4
A linguagem de Modelagem MAS-ML

Neste capítulo, apresentamos a linguagem de modelagem MAS-ML. Ela estende UML com base nas propriedades dinâmicas e estruturais apresentadas no metamodelo de TAO. Nossa extensão de UML concentra-se no diagrama de classes estático e no diagrama de seqüência de interação. Esses dois diagramas foram escolhidos porque são os mais usados e porque é possível usá-los para incorporar os aspectos dinâmicos e estruturais do metamodelo de TAO.

Devido ao conjunto de diferentes entidades e relacionamentos definido no metamodelo de TAO que foi incorporado no metamodelo de UML, propomos diferentes diagramas estruturais cujo foco está nos diversos aspectos de extensões que devem ser cobertos pela MAS-ML resultante. Os diagramas estruturais que compõem a MAS-ML são o diagrama de classes estendido e dois novos diagramas chamados diagrama de organização e diagrama de papel (consulte as Seções 4.3.2, 4.3.3 e 4.3.4). Esses três diagramas – diagrama de classes, organização e papel – mostram todas as entidades e relacionamentos definidos em TAO. Além disso, propomos estender o diagrama de seqüência de UML para representar os aspectos dinâmicos de SMAs, ou seja, para representar as interações entre a instância do SMA e as intra-ações definidas por cada instância.

4.1.
Os Mecanismos de Extensão de UML

A fim de modelar as entidades e os relacionamentos definidos em TAO, o metamodelo de UML foi estendido. Como nosso objetivo é produzir uma extensão conservativa de UML (Turski et al., 1987), as metaclasses definidas em UML não foram modificadas durante a extensão. Os mecanismos de extensão descritos em UML são a criação de novas metaclasses, definições de *tags*, restrições e estereótipos.

A criação de novas metaclasses é definida como o mecanismo de extensão de primeira classe porque modifica os metamodelos existentes (UML, 2004). As definições de *tags* especificam novos tipos de propriedades que podem ser

associados aos elementos do modelo. As restrições também podem ser acopladas a qualquer elemento do modelo a fim de refinar lingüisticamente sua semântica. Um estereótipo é o elemento de um modelo que define valores adicionais (com base na definição das *tags*), outras restrições e, opcionalmente, uma nova representação gráfica. Todos os elementos do modelo marcados (classificados) por um ou mais estereótipos recebem os valores e as restrições definidos pelo estereótipo, além dos atributos, associações e superclasses que o elemento tem em UML padrão. Os estereótipos oferecem uma forma de definir subclasses virtuais de metaclasses de UML com novos metaatributos e outra semântica.

Ao estender UML de acordo com os conceitos de TAO, não é possível usar apenas os mecanismos de extensão de *tags*, restrições e estereótipos fornecidos por UML. Portanto, foram adicionados ao metamodelo de UML novas metaclasses e novos estereótipos associados a novas entidades, propriedades e relacionamentos definidos em TAO e que não estavam presentes em UML.

4.2. Metamodelo de MAS-ML

A partir do conjunto de entidades definidas em TAO, apenas uma entidade é descrita em UML. A definição apresentada em TAO para a entidade *object class* é semelhante à definição apresentada no metamodelo de UML para a metaclasse chamada *Class*. Então, usamos um mapeamento direto a partir dessa entidade de TAO até a respectiva metaclasse de UML.

A partir do conjunto de relacionamentos definidos em TAO, os relacionamentos *association*, *aggregation*, *dependency* e *specialization/generalization* também são definidos em UML. No entanto, as definições apresentadas em TAO estendem as definições apresentadas em UML. Os relacionamentos de UML são aplicados às classes de objeto, e os relacionamentos de TAO são aplicados não só nas classes de objeto, mas também nas classes do agente e nas classes de organizações, por exemplo. De fato, os relacionamentos de UML são aplicados à metaclasse *Element* e não diretamente à metaclasse *Class* (a metaclasse da classe de objeto). A metaclasse *Element* é a superclasse da metaclasse *Classifier*, e a metaclasse *Classifier* é a superclasse da metaclasse *Class*.

Como os relacionamentos de UML são aplicados à metaclasse *Element*, e todas as abstrações de primeira ordem relacionadas a agentes (agente, organização, ambiente, papel de objeto e papel do agente) estendem a metaclasse *Classifier*, não há necessidade de criar novas metaclasses para representar os relacionamentos estendidos. Ao estender o metamodelo de UML incorporando especializações relacionadas a agentes de *Classifier*, os relacionamentos *specialization*/*generalization*, *aggregation*, *association* e *dependency* estão diretamente relacionados a essas novas metaclasses. Entretanto, na Seção 3.1.6, descrevemos as extensões feitas a cada relacionamento, enfatizando quais abstrações podem ser vinculadas por meio de cada relacionamento.

Conforme mencionado anteriormente, todas as abstrações relacionadas a agentes estendem a metaclasse *Classifier*. Essa metaclasse está relacionada às metaclasses *StructuralFeature* e *BehavioralFeature*. Uma característica estrutural é uma característica de um classificador que especifica a estrutura de instâncias do mesmo. Uma característica comportamental é uma característica de um classificador que especifica um aspecto de comportamento de suas instâncias. A metaclasse *StructuralFeatures* é uma generalização da metaclasse *Property* (*attributes* de uma classe são representados como instâncias de *Property*), e a metaclasse *BehavioralFeatures* é uma generalização das metaclasses *Operation*, de acordo com a definição de UML (UML, 2004).

Nas próximas subseções, apresentamos as extensões propostas para o metamodelo de UML. Descrevemos as metaclasses e os estereótipos criados para representar as entidades, as propriedades e os relacionamentos definidos em TAO. Ademais, também definimos as associações entre as novas metaclasses. Essas associações são derivadas de relacionamentos independentes do domínio (relacionamentos *inhabit*, *ownership* e *play*).

4.2.1. Agente

Com base na diferença entre *agentes* e *objetos* apresentada no Capítulo 3, não criamos um estereótipo `<<agent>>` e o associamos à metaclasse de UML *Class* como fizeram outras propostas, AUML e AOR por exemplo. Se um estereótipo tivesse sido usado, os agentes teriam as mesmas propriedades e relacionamentos definidos na metaclasse *Class*, além de outros valores e restrições definidos pelos

estereótipos. Tampouco estendemos a metaclasse *Class*, porque classe é um tipo de *Classifier* cujas características são atributos e operações (UML, 2004). Agentes e objetos definem diferentes características comportamentais e estruturais.

A fim de estender o metamodelo de UML para descrever agentes, criamos uma nova metaclasse chamada *AgentClass*. A *AgentClass* estende a metaclasse de UML *Classifier* e, portanto, está associada às metaclasses *StructuralFeature* e *BehavioralFeature*.

4.2.1.1. Propriedades do Agente

As características estruturais definidas por um agente são *objetivo* e *crença,* e suas características comportamentais são *ação* e *plano*. A fim de descrever *AgentClass* completamente, é necessário definir as características do agente *objetivo*, *crença*, *ação* e *plano*.

Objetivo e Crença

Os objetivos e as crenças são características de agentes, organizações, ambientes ativos e papéis do agente. A característica crença associada a agentes, organizações, ambientes ativos e papéis do agente possuem a mesma definição. Contudo, a característica objetivo relacionada a agentes, organizações e ambientes ativos não tem a mesma definição da característica objetivo relacionada a papéis do agente. Um objetivo associado a um agente, organização ou ambiente ativo está relacionado a planos que alcançam o objetivo, mas um objetivo associado a um papel do agente não está relacionado a planos.

Agentes, organizações, ambientes ativos e papéis podem ter muitos objetivos e crenças. Objetivos e crenças são expressos como atributos de uma entidade caracterizada por um tipo, um nome e um valor padrão que pode ser alterado durante a execução da entidade. Como podem ser expressos usando diferentes tipos, seu valor deve ser associado a um tipo no qual são descritos. Por exemplo, os objetivos podem ser modelados usando uma lógica temporal linear em tempo real (Letier, 2002), e a lógica modal pode ser usada para dar significado a conceitos como crença e conhecimento (Weiss, 1999).

