process. This is because it is composed of primitive dynamic processes and other high-level dynamic processes involving interactions between agents and environments, agents and organizations to create and destroy roles and the change of role states.

Agent leaving an environment

An agent may move from one environment to another if it is specified as having the mobility characteristic. An agent discovers other environments based on the relationship between the environments. An environment may permit or deny an agent to enter or leave it. An agent that inhabits an environment follows the axioms associated with the environment. Moreover, an agent must also follow the axioms of the organizations where it is playing roles and the restrictions of the roles.

An agent cannot inhabit more than one environment. Moreover, an agent cannot play roles in an environment that it does not inhabit. When an agent leaves an environment, its

registration in that environment is canceled and it stops playing roles in that environment; i.e., roles that were in the active state change their state. The roles can be canceled or can become inactive.

Agent entering an environment

When the agent enters another environment, a registration in the new environment is created to represent the agent and a role has to be associated with the agent. The dynamic process of entering an environment is composed of the dynamic process of entering an organization. If the agent is entering a new organization it chooses a new role to play; otherwise, it may reactivate one of its roles. The dynamic process of creating a role or reactivating it is described in Section 3.2.1.1.

3.2.2.2 Organization moving from one environment to another

Organization leaving an environment

An organization may move from one environment to another if it is specified as having the mobility characteristics. What is different about the movement of an organization from the movement of an agent is the internal characteristics of an organization. As roles are being played in the organization, before the organization moves to another environment agents and sub-organizations must stop playing their roles. The roles instances being played in the organization may be destroyed or may become inactive before the organization moves to the other environment. The agents and sub-organization that were playing roles in the organization can also move to the other environment to reactivate their roles.

As with agents, an organization must follow the axioms associated with the environment it is leaving and the axioms associated with the environment it wants to enter. Moreover, it must also obey the axioms of the organizations where it is playing roles. These organizations may not permit their sub-organizations to leave. Sub-organizations must cancel the roles they are playing or deactivate them to leave organizations.

Organization entering an environment

The dynamic process of an organization entering an environment is as simple as the dynamic process of an agent entering an environment. The organization is registered in the new environment and it looks for a role to play in an organization of the new environment. It can reactivate its roles or it can create new roles.

3.2.3 High-level dynamic processes hierarchy

Some high-level dynamic processes described in the previous section are composed of other high-level dynamic processes. Figure 4 illustrates the hierarchy of these processes. The dynamic process that defines the movement of mobile elements from one environment to another is composed of the processes of leaving and entering an environment. The process of leaving an environment is composed of the process of leaving an organization, since an element that wants to leave an environment first must leave the organizations where it is playing roles. The process of leaving an organization may be followed by the process of entering an organization. If the element that leaves an organization is not leaving the environment, and is not playing other roles, the element must enter another organization to play a role. And finally, the dynamic process of entering an environment is composed of the process of entering an organization. An element that enters an environment must enter an organization to play roles.

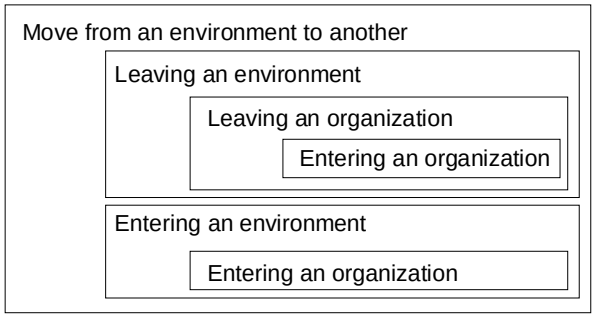

Figure 4 – High-level dynamic process hierarchy

4 UML Sequence diagram

A UML sequence diagram represents a set of interactions between objects playing roles in collaborations. We propose to extend the sequence diagram to represent the interaction between the elements that compose a MAS — i.e., between objects, agents, organizations and environments. The extensions proposed to the UML sequence diagram were based on the TAO structural model [23] and on the definition of the domain-independent dynamic processes presented in Sections 2 and 3, respectively. In this paper we are not concerned with the domain-dependent dynamic processes or with the internal behavior of the elements. We have concentrated our work on the domain-independent dynamic processes that characterize interactions between the elements.

While extending the UML sequence diagram, three sequence diagram elements were created and the existing diagram element called object was modified. We have created elements

to represent agents, organizations and environment and we have defined new pathnames to completely identify an element instance in a sequence diagram. Furthermore, we have created new stereotypes to identify new interaction types and have extended the definition of the stereotypes <<create>> and <<destroy>> defined in the UML metamodel. The extensions proposed to the UML sequence diagram make it possible, for instance, to represent the dynamic processes presented in the previews sections.

4.1 Sequence Diagram Elements

In UML, a sequence diagram element is represented by its pathname and by an icon. We propose the creation of different icons to represent different elements and define different pathnames associated with the elements. It is necessary to create three diagram elements to represent agents, organizations and environment, three new icons and associate new pathnames to the elements. Moreover, it is necessary to redefine the pathname associated with an object. UML defines the following name structure called pathname:

simple pathname _ object : class
complete pathname _ object/classifierRole : class :: package

The simple pathname of an object describes the object name and the class name separated by a colon. The complete pathname describes, to the left of the colon, the object and the name of the role of the object in the interaction. To the right of the colon, the complete pathname describes the class name and the package name of the class. The pathname associated with an element instance identifies it as a unique element in the system. The complete pathname completely specifies an instance participating in an interaction because it describes the instance by identifying the class in which the instance was based and its complete structure of packages.

4.1.1 Object pathname

TAO states that an object can interact with other elements by playing roles in organizations that inhabit an environment. If it is the case that the object is playing a role, to completely specify an object participating in an interaction we must mention the role instance that it is playing, the organization instance where the role instance is, and the environment instance the organization inhabits. An object may play more than one role that may be instances of different role classes and more than one role that may be instances of the same role class. Thus, the complete pathname of an object may describe the role it is playing in the interaction, the organization where it is playing the role and the environment that it inhabits. The complete pathname associated with

objects in sequence diagrams describes, to the left of the colon, the instances names and, to the right of the colon, the class names:

object/role/org/env : Class/ObjectRoleClass/OrganizationClass/EnvironmentClass

Organizations may be sub-organizations of other organizations. In the case of a sub-organization, it is necessary to represent all organization instances involved to completely specify an object. In this case, the pathname of the object should be modified to represent the hierarchy of organizations and the corresponding classes of each organization:

object/role/org1/.../orgN/env : Class/ObjectRoleClass/OrgClass1/.../OrgClassN/EnvClass

The complete pathname can be replaced by a simple pathname where appropriate. The information about the role instance played by the object, about the location of the organization and about the environment that it inhabits may be suppressed. The designer may suppress this information when the context of the interaction is well known or irrelevant. For example, the pathname may not contain the information about the environment if there is only one environment instance in the application (see (a)) or may not contain the information about the organization if there is only one organization instance in a given environment (see (b)).

a) object/role/org : Class/ObjectRoleClass/Organization
b) object/role/env : Class/ObjectRoleClass/Environment

4.1.2Agent pathname

The complete pathname that represents an agent in a sequence diagram is similar to the one that represents an object. The pathname describes the agent instance, the role instance that the agent is playing, the organization hierarchy where the agent is playing the role, the environment that it inhabits and their corresponding class names.

agent/role/org1/.../orgN/env : AgentClass/AgentRoleClass/OrgClass1/.../OrgClassN/EnvClass

The information about the role, organizations and environment may be suppressed and a simple pathname may be used. For example, if the information about the organization and the environment are well known, the pathname of the agent may be as simple as:

agent/role : AgentClass/AgentRoleClass

4.1.3 Organization pathname

A sub-organization may play a role in organizations and may interact with other elements. The sub-organization pathname describes the sub-organization instance, the role that the sub-organization is playing, the complete organization hierarchy where the sub-organization is defined, the environment that it inhabits and their corresponding class names. A sub-organization is represented by the following pathname:

suborg/role/org1/…/orgN/env : OrgClass/AgentRoleClass/OrgClass1/…/OrgClassN/EnvClass

A simple pathname may be used when appropriate. The information about the role, about the organization hierarchy where the sub-organization is defined and about the environment may be suppressed. For example, if the sub-organization plays only one role, and the organization hierarchy and the environment are well known, the pathname of the sub-organization may be as simple as:

suborg : OrgClass

4.1.4 Environment pathname

The pathname that completely specifies the environment is the simplest one. It describes the environment instance name and its class name:

env : EnvironmentClass

Table 1 identifies the instances that may appear in sequence diagrams, their icons and their associated complete pathnames. Objects are shown as rectangles, agents are shown as rounded rectangles and organizations are shown as double rounded rectangles. The environment is shown as a rounded rectangle if it is a proactive element and as a rectangle if it is a reactive element.

Table 1. The sequence diagram elements

Instances	Diagram Element
Object	*obj/role/org/env : Class/ObjectRole/Organization/Environment*
Agent	*agent/role/org/env : Agent/AgentRole/Organization/Environment*
Organization	*org/role/org/env : Organization/AgentRole/Organization/Environment*
Environment	*Proactive element*: *env : Environment*; *Reactive element*: *env : Environment*

4.2 The Stereotypes Associated with Interactions

The stereotypes `<<create>>` and `<<destroy>>` defined in UML are used to represent the creation and destruction of objects. We propose to use the same stereotypes to represent the creation and destruction of agents, organizations and environments. However, the semantics of those stereotypes need to be specialized when using them in association with agents and organizations. As agents and organizations play at least one role, a role must be associated with agents and organization when they are created. And since agent roles must be played by an agent or an organization, when an agent or organization is destroyed its roles must also be destroyed.

4.2.1 Stereotypes <<create>> and <<destroy>>

The stereotype `<<create>>` was specialized to represent the creation of agents, organizations and the environment and also to represent the association of a role instance to agent and organization instances. When an agent or an organization is created, a role is also created and associated with the agent or organization. We also propose to extend the use of the stereotype `<<create>>` to represent the creation of an object and the association of a role to the object. As the stereotype `<<create>>` represents creation of an element, the lifetime of the new element starts with the receipt of the message stereotyped as `created`. If the element is an agent, an organization or an object created to play a role, the receipt of the message stereotyped as `created` also indicates that a new role instance was created and associated with the element.