A fim de representar objetivos e crenças, um estereótipo chamado `<<goal>>` e um estereótipo chamado `<<belief>>` foram definidos com base na metaclasse *Property*. Uma propriedade de uma classe é um atributo (UML, 2004). O

estereótipo `<<belief>>` é uma simples extensão de *Property*. Não especifica qualquer restrição ou valor de *tag*. Apenas identifica os atributos que são *crenças*. O estereótipo `<<goal>>` descreve uma *tag* e uma restrição. A *tag planTag* liga o objetivo a um conjunto de *planos* que alcançam o *objetivo*. A restrição declara que o estereótipo `<<goal>>` é definido por um agente, uma organização, um ambiente ativo ou por um papel do agente. Quando um agente, uma organização ou um ambiente ativo define um objetivo, a *tag planTag* é associada a um conjunto de planos, mas quando um papel do agente define um objetivo, a *planTag* deve estar vazia, uma vez que papéis não definem planos.

Ação

As *Ações* são características comportamentais de agentes. Entretanto, não podem ser definidas como um estereótipo com base na metaclasse *Operation*, uma vez que a definição de *ações* e *operações* é diferente. Uma *operação* pode ser implementada por um *método* que pode ser chamado/solicitado por um objeto. As ações associadas a agentes nunca são chamadas por outro agente, mas apenas executadas sob o controle do próprio agente. Uma entidade nunca solicita a um agente que execute uma determinada ação. Os agentes interagem enviando e recebendo mensagens e não chamando a execução de ações. Durante a execução de uma ação, um agente pode, por exemplo, enviar e receber mensagens, chamar métodos, alterar papéis e mudar seu estado mental.

Além disso, não usamos a metaclasse *Action* definida em UML para representar ações de agente porque a metaclasse não estende a metaclasse *BehavioralFeature* e, portanto, não pode ser descrita como uma característica de um *Classifier*. Criamos uma nova metaclasse chamada *AgentAction* que estende a metaclasse *BehavioralFeature* a fim de representar as ações executadas por agentes. Semelhante à metaclasse *Operation*, a nova metaclasse *AgentAction* está associada à metaclasse *Constraints* a fim de definir as precondições e as pós-condições. Uma restrição é uma condição expressa em um texto de linguagem natural ou em uma linguagem legível pela máquina para fins de declaração de algumas semânticas de uma entidade (UML, 2004).

As precondições de uma ação definem as condições que devem ser verdadeiras quando a ação é chamada. Essas precondições são pressupostas por uma implementação dessa ação. As pós-condições para uma ação definem as condições que serão verdadeiras quando a execução da ação é concluída com

sucesso, pressupondo que as precondições foram satisfeitas. Essas pós-condições devem ser satisfeitas por qualquer implementação da ação.

Plano

As características comportamentais de agentes também são compostas por seus planos. Uma nova metaclasse chamada *AgentPlan* foi criada para especificar os planos das características do *agente* em UML. Um *plano* está associado a *objetivo* e é representado por uma seqüência de *ações* executada por um *agente* para alcançar o *objetivo*.

Foi criada uma nova metaclasse a fim de representar planos, porque não há no metamodelo de UML metaclasses com o mesmo significado de *plano*. A metaclasse *AgentPlan* é uma especialização da metaclasse *BehavioralFeature*. Não foi descrito um estereótipo para descrever *plan* com base na definição da metaclasse *AgentAction* porque um *plan* e uma *action* não compartilham as mesmas propriedades e relacionamentos. A Figura 12 ilustra o metamodelo de UML estendido a fim de incorporar a metaclasse *AgentClass* e as metaclasses e estereótipos que representam propriedades do agente.

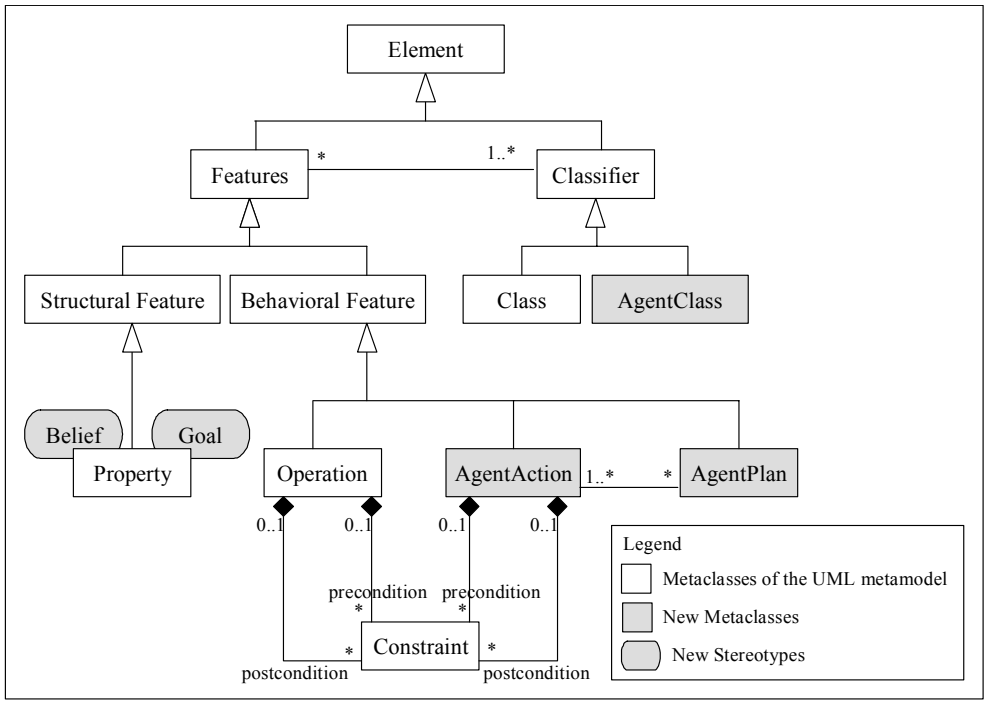


Figura 12 – O metamodelo de UML estendido para incorporar agentes e suas propriedades.

4.2.1.2. Associações de Agentes

Os agentes exercem pelo menos um papel em uma organização. Um agente pode exercer mais de um papel em diferentes organizações, mas um papel do agente não pode ser exercido por mais de um agente. Com o objetivo de representar agentes que exercem papéis em organizações, a metaclasse *AgentClass* está associada a metaclasses que descrevem os papéis do agente e as organizações. Ademais, a metaclasse *AgentClass* também é associada à metaclasse que representa ambientes porque cada agente deve residir em um ambiente. O conjunto de associações entre a metaclasse *AgentClass* e outras metaclasses é ilustrado na Figura 13.

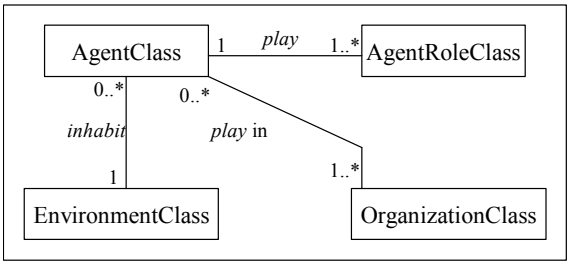


Figura 13 – As associações entre *AgentClass* e outras metaclasses.

4.2.2. Papel de Objeto

Um *object role* guia e restringe o comportamento de um objeto, uma vez que manipula o objeto e descreve um conjunto de características vistas por outras entidades. Um papel de objeto pode restringir suas características, mas também pode adicionar informações (atributos) e comportamento (métodos) ao objeto que exerce o papel.

Para estar em conformidade com essas idéias, temos duas opções: criar uma nova metaclasse ou adaptar a metaclasse *ConnectableEntity* definida em UML. A metaclasse *ConnectableEntity* precisaria ser adaptada porque não descreve as propriedades (ou características) vistas por outros objetos. Uma metaclasse não descreve as propriedades dos objetos restringidas pelo papel de objeto ou as propriedades adicionadas ao conjunto de propriedades do objeto.

Propomos criar uma nova metaclasse *ObjectRoleClass* porque estamos interessados em descrever uma extensão conservativa (Turski et al., 1987) para o metamodelo de UML. Essa nova metaclasse é uma especialização de *Classifier*.

Então, a *ObjectRoleClass* é associada às metaclasses *BehavioralFeature* e *StructuralFeature*.

4.2.2.1. Propriedades de Papel de Objeto

As características de *ObjectRoleClass* representam a visão que outras entidades têm do objeto que está exercendo o papel. A metaclasse *ObjectRoleClass* pode restringir o acesso a atributos (*StructuralFetures*) e métodos (*BehavioralFeatures*) descritos na classe do objeto e também pode adicionar novos atributos e métodos à classe. A Figura 14 ilustra o metamodelo de UML estendido a fim de incorporar a metaclasse *ObjectRoleClass*.