The stereotype `<<destroy>>` was specialized to represent the destruction of agents, organizations and environments, and the destruction of all role instances associated with agents,

organizations and objects. When an agent, an organization or an object is destroyed all the role instances that they are playing must also be destroyed at the same time. The lifetime of the element being destroyed ends with the receipt of the message stereotyped as `destroy` and are given the visual cue of a large X. Figure 5 illustrates the creation and destruction of an organization by an agent. The first block illustrates the creation of an organization. When the organization *Apple Sale* is created, a role instance called *Special Group number 1* is also created. The second and the third block illustrate the destruction of an organization. The second block represents the organization *Apple Sale* being destroyed when playing the role *Special Group number 1*. To better illustrate the destruction of all role instances, the designer may represent the organization being destroyed without any role instance associated, as represented in block number three.

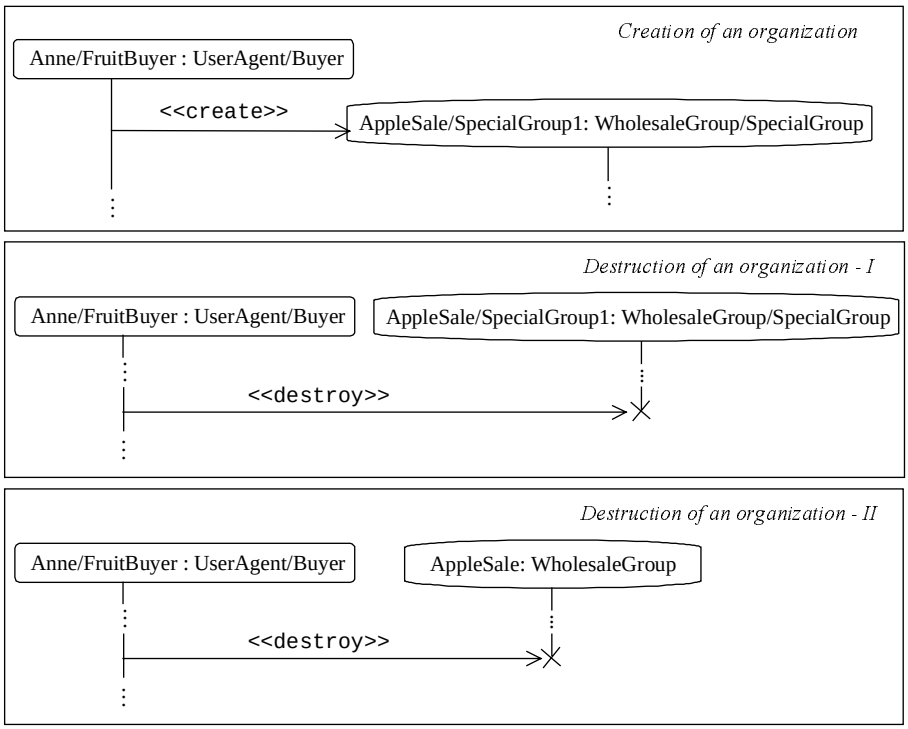

Figure 5 – Creation and destruction of an organization

Five new stereotypes associated with agents, organizations and objects were created to represent an agent, organization or object committing to a role and canceling a role and to represent an agent or an organization changing its roles, changing the state of a role from active to inactive and from inactive to active.

4.2.2 Stereotype <<role_commitment>>

The stereotype `<<role_commitment>>` was created to represent an agent, organization or object committing to a role. When an agent, organization or object commits to a new role it does not stop playing other roles. A new role instance is created and is associated with the element. The lifetime of a role starts with the receipt of the message stereotyped as `role_commitment`. It does not indicate that a new element instance is created but, rather, a new role instance.

The new role may be an instance of any role class defined by any organization class that inhabits the same environment as the element. Elements cannot play different roles in different environments. Therefore, the stereotype `<<role_commitment>>` can represent an agent or an organization entering an organization to play a role but cannot represent an agent or an organization entering a new environment.

Since the creator of an agent role is the agent or organization that will play the role (Section 3.1.1.1), the element that sends the message stereotyped as `role_commitment` is the element (agent or organization) that will play the role. The first block illustrated in Figure 6 exemplifies an agent that is playing the role *Fruit Buyer* committing with the role *Clothes Buyer* — i.e., creating the role *Clothes Buyer* and committing with the role. Since the creator of an object role is an agent or organization (Section 3.1.1.5), the element that sends the message stereotyped as `role_commitment` to the object is an agent or organization. The second block illustrated in Figure 6 exemplifies an agent playing the role *Fruit Buyer* creating the role *Desirer Fruit* and associating this role to the object *Apple*.