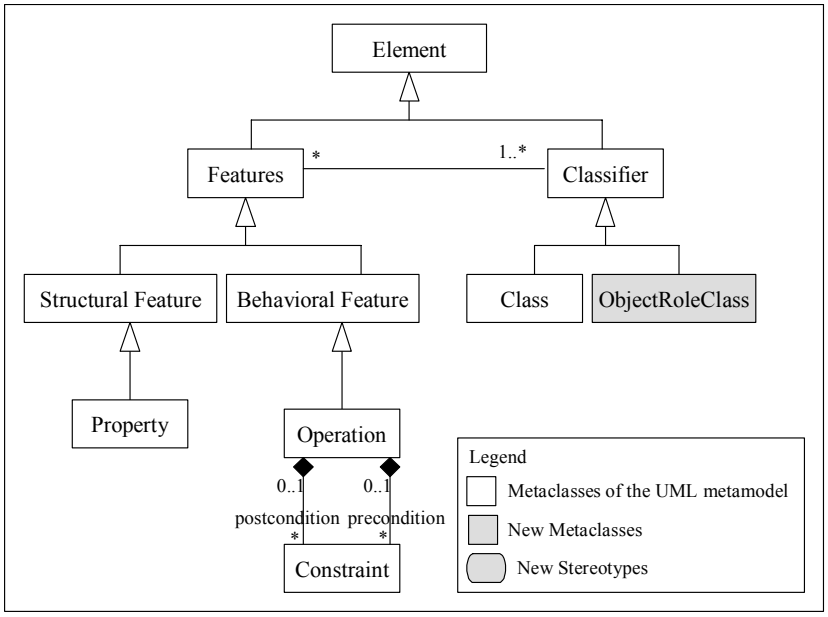


Figura 14 – O metamodelo de UML estendido para incorporar papéis de objeto.

4.2.2.2. Associações de Papel de Objeto

Os papéis de objeto são *definidos* no contexto de uma organização e são exercidos por objetos. Um objeto pode *exercer* mais de um papel em diferentes organizações, mas um papel de objeto só pode ser exercido por um objeto. A fim de representar os relacionamentos entre papéis de objeto, organizações e objetos, são fornecidas duas associações que ligam a metaclasse *ObjectRoleClass* e as metaclasses que representam organizações e objetos. O conjunto de associações entre a metaclasse e outras metaclasses é ilustrado na Figura 15.

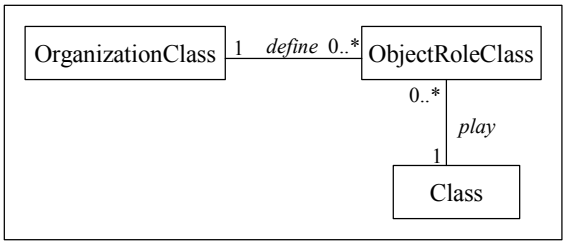


Figura 15 – As associações entre *ObjectRoleClass* e outras metaclasses.

4.2.3. Papel de Agente

Um *papel de agente* está interessado na orientação e na restrição do comportamento autônomo de um *agente* (ou uma organização) ao descrever os objetivos de um agente (ou organização), outras crenças, as ações que um agente (ou organização) deve realizar e as ações que um agente (ou organização) pode realizar enquanto está exercendo o papel.

Para ilustrar melhor a diferença entre o papel exercido por um objeto e o papel exercido por um agente, criamos uma metaclasse chamada *AgentRoleClass* a fim de representar os papéis exercidos por agentes. A *AgentRoleClass* estende a *Classifier* e, então, descreve as características estruturais e comportamentais de papéis de agentes.

4.2.3.1. Propriedades dos Papéis de Agente

As características estruturais da metaclasse *AgentRoleClass* são os *objetivos* e as *crenças* relacionados a papéis de agente. As crenças descritas pelo papel de agente são adicionadas às crenças do agente. Os objetivos descritos pelo papel de agente podem adicionar novos objetivos do agente ou podem estar relacionados a um subconjunto de objetivos do agente. As características comportamentais da metaclasse *AgentRoleClass* descrevem os *deveres, direitos* e *protocolos* relacionados a papéis de agente. Observe que a *AgentRoleClass* não adiciona características comportamentais ao *agente* que está exercendo o *papel*, mas restringe suas *ações* ao descrever *deveres, direitos* e *protocolos*.

Dever e Direito

Dever e *direito* são definidos como estereótipos da metaclasse *AgentAction*. O estereótipo `<<duty>>` identifica *ações* que *agentes* que estão exercendo o *papel* devem realizar, e o estereótipo `<<right>>` identifica as *ações* que os *agentes* que

estão exercendo *papel* têm permissão de realizar. Os deveres e os direitos não descrevem a implementação das ações, mas descrevem sua assinatura.

Protocolo

Uma nova metaclasse chamada *AgentProtocol* foi criada para descrever os *protocolos* aos quais o *agente* deve obedecer ao exercer o *papel* e interagir com outras entidades do sistema. A metaclasse *AgentProtocol* especializa a metaclasse *BehaviorFeature* porque faz parte do comportamento de *papéis do agente*. Um *protocolo* de agente define o conjunto de *mensagens* que um *agente* pode enviar a outro *agente* em uma interação e as *mensagens* que pode receber de outros agentes.

Mensagem

A fim de definir completamente *protocolos*, é importante definir o que é uma *mensagem* enviada e recebida por um agente. As *mensagens* enviadas e recebidas por um *agente* são diferentes das *mensagens* enviadas e recebidas por um *objeto*. O metamodelo de UML define uma mensagem enviada e recebida por objetos como um tipo específico de comunicação em uma interação. Uma comunicação pode emitir um sinal, chamando uma *Operação* ou criando e destruindo uma *instância* (UML, 2004). Assim, uma *mensagem* enviada e recebida por um *objeto* pode estar relacionada a uma *operação* e, conseqüentemente, também relacionada a um *método* executado pelo *objeto* que recebe a *mensagem*. Conforme visto anteriormente, uma *mensagem* enviada e recebida por um *agente* nunca está conectada às *ações* executadas pelo agente que recebe a *mensagem*. Um *objeto* pode chamar *métodos* de outro *objeto* enviando *mensagens* a ele, mas um *agente* não pode chamar a execução de *ações* de outro *agente* que está enviando uma *mensagem* a ele.

A fim de tornar explícita a diferença entre *mensagens de objetos* definidas pelo metamodelo de UML e as *mensagens de agentes*, definimos uma nova metaclasse chamada *AgentMessage* para representar as *mensagens* específicas em protocolos. As características relacionadas a uma *mensagem* são a marca que especifica o tipo de mensagem, o conteúdo que pode ser vazio, o emissor e o destinatário da mensagem (Caire et al., 2002). A Figura 16 ilustra o metamodelo de UML estendido a fim de incorporar a metaclasse *AgentRoleClass* e as metaclasses e os estereótipos que representam as propriedades do papel do agente.

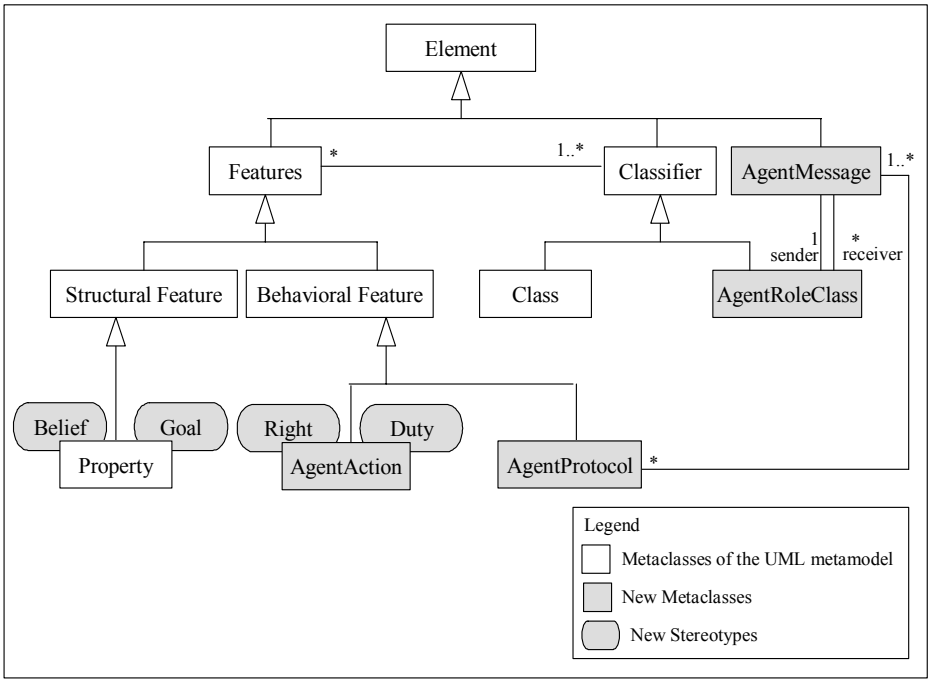


Figura 16 – O metamodelo de UML estendido para incorporar papéis do agente e suas propriedades.