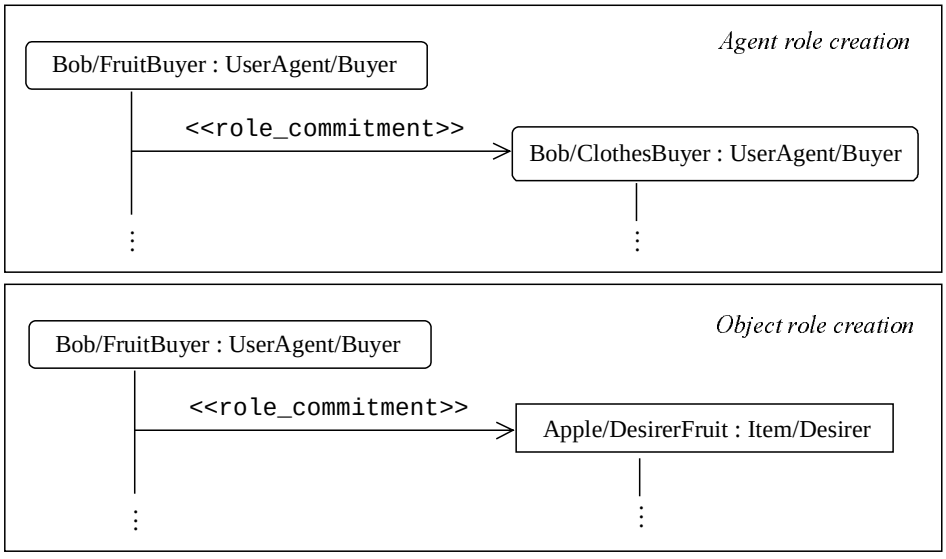

Figure 6 – Committing to a role

4.2.3 Stereotype <<role_cancel>>

The stereotype `<<role_cancel>>` represents that a role is being canceled. The commitment between the element – agent, organization or object – and the role is canceled and the element stops playing the role. The other roles associated with the elements are not modified. The act of canceling a role can be executed by an agent or an organization if it is playing more than one role. An agent or an organization cannot cancel its unique role unless it is changing from an organization to another or from an environment to another and so is changing its role. An agent and an organization must play at least one role in an organization. The lifetime of the role being canceled ends with the receipt of the message stereotyped as `role_cancel` and it is given the visual cue of a large X.

Since the destructor of an agent role is the agent or organization that is playing the role, the element that sends the message stereotyped as `role_cancel` is the element (agent or organization) that is playing the role. Since the destructor of an object role is an agent or organization, the element that sends the message stereotyped as `role_cancel` to the object is an agent or organization. Figure 7 shows an agent canceling one of its roles and an agent canceling the role of an object. An agent can cancel a role while playing the role or can cancel a role while playing another role as presented in the first block.

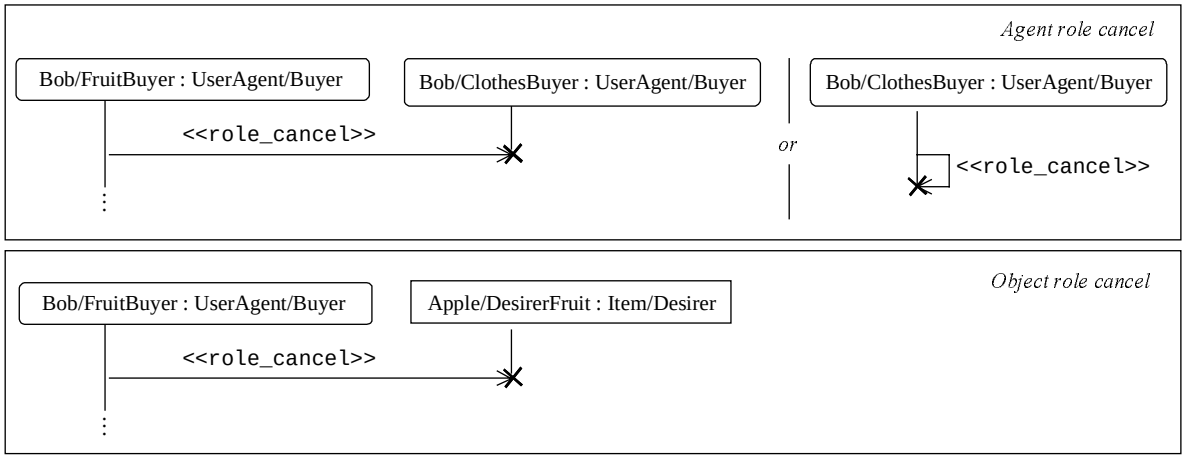

Figure 7 – Canceling a role

It is possible to illustrate an agent or an organization leaving another organization using the stereotype `<<role_cancel>>`. An agent leaving an environment cannot be illustrated using this stereotype since to leave an environment an agent must stop playing all its roles in that environment and start playing at least one role in another environment.

4.2.4 Stereotypes <<role_deactivate>> and <<role_activate>>

The stereotype `<<role_deactivate>>` changes the state of a role that an agent or organization is playing from active to inactive. The lifetime of the role becomes inactive with the sending of the message stereotyped as `role_deactivate` and it is given the visual cue of a large — ending the active state. The stereotype `<<role_activate>>` changes the state of a role from inactive to active. The lifetime of the role becomes active with the receipt of the message stereotyped as a `role_activate` and it is also given the visual cue of a large — starting the active state. The other roles that the agent or organization is playing do not change when a specific role changes its state.

4.2.5 Stereotypes <<role_change>>

The stereotype `<<role_change>>` represents an agent or an organization changing its role. An object does not change one role to another because it does not have the autonomy to choose its roles. An agent or an organization chooses the role of an object. Agents and organizations create roles and associate them to objects (commitment) or cancel the roles of the objects.