4.2.3.2. Associações de Papéis de Agente

Os papéis de agente são *definidos* no contexto de uma organização e *são exercidos* por um agente ou uma suborganização. A partir da perspectiva de entidades fora de uma organização, suborganizações podem ser vistas como agentes que *exercem* papéis na organização em que são definidos. Com o objetivo de representar esses relacionamentos, a metaclasse *AgentRoleClass* está associada a metaclasses que representam agentes e organizações. O conjunto de associações entre a metaclasse *AgentRoleClass* e outras metaclasses está ilustrado na Figura 17.

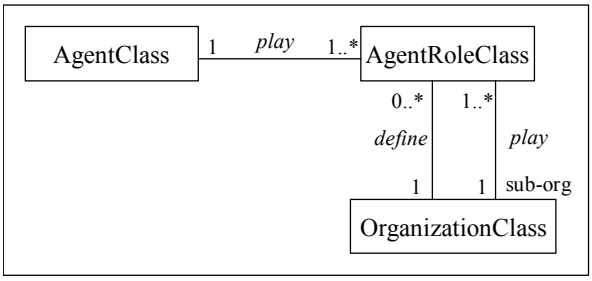


Figura 17 – As associações entre *AgentRoleClass* e outras metaclasses.

4.2.4. Organização

Apesar de termos definido em TAO que uma *organização* estende as propriedades e os relacionamentos definidos pelos *agentes*, não criamos um estereótipo `<<organization>>` associado à metaclasse *AgentClass*. As organizações estendem a noção de agentes porém descrevem outras propriedades e relacionamentos. Uma *organização* define um conjunto de *axiomas* (*regras* e *leis*) ao qual *agentes* e *suborganizações* devem obedecer. Os *axiomas* caracterizam as restrições globais da *organização*. Além disso, uma *organização* também define os *papéis* que devem ser exercidos pelos *agentes* e pelas *suborganizações* dentro dela. Para representar a *organização*, foi criada uma nova metaclasse chamada *OrganizationClass*. Essa metaclasse especifica a metaclasse *AgentClass* proposta na Seção 4.2.1.

4.2.4.1. Propriedades da Organização

Como as organizações são extensões de agentes, elas têm *objetivos*, *crenças*, *ações* e *planos* e exercem *papéis* nas organizações em que são definidas. Os *objetivos*, *crenças* e *axiomas* da organização constituem as características estruturais da organização. Por sua vez, as *ações* e os *planos* são suas características comportamentais.

Axioma

A fim de representar os *axiomas* de uma organização, criamos o estereótipo `<<axiom>>` relacionado à metaclasse *Property*. Como objetivos e crenças, os axiomas também podem ser expressos como atributos. Um axioma é identificado por seu nome e possui um valor que descreve a si mesmo. Com o objetivo de interpretar o valor de um axioma, é importante associar um tipo a ele. A Figura 18 ilustra o metamodelo de UML estendido a fim de incorporar a metaclasse *OrganizationClass* e as metaclasses e os estereótipos que representam as propriedades da organização.

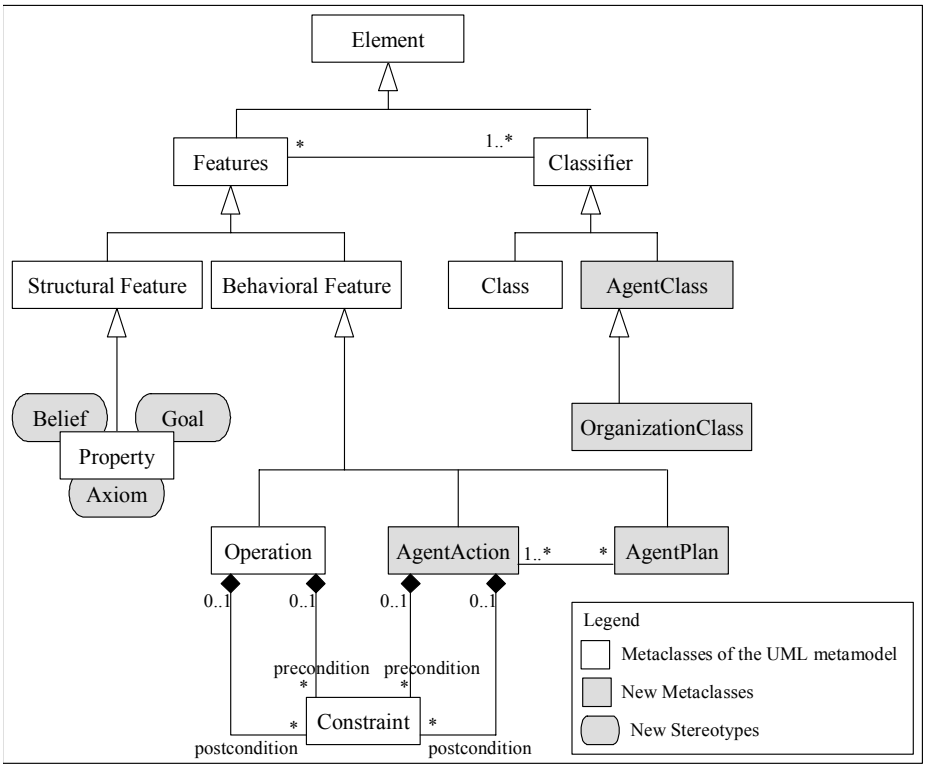


Figura 18 – O metamodelo de UML estendido para incorporar organização e suas propriedades.

4.2.4.2. Associações da Organização

As organizações *definem* os papéis do agente e os papéis de objeto. A fim de representar os relacionamentos independentes do domínio entre as organizações e os papéis definidos, foram definidas as associações que ligam a metaclasse *OrganizationClass* às metaclasses *AgentRoleClass* e *ObjectRoleClass*.

Qualquer organização pode *definir* várias suborganizações, e todas elas devem fazer parte de uma organização. Uma suborganização é uma organização que *exerce* um papel *em* outra organização. O relacionamento entre uma organização e suas suborganizações é representado pelo relacionamento *aggregation* que liga a metaclasse *OrganizationClass* a ela. O relacionamento entre uma suborganização e um papel do agente é representado por um relacionamento *association* ligando a metaclasse *OrganizationClass* e a metaclasse *AgentRoleClass*.

Os papéis *definidos* por uma organização são exercidos por agentes, suborganizações e objetos no contexto das organizações. Os relacionamentos entre organizações e os agentes e objetos que *exercem* papéis *nas* organizações

são representados associando a metaclasse *OrganizationClass* a *AgentClass* e associando a metaclasse *OrganizationClass* à metaclasse *Class*.

Como organizações residem em um ambiente, o relacionamento entre organizações e ambientes é representado associando a metaclasse *OrganizationClass* e a metaclasse *EnvironmentClass*. A Figura 19 ilustra o conjunto de associações relacionadas à metaclasse *OrganizationClass*.

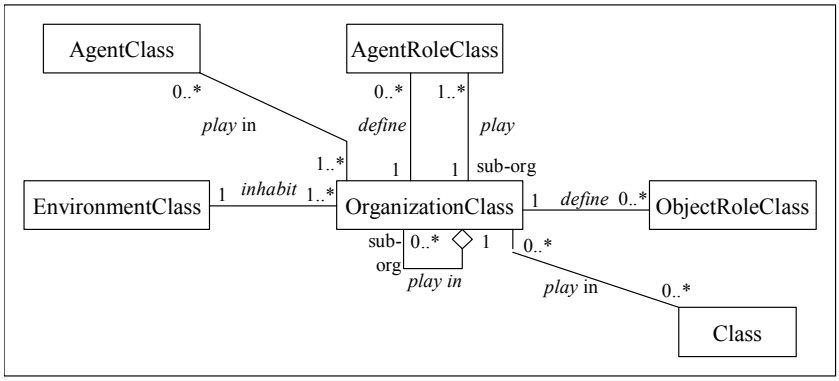


Figura 19 – As associações entre *OrganizationClass* e outras metaclasses.