The element that changes its roles stops playing a role and starts playing another role. On one hand, the element can create a new role or can activate one of its inactive roles. On the other hand, the role that the element was playing may be canceled or may become inactive. The receipt of the message stereotyped as `role_change` may indicate that a new role has been created or that a role has been reactivated. Moreover, the message also indicates that the original role has been canceled or that it has become inactive. Figure 8 demonstrates these four possibilities.

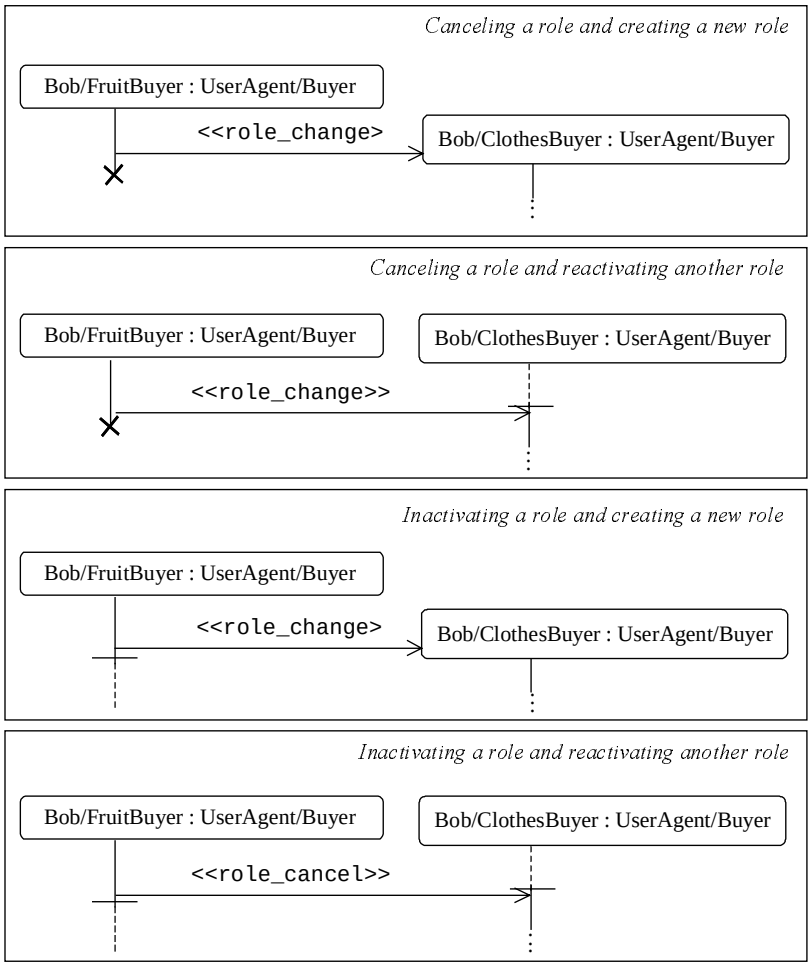

Figure 8 – Changing roles

Figure 9 illustrates the use of the five new stereotypes. Suppose that the agent *Bob* commits to the role *Clothes Buyer* that is defined in the organization *Clothes Retail* when playing the role *Fruit Buyer* in the organization *Fruit Retail*. At level I *Bob* is playing the role *Fruit Buyer* and the role *Clothes Buyer* at the same time in a different organization.

After a while, *Bob* deactivates the role *Clothes Buyer* and then reactivates it again. At level II, *Bob* is playing only the role *Fruit Buyer* and at level III, *Bob* again is playing the roles *Fruit Buyer* and *Clothes Buyer*. When playing the role *Clothes Buyer*, *Bob* cancels the role *Fruit Buyer*. At level IV, Bob is playing only the role *Clothes Buyer*. Then, Bob changes his role from *Clothes Buyer* to *Clothes Seller*. At level V, *Bob* stops playing role *Clothes Buyer* at the Environment 1 and starts playing the role *Clothes Seller* at the Environment 2. The role *Clothes Buyer* becomes inactive. After that, *Bob* changes its role again. *Bob* reactivates the role *Clothes Buyer* and cancels the role *Clothes Seller*. At level VI, *Bob* is playing only the role *Clothes Buyer*.

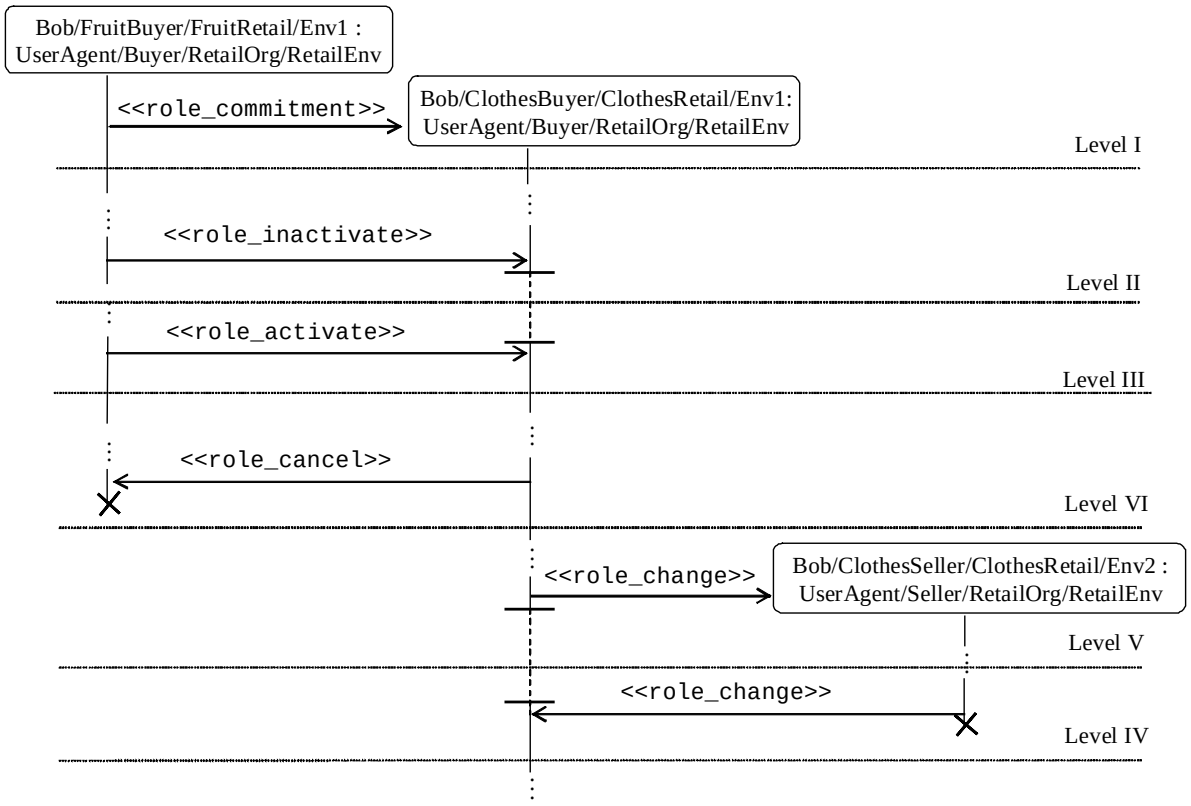

Figure 9 – The five new stereotypes

4.2.6 Using the stereotypes to model the dynamic aspects of a MAS

Using the extension of the stereotypes <<create>> and <<destroy>> and the five new stereotypes presented in this paper, it is possible to illustrate all the primitive dynamic processes and the high-level dynamic processes of a MAS. The creation of elements and the association of roles with agents, organizations and objects are represented using the stereotype <<create>>. The destruction of elements and the destruction of all roles played by agents, organization and objects are modeled using the stereotype <<destroy>>. All primitive dynamic processes are modeled using these two stereotypes.

When using the stereotypes <<create>> and <<destroy>> the restrictions defined in the primitive dynamic processes of creating and destroying an element must be obeyed. The designer must guarantee that the *pre-conditions* and *post-conditions* are satisfied. The creator and the destructor must be the one described in the *creator* and *destruction* restrictions. The use of these stereotypes encapsulates what is described by the creation and destruction of the element.

The dynamic process of an agent or sub-organization entering an organization can be illustrated using the stereotypes <<role_commitment>> and <<role_activate>> since these

stereotypes indicate that a role is being associated with an agent or a sub-organization. A new role can be created or an old role can be reactivated in a different organization. Moreover, the behavior described by an agent or organization leaving an organization can be represented using the stereotypes <<role_cancel>> and <<role_deactivate>>. An agent can cancel all the roles it was playing in an organization or can deactivate them. The stereotype <<role_change>> can illustrate an agent or organization leaving an organization and entering another organization.

The dynamic process of an agent or sub-organization entering an environment cannot be illustrated using the stereotypes <<role_commitment>> or <<role_activate>> since an agent or an organization playing a role in an environment cannot commit to another role or reactivate another role in another environment. Agents and organizations cannot play roles in different environments. Furthermore, the stereotypes <<role_cancel>> and <<role_inactive>> cannot illustrate an agent or organization leaving an environment. The use of those stereotypes to illustrate an agent leaving an environment would assume that the agent or organization was playing roles in different environments. The designer should use the stereotype <<role_change>> to model an agent or organization leaving an environment, emphasizing that the agent or organization stops playing a role in an environment and starts playing a role in another environment. The element moves from an environment to another changing its roles.

Moreover, when modeling a high-level dynamic process, the designer has two options: model the whole dynamic process as described in the text or suppress the details when a dynamic process is a well-defined process. As illustrated in Figure 9, the complexity of an agent changing from one environment to another that involves choosing another environment, choosing another organization and choosing another role, for instance, is encapsulated by the stereotype <<role_change>> at level II. The whole process of an agent or organization moving from one environment to another is well known, so it can be omitted here.