4.2.5. Ambiente

Uma metaclasse abstrata chamada *EnvironmentClass* foi criada ao estender a metaclasse *Classifier* para representar ambientes. A metaclasse *EnvironemntClass* não define qualquer propriedade, uma vez que depende das características do ambiente. Um ambiente pode ser uma entidade ativa como um agente ou uma entidade passiva como um objeto tradicional. No entanto, a metaclasse define as associações entre ela e outras metaclasses da linguagem MAS-ML.

4.2.5.1. Propriedades do Ambiente

Apesar de os objetos e ambientes passivos compartilharem propriedades semelhantes, um ambiente passivo não pode ser definido ao criar um estereótipo com base na metaclasse *Class* porque os objetos e os ambientes passivos não definem os mesmos relacionamentos. Os objetos podem exercer os papéis em organizações, e ambientes são o habitat de objetos e organizações. Uma metaclasse chamada *PassiveEnvironmentClass* foi criada para representar ambientes passivos. Ela estende a metaclasse *EnvironmentClass*. As

características estruturais de ambientes passivos são *atributos* e suas características comportamentais são *métodos*.

Os agentes e ambientes ativos compartilham propriedades semelhantes. Entretanto, agentes e ambientes ativos não definem os mesmos relacionamentos e, portanto, o ambiente ativo não pode ser definido como um estereótipo com base na metaclasse *AgentClass*. Os agentes exercem os papéis em organizações, e os ambientes são o habitat de agentes e organizações. Uma metaclasse *ActiveEnvironmentClass* foi criada ao estender a metaclasse *EnvironmentClass* para representar ambientes ativos. As características estruturais de ambientes ativos são *objetivos* e *crenças*, e suas características comportamentais são *ações* e *planos*. As metaclasses que representam o ambiente são ilustradas na Figura 20.

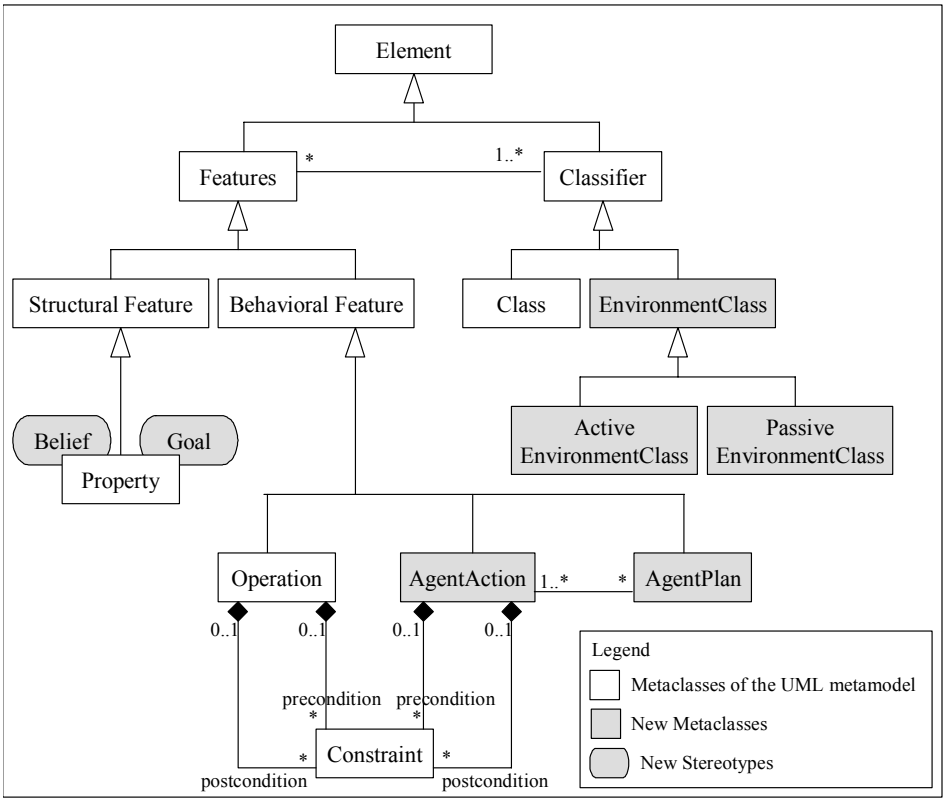


Figura 20 – O metamodelo de UML estendido para incorporar ambientes e suas propriedades.

4.2.5.2. Associações do Ambiente

Os ambientes são o habitat de objetos, agentes e organizações. Todos os agentes, as organizações e os objetos *residem* em um ambiente. Esses relacionamentos são independentes do domínio e, por isso, devem ser

representados no metamodelo de MAS-ML. As associações foram descritas para ligar a metaclasse abstrata *EnvironmentClass* às metaclasses que definem agentes, organizações e objetos, conforme ilustrado na Figura 21.

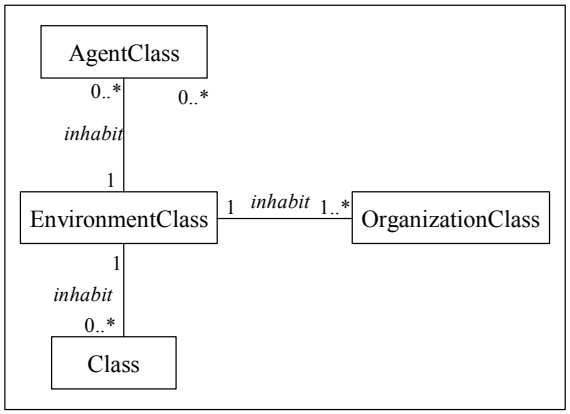


Figura 21 – As associações entre *EnvironmentClass* e outras metaclasses.

A Figura 22 apresenta um subconjunto de metaclasses do metamodelo de UML e as extensões feitas pela MAS-ML. Essa figura mostra as novas metaclasses e os novos estereótipos propostos pela MAS-ML relacionados às entidades e propriedades descritas em TAO. Os ícones que representam os estereótipos estão associados às metaclasses nas quais os estereótipos se baseavam.

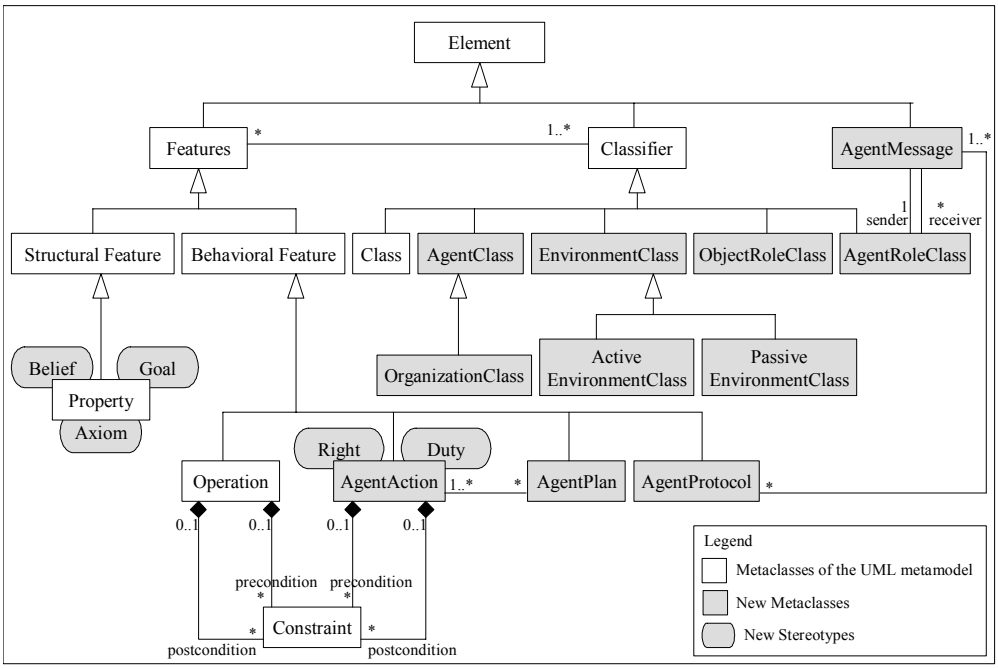


Figura 22 – O metamodelo de UML estendido para incorporar as entidades da MAS-ML e suas propriedades.

A Figura 23 ilustra as associações entre as metaclasses que representam as entidades de TAO. Conforme mencionado anteriormente, essas associações foram definidas com base em relacionamentos independentes do domínio (*inhabit*, *play* e *ownership*) definidos no TAO.

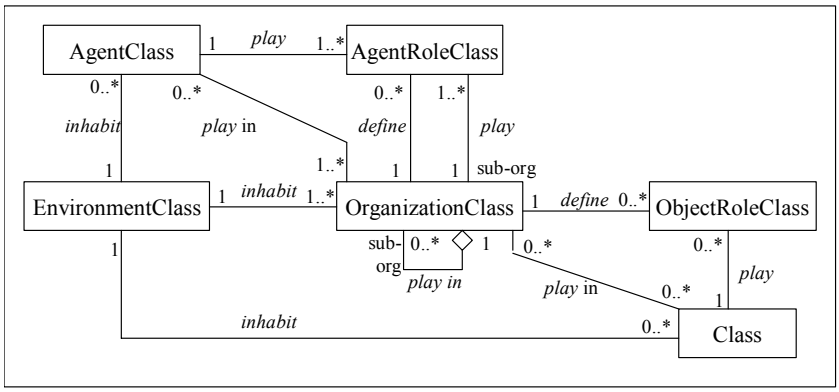


Figura 23 - As associações entre as metaclasses que representam as entidades do TAO.

4.2.6. O Relacionamento *Association*

O relacionamento *association* de UML é definido pela metaclasse *Association* que estende a metaclasse abstrata *Relationship*. Em UML, o relacionamento *association* é usado entre classes de objeto. Esse relacionamento foi estendido para também ser usado entre classes de papel (papéis de objeto e papéis do agente), entre classes do ambiente, entre classes do agente e classes de objeto, entre classes de organização e classes de objeto e entre classes de papel e classes de objeto. Como o relacionamento *association* já foi definido em UML, não foi preciso criar uma nova metaclasse para representar esse relacionamento.

4.2.7. O Relacionamento *Aggregation*

A UML não define uma metaclasse para representar explicitamente o relacionamento *aggregation*. *Aggregation*s são definidas como uma associação com características particulares. Um relacionamento *aggregation* define um relacionamento total/parcial entre as entidades que participam do relacionamento. Ele é aplicado para ligar classes de objeto em UML e foi estendido para também ser aplicado em papéis de objeto e papéis do agente. No metamodelo de MAS-ML, o relacionamento *aggregation* também é definido com base na metaclasse

que representa o relacionamento *association*. Uma nova metaclasse não foi criada para representar *aggregation*s.

4.2.8. O Relacionamento *Generalization*

A metaclasse *Generalization* define o relacionamento *specialization*/ *generalization* definido em UML. Essa metaclasse estende a metaclasse abstrata *DirectedRelationship* que estende a metaclasse abstrata *Relationship*. Um relacionamento direcionado faz referência a uma ou mais entidades de origem e uma ou mais entidades de destino. Em um relacionamento direcionado, as entidades de origem e destino possuem uma semântica diferente.

Uma generalização é um relacionamento taxonômico entre um classificador mais geral e um mais específico. Cada instância do classificador específico também é uma instância indireta do classificador geral (UML, 2004). Um relacionamento *specialization*/*generalization* usado normalmente entre classes de objeto, foi estendido também para ser usado entre classes do agente, classes de organização, classes de papel de objeto, classes de papel do agente e classes de ambiente.

4.2.9. O Relacionamento *Dependency*

O relacionamento *dependency* de UML é definido pela metaclasse *Dependency* que estende a *DirectedRelationship*. Uma *dependency* é um relacionamento que significa que uma única entidade ou um conjunto de entidades de modelo (clientes) requer outras entidades de modelo (fornecedores) para sua especificação ou implementação. Isso significa que a semântica completa das entidades dependentes é semântica ou estruturalmente dependente da definição da(s) entidade(s) do fornecedor (UML, 2004). Um relacionamento *dependency* foi estendido para ser usado entre classes de objeto, classes de papel do agente, classes de papel de objeto e entre classes de papel do agente e classes de papel de objeto.

4.2.10. O Relacionamento *Inhabit*

Uma nova metaclasse chamada *Inhabit* foi criada para representar o relacionamento *inhabit*. Essa metaclasse estende a metaclasse

DirectedRelationship. O relacionamento *inhabit* está relacionado ao habitat e às entidades que residem (os cidadãos do) no habitat. Esse relacionamento pode ser aplicado às classes de ambiente e classes do agente, a classes de ambiente e classes de objeto e a classes de ambiente e classes de organização. Quando um ambiente está relacionado a uma classe de entidade (classe do agente, de objeto ou organização) pelo relacionamento *inhabit*, isso significa que todas as instâncias de entidades estão residindo nas instâncias de ambiente dessa classe. Observe que todas as instâncias de entidades devem residir em uma instância de ambiente.

4.2.11. O Relacionamento *Ownership*

A nova metaclasse *Ownership* que estende a metaclasse *DirectedRelationship* foi criada para representar o relacionamento *ownership*. A *ownership* especifica que uma entidade (o membro) é definida no escopo de outra entidade (o proprietário) e que um membro deve obedecer a um conjunto de restrições globais definidas por seu proprietário. Um relacionamento *ownership* pode ser aplicado a classes de organização e classes de papel de objeto em que as organizações são as proprietárias e os papéis são os membros. Quando uma classe do papel está relacionada a uma classe de organização pelo relacionamento *ownership*, isso significa que as instâncias de papel fazem parte das instâncias de organização. Observe que uma instância de papel pode fazer parte de apenas uma instância de organização, uma vez que uma instância de papel não pode ser exercida em mais de uma organização.

4.2.12. O Relacionamento *Play*

Para representar o relacionamento *play*, foi criada a metaclasse *Play*. Ela estende a *DirectedRelationship*. O relacionamento *play* especifica os papéis que podem ser exercidos por uma entidade. Ele pode ser aplicado a classes do agente e classes de papel do agente, classes de organização e classes de papel do agente e classes de objeto e classes de papel de objeto. Quando uma classe de entidade está relacionada a uma classe do papel pelo relacionamento *play*, isso significa que a instância de entidade pode executar instâncias de papel.

4.2.13.
O Relacionamento *Control*

O relacionamento *control* é representado pela nova metaclasse *Control* estendendo a *DirectedRelationship*. Ele define que a entidade controlada deve fazer tudo que a entidade do controlador pedir. Este relacionamento se aplica a papéis de agente. Uma instância de papel de agente pode ser o controlador de outra instância de papel de agente.

A Figura 24 mostra as novas metaclasses relacionadas aos relacionamentos descritos em TAO propostos pela MAS-ML ao metamodelo de UML.

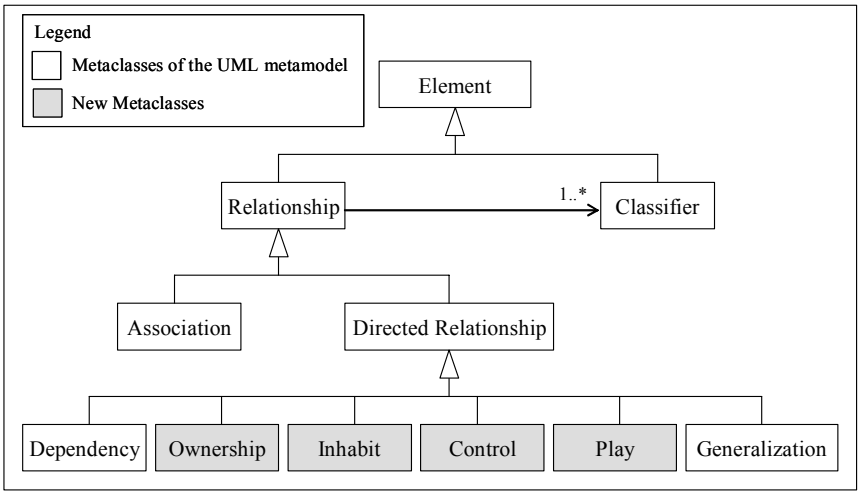


Figura 24 – O metamodelo de UML estendido para incorporar os relacionamento MAS-ML.

4.3.
Os Diagramas Estruturais da MAS-ML

Ao introduzir novas abstrações no metamodelo de UML, precisamos criar novos elementos de diagrama para representar as seis novas entidades e os oito relacionamentos (Seção 4.3.1). A princípio, como TAO define um conjunto de seis entidades e oito relacionamentos diferentes, precisamos de quinze novos elementos de digrama para representá-los. Felizmente, uma entidade e alguns relacionamentos definidos em TAO também são apresentados no metamodelo de UML.