5 Related Work

In this paper we present primitive and high-level dynamic processes that are associated with the different relationships that exist among the elements of a MAS. As the elements are related and interact we classify the creation and destruction of the elements as primitive dynamic processes since they are the basis for other interactions. We propose to represent the creation and destruction of the elements of a MAS taking into consideration their characteristics and extending the stereotypes <<create>> and <<destroy>> defined in the UML sequence diagrams. Papers available in the literature such as [17][18][1][2] that propose to extend the UML sequence

diagram according to the characteristics of MASs do not take into consideration the creation and destruction of the MAS elements.

Although other initiatives such as [24][5][16][7][6] also agree that organizations, roles and agents are related and that they inhabit environments [19][16], they do not describe the dynamic aspects that involve these relations. Organizations are related to agents since they define roles [24][5][7][6] that are played by agents [1][24][16][7][6] and sub-organizations. Each agent can play one or more roles in an organization [5][16][6] and an agent can play roles in different organizations [5][6]. Since agents can play roles in different organizations, agents can enter [7][6] and leave organizations. Agents can commit to new roles, can change their roles [24][16][6] and can cancel them. Based on the relation between agents, organizations and roles, we have defined high-level domain independent dynamic processes. We classify the behavior of an agent or sub-organization entering and leaving an organization as domain-independent high-level dynamic processes since all MASs that describe agents, organizations and roles define the same kind of relationships among them, and since they are based on and supported by the primitive dynamic processes. Furthermore, the behavior of agents and sub-organizations moving from one environment to another also is defined as high-level dynamic processes, since agents and sub-organizations inhabit environments and have the ability to move from one environment to another. The dynamic processes that describe an agent or sub-organization moving from one environment to another are high-level domain-independent because every MAS that defines the moving capability of agents and sub-organizations may define how they move from one environment to another. This is also so because these processes are based on other high-level dynamic processes.

In [17][18][1][2], the authors define interaction protocols between the agents of a MAS. They propose to extend the UML sequence diagram to support concurrent threads of interactions. They do not propose a representation for the creation of an agent or for its destruction. Although they define that agents can play more than one role, they do not define the representation of an agent committing to a new role or canceling one of its roles. Moreover, they do not define the representation of other MAS elements, such as environments and organizations. Our proposal can be considered complementary to their proposal because we do not describe interaction protocols and they do not represent primitive and high-level dynamic processes involving those elements.

6 Conclusions and Future Work

In this paper we propose to extend the UML sequence diagram to model some dynamic aspects of MASs. We present a structured formulation of the domain-independent dynamic aspects of the

elements described in TAO. Analyzing the domain-independent relations between the elements of MASs we derive a set of stereotypes that extend the original UML sequence diagram to be able to model MASs dynamic aspects. As a consequence, we have extended the TAO meta-model by adding a description of their domain-independent dynamic aspects to the description of the static aspects of MASs.

It is necessary to investigate the internal behavior and characteristics of each element of a MAS in order to completely extend the UML sequence diagram. The internal behavior of each element may generate new features to be included in the original sequence diagram. Odell [18] has proposed some extensions to the UML sequence diagram by analyzing the internal behavior of agents but so far does not analyze the internal behavior of other MAS elements, such as organizations and environment. Moreover, it is necessary to evaluate domain-dependent dynamic aspects of the elements of a MAS. Different domains may require different behavioral characteristics. However, due to the variety of possible MAS application domains only extensive experimentations with such systems will highlight new behavioral patterns.

As we have not yet completely extended the UML metamodel — i.e., as we have not developed extensions for all UML diagrams — this paper contains only a partial version of MAS-ML. We are in the process of analyzing all the UML diagrams and modifying them when appropriate, based on the extension of the UML foundation package. Case studies will illustrate different levels of complexity and will cover a wide spectrum of MASs characteristics. While developing the case studies we will also address the issue of mapping MAS-ML models into OO implementations. Preliminary work in this area has already been developed.

Acknowledgement

This work has been partially supported by CNPq under grant 140646/2000-0 for Viviane Silva. Viviane Silva and Carlos Lucena also are supported by the PRONEX Project under grant 664213/1997-9, by the ESSMA project under grant 552068/2002-0 and by the 1st art. of decree number 3.800, of 04.20.2001. The authors would also like to acknowledge the contributions of Ricardo Choren.

References

[1] Bauer B. (1999) "Extending UML for the Specification of Agent Interaction Protocols". Submitted for the 6th Call for Proposal of FIPA.

[2] Bauer B., Müller, J. and Odell, J. (2001) "Agent UML: A Formalism for Specifying Multiagent Interaction," In Ciancarini P. and Wooldridge M. (eds.) International Journal of Software Engineering and Knowledge Engineering, Vol. 11, No. 3, pp. 207-230.

[3] Booch G., Rumbaugh J., and Jaconbson I. (1999) “The Unified Modeling Language User Guide”, First Edition, Addison-Wesley Longman Publishing Co., Inc.

[4] Caire G. (2001) "MESSAGE: Methodology for Engineering Systems of Software Agents Initial Methodology". In Technical report, EDIN 0224-0907, Project P907, EURESCOM.

[5] Ferber J., Gutknecht O., Jonker C., Mueller J., and Treur J. (2000) “Organization Models and Behavioural Requirements Specification for Multi-Agent Systems” In Proceedings of the ECAI 2000 Workshop on Modelling Artificial Societies and Hybrid Organizations.