Devido ao conjunto de diferentes entidades e relacionamentos definido no metamodelo de TAO que foi incorporado ao metamodelo de UML, propomos diferentes diagramas estruturais cujo foco está nos diversos aspectos de extensões

que devem ser cobertos pela MAS-ML resultante. Os diagramas estruturais que compõem a MAS-ML são o diagrama de classes estendido e dois novos diagramas chamados diagrama de organização e diagrama de papel (Seções 4.3.2, 4.3.3 e 4.3.4). Os três diagramas estruturais – diagrama de Classes, Organização e Papel – mostram todas as entidades e todos os relacionamentos definidos em TAO.

4.3.1. Elementos de Diagramas Estruturais

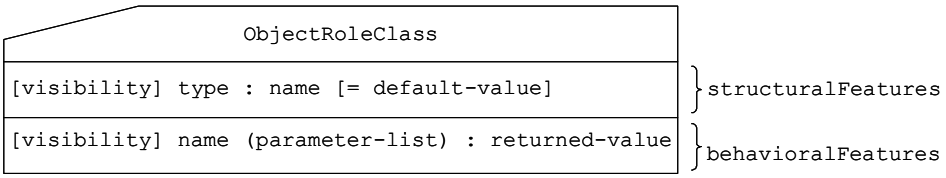
A definição de TAO da entidade objeto/classe é semelhante à definição da metaclasse *Class* de UML. Assim, não criamos um novo elemento de diagrama para representar essa entidade. Associamos novos elementos de diagrama às metaclasses *AgentClass*, *OrganizationClass*, *EnvironmentClass*, *ObjectRoleClass* e *AgentRoleClass*. Todos os elementos de diagrama da MAS-ML que representam as entidades da mesma foram definidos com três compartimentos separados por linhas horizontais. O compartimento superior possui o nome da classe; o compartimento intermediário tem as características estruturais, e o inferior contém as características comportamentais da entidade. Os compartimentos intermediários e o inferior podem ser omitidos, quando necessário.

Os relacionamentos *association*, *specialization*, *aggregation* e *dependency* descritos em TAO também são descritos em UML com uma semântica semelhante. A diferença entre os relacionamentos definidos em TAO e aqueles definidos em UML são as entidades que participam do relacionamento. Apesar de TAO aplicar esses relacionamentos a entidades como organização, agente e papel do agente, nós não definimos novos elementos de diagrama para representar os relacionamentos. Criamos novos elementos de diagrama e os associamos aos novos relacionamentos definidos em TAO que não existem em UML. Esses relacionamentos são *inhabit*, *ownership*, *play* e *control*.

4.3.1.1. ObjectRoleClass

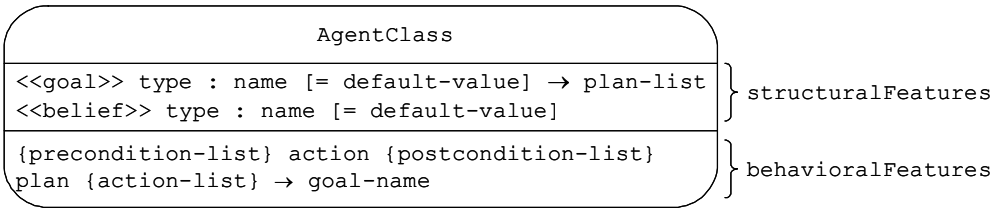
Uma *ObjectRoleClass* é mostrada como um retângulo sólido com um ângulo no canto esquerdo. O compartimento superior contém o nome da classe do papel que deve ser único em seu espaço de nome incluído. O compartimento intermediário contém a lista de atributos e a inferior, uma lista de métodos.

A notação usada para descrever os atributos e os métodos de uma classe do papel de objeto é equivalente à notação usada para descrever os atributos e os métodos de uma classe. Um atributo é basicamente descrito por sua visibilidade, nome, tipo e valor padrão (que pode ser omitido) e um método descrito por sua visibilidade, nome, lista de parâmetros de entrada e valor retornado.



4.3.1.2. AgentClass

Uma *AgentClass* é mostrada como um retângulo com cantos arredondados com três compartimentos separados por linhas horizontais. O compartimento superior contém o nome da classe do agente que deve ser único em seu espaço de nome incluído. O compartimento intermediário contém a lista de objetivos e crenças do agente e o inferior, uma lista de ações e planos.



Como objetivos e crenças são estereótipos de *Property* e, por isso, podem ser descritos como um atributo, suas notações são semelhantes à notação de um atributo. Objetivos e crenças são descritos por seus nomes, tipos e valores padrão (que podem ser omitidos). Ademais, um objetivo também identifica os planos que alcançam o objetivo. A característica visibilidade associada a *Properties* não se aplica a objetivos e crenças, pois não podem ser acessados por outras entidades, só pelo agente que os define.

No caso de um objetivo ser subdividido em subobjetivos, estes devem ser descritos. Um subobjetivo também é descrito como um atributo e está associado aos planos que alcançam os subobjetivos. Como o objetivo é alcançado ao executar os planos associados aos subobjetivos, os planos precisam ser associados ao objetivo quando é composto por subobjetivos. A caixa a seguir ilustra a definição de subobjetivos.

```
<<goal>> type : name [= default-value] [→ plan-list] {
 <<sub-goal>> type : name [= default-value] → plan-list
 <<sub-goal>> type : name [= default-value] → plan-list
 ...
}
```

Uma ação é expressa como uma lista de precondições, sua assinatura (nome) e uma lista de pós-condições. As precondições e as pós-condições de um ação são descritas entre chaves{ } por seu tipo, nome e valor. Uma ação pode não ter precondições ou pós-condições e, portanto, as chaves podem estar vazias. Um plano é definido por um nome, uma lista de nomes de ações e o objetivo que o plano alcança.

A fim de exemplificar a definição de *crenças*, suponha que um agente possua uma crença que define o preço máximo do item que ele pretende comprar. Nesse exemplo, a crença é expressa usando a lógica do modelo e o conceito *Bel* primitivo cognitivo apresentado em (Weiss, 1999). A crença é ilustrada a seguir:

```
<<belief>> ModalLogic : MaximumPrice = "Bel(MaximumPrice=$50,00)"
```

Um *objetivo* pode ser exemplificado usando lógica temporal. Suponha que o agente tenha um objetivo de comprar um item se o seu preço for igual ou menor do que o preço máximo descrito em sua crença. O objetivo "toBuy" é associado ao plano "negotiate". Nesse exemplo, o objetivo é expresso usando lógica temporal (Letier et al., 2002), conforme ilustrado na caixa a seguir:

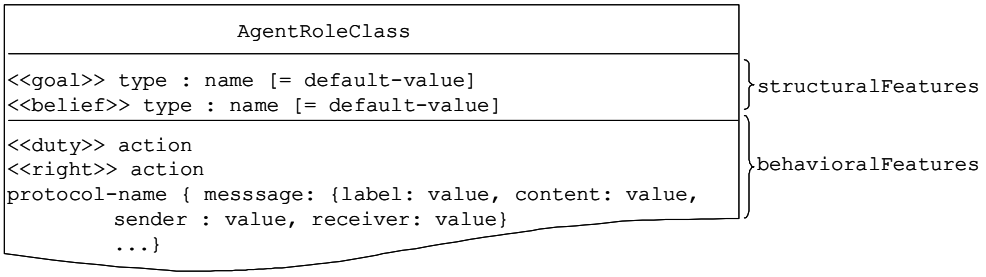
```
<<goal>> LinearTemporalLogic : toBuy =
     "ItemPrice = MaximumPrice => o ToBuyItem = true" → Negotiate
```

Suponha que o *plano* "negotiate" é composto pelas *ações* "send_proposal", "evaluate_conter_proposal" e "send_answer". A ação "send_proposal" e o plano "negotiate" são ilustrados na caixa a seguir:

```
{} Send_proposal {boolean : proposal_sent = true}
...
Negotiate {Send_proposal, Evaluate_conter_proposal, Send_answer}
                                                    → toBuy
```

4.3.1.3. AgentRoleClass

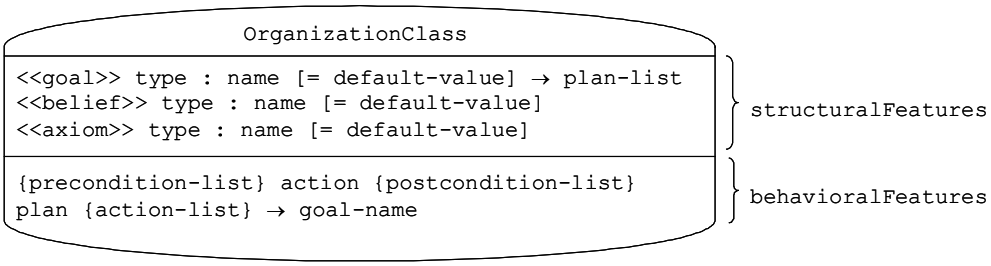
Uma *AgentRoleClass* é representada por um retângulo sólido com uma curva na parte inferior. Ela possui três compartimentos separados por linhas horizontais. O compartimento superior contém o nome de papel do agente que deve ser único em seu espaço de nome incluído. O compartimento intermediário de lista contém a lista de objetivos e crenças associados ao papel, e o inferior, uma lista de deveres, direitos e protocolos.