[6] Giampapa J. and Sycara K. (2002) "Team-Oriented Agent Coordination in the RETSINA Multi-Agent System", Technical Report CMU-RI-TR-02-34, Robotics Institute, Carnegie Mellon University, December.

[7] Hannoun M., Boissier O. and Sichman J. (2000) "Moise: An organizational model for multi-agent systems," In Monard M., Sichman J., (eds.) Advances in Artificial Intelligence, th Ibero-American Conference on AI, 15th Brazilian Symposium on AI, LNAI 1952 Springer, pp. 156.

[8] Huhns M. and Singh M. (1998) “Agents and Multi-agent Systems: Themes, Approaches and Challenges,” In Huhns M. and Singh M., (eds.) Readings in Agents, Morgan Kaufmann, pp. 1-23.

[9] Jacobson I. (1992) “Object-Oriented Software Engineering”, First Edition, Addison-Wesley Longman Publishing Co., Inc.

[10] Jennings N. (1999) “Agent-Oriented Software Engineering,” In Proceeding of the Twelfth International Conference on Industrial and Engineering Applications of Artificial Intelligence, Cairo, Egypt, pp. 4-10.

[11] Jennings N. (2000) “On Agent-based Software Engineering,” In Artificial Intelligence, Vol. 117, n. 2, pp. 277-296.

[12] Kinny D. and Georgeff M., (1997) “Modeling and design of multi-agent systems,” In Müller J., Wooldridge M. and Jennings N. (eds.) Intelligent Agents III, Springer 1193, pp. 1-20.

[13] Kristensen B., Østerbye K. (1996) “Roles: Conceptual Abstraction Theory and Practical Language Issues” In Theory and Practice of Object Sytems Vol. 2, n. 3, 143–160.

[14] Lind J. (2000) “MASSIVE: Software Engineering for Multi-agent Systems,” PhD Dissertation, Universität des Saarlandes, Saarbrücken, Germany.

[15] Lind J. (2001) “Issues in agent-oriented software engineering.” In Ciancarini P., Wooldridge M., (eds.) Agent-Oriented Software Engineering, First International Workshop, AOSE 2000, LNCS 1957 Springer, Limerick, Ireland., pp. 45-58.

[16] Mellouli S., Mineau G. and Pascot D. (2002) “Multi-Agent System Design” In: Third International Workshop Engineering Societies in the Agents World, Madrid, Spain.

[17] Odell J., Parunak H. and Bauer B. (2000) "Extending UML for Agents," In Wagner G., Lesperance Y. and Yu E., (eds.) Proceedings of the Agent-Oriented Information Systems Workshop (AOIS), Austin, pp. 3-17.

[18] Odell J., Parunak H. and Bauer B. (2001) "Representing agent interaction protocols in UML," In Ciancarini P., Wooldridge M., (eds.) Agent-Oriented Software Engineering, First International Workshop, AOSE 2000, LNCS 1957 Springer, Limerick, Ireland. pp. 121-140.

[19] Odell J., Parunak H., Fleischer M. and Breuckner S (2003) "Modeling Agents and their Environment" In Giunchiglia F., Odell J., Weiß G., (eds.) Agent-Oriented Software Engineering III, Third International Workshop, AOSE 2002, LNCS 2585 Springer, Bologna, Italia.

[20] OMG (2000) Object Management Group – Agent Platform Special Interest Group: Agent Technology – Green Paper. Version 1.0.

[21] Shoham Y. (1993) “Agent-Oriented Programming,” Artificial Intelligence, 60.

[22] Silva V., Garcia A., Brandao A., Chavez C., Lucena C. and Alencar P. (2003) “Taming Agents and Objects in Software Engineering,” In Garcia A., Lucena C., Zamboneli F., Omicini A, and Castro J. (eds.) Software Engineering for Large-Scale Multi-Agent System, LNCS, Springer.

[23] Silva V., Lucena C. (2003) "From a Conceptual Framework for Agents and Objects to a Multi-Agent System Modeling Language", Technical Report CS2003-03, School of Computer Science, University of Waterloo, Canada (submitted to JAAMAS).

[24] Stone P. and Veloso M. (1999) “Task Decomposition, Dynamic Role Assignment, and Low-Bandwidth Communication for Real-Time Strategic Teamwork,” Artificial Intelligence.

[25] Turski W. and Maibaum T. (1987), “Specification of computer programs,” Addison-Wesley Longman Publishing Co., Inc., Boston, MA.

[26] Wagner G. (2000) “The Agent-Object-Relationship Metamodel,” In Proceedings of the 2nd International Symposium: From Agent Theory to Agent Implementation together with EMCRS 2000, Vienna, Austria.

[27] Wooldridge M. and Ciancarini P. (2001) “Agent-Oriented Software Engineering: the State of the Art.” In Ciancarini P., Wooldridge M., (eds.) Agent-Oriented Software Engineering, First International Workshop, AOSE 2000, LNCS 1957 Springer, Limerick, Ireland, pp. 1-28.

[28] Wooldridge M., Jennings N. and Kinny D. (1999), "A Methodology for Agent-Oriented Analysis and Design," In Proceedings of the Third International Conference on Autonomous Agents (Agents'99), ACM Press, pp. 69-76.