Os objetivos e as crenças associados a uma classe do papel do agente são descritos por seu tipo, nome e valor padrão. Observe que os objetivos associados aos papéis do agente não identificam planos. Os deveres e os direitos são descritos pelo nome da ação associada a eles. Um protocolo é descrito por seu nome e uma lista de mensagens. Uma mensagem relacionada a um protocolo descreve o rótulo, o tipo de conteúdo, o emissor e o destinatário da mensagem.

4.3.1.4. OrganizationClass

Uma *OrganizationClass* é mostrada na forma de um losango com três compartimentos separados por linhas horizontais. O compartimento superior contém o nome da classe da organização que deve ser único em seu espaço de nome incluído. O compartimento intermediário de lista contém a lista de objetivos, crenças e axiomas da organização, e o inferior, uma lista de ações e planos da mesma. Esses dois compartimentos podem ser omitidos, quando necessário.



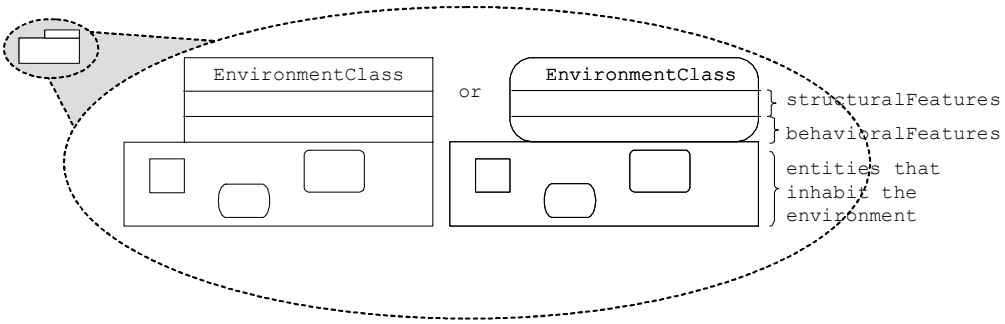
A descrição dos objetivos, crenças, ações e planos relacionados às organizações é equivalente aos objetivos, crenças, ações e planos relacionados a agentes. Como os axiomas são estereótipos de *Property*, um axioma é descrito por seu tipo, nome e valor padrão. Observe que os axiomas de uma organização devem ser acessados por todas as entidades na organização e, assim, não há a necessidade de associar a característica de visibilidade a elas.

A fim de identificar a classe da organização que gera as instâncias da organização principal, o estereótipo `<<main_organization>>` deve ser associado à

classe da organização. Usando esse estereótipo, fica claro que deve haver apenas uma instância dessa classe no sistema e que essa instância é a organização principal.

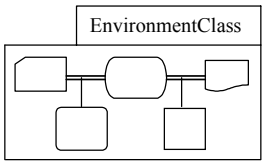
4.3.1.5. EnvironmentClass

Uma *EnvironmentClass* é mostrada como um pacote que reúne todas as entidades que residem no ambiente. O ícone do pacote possui dois compartimentos. O inferior contém as entidades que residem no ambiente, e o superior, a especificação do mesmo. O compartimento superior pode ser modelado como um retângulo (usando a forma de uma classe) ou como um retângulo com cantos arredondados (usando a forma de uma classe do agente). Ele pode ser subdividido em três subcompartimentos separados por linhas horizontais. O compartimento superior contém o nome da classe do ambiente que deve ser único em seu espaço de nome incluído. O compartimento intermediário de lista contém a lista de características estruturais do ambiente, enquanto o inferior contém a lista de características comportamentais do ambiente. Se o ambiente for um objeto, suas características comportamentais são métodos, e suas características estruturais são atributos. Se ele for um agente, suas características comportamentais são ações e planos, e suas características estruturais são objetivos e crenças.



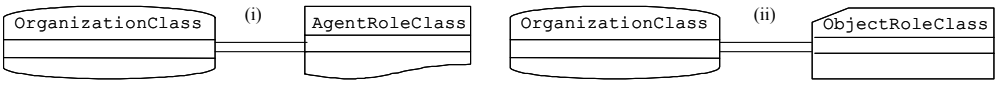
4.3.1.6. O Relacionamento *Inhabit*

O relacionamento *inhabit* aplica-se a *EnvironmentClass* e seus cidadãos *AgentClass*, *Class* e *OrganizationClass*. O relacionamento é mostrado inserindo o cidadão no compartimento inferior definido pelo elemento de diagrama associado a *EnvironmentClass*.



4.3.1.7. O Relacionamento *Ownership*

Esse relacionamento liga (i) *OrganizationClass* e *AgentRoleClass* e (ii) *OrganizationClass* e *ObjectRoleClass*. Esse relacionamento é mostrado como uma linha dupla que liga o proprietário e o membro. Além disso, as multiplicidades associadas à classe do papel do agente e às classes de papel de objeto podem ser definidas. A multiplicidade indica o número de instâncias de papel que podem ser exercidas em uma instância de organização. Entretanto, a multiplicidade associada à classe de organização é sempre uma, porque uma instância de papel não pode ser exercida em mais de uma instância de organização.

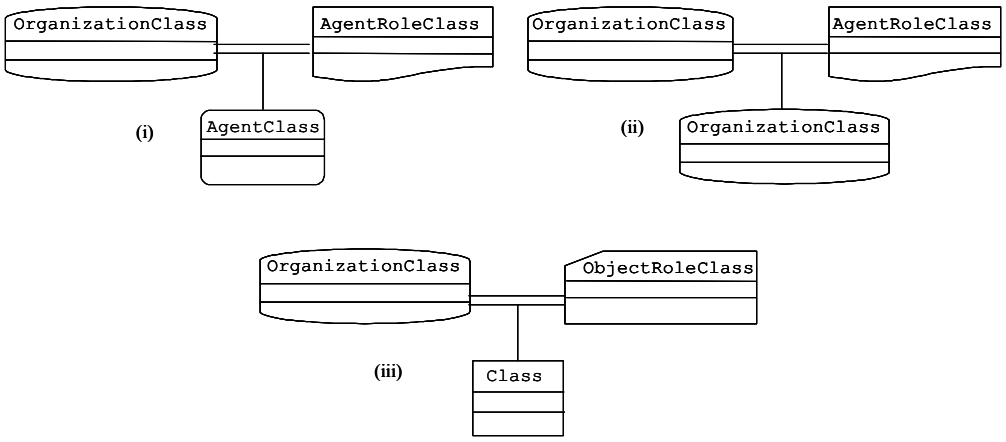


4.3.1.8. O Relacionamento *Play*

O relacionamento *play* associa um objeto, um agente ou uma suborganização a um papel definido em uma organização. Ele identifica uma identidade, um papel e uma organização em que a entidade exerce o papel. Baseia-se no relacionamento *ownership* que identifica uma organização e o papel que pode ser exercido nela. Como diferentes organizações podem definir o mesmo tipo de papel, o relacionamento *play* não seria totalmente definido se usássemos uma ligação simples entre um agente e um papel.

O relacionamento *play* liga (i) uma *AgentClass* ao relacionamento entre uma *OrganizationClass* e uma *AgentRoleClass*, (ii) uma *OrganizationClass* ao relacionamento entre uma *OrganizationClass* e uma *AgentRoleClass* e (iii) uma *Class* ao relacionamento entre uma *OrganizationClass* e uma *ObjectRoleClass*. Ele é mostrado como uma linha simples que liga (i) uma *AgentClass*, (ii) uma *OrganizationClass* ou (iii) uma *Class* à linha dupla que descreve o relacionamento *ownership* que identifica uma organização e um papel.

A fim de representar algumas instâncias de classe, organização ou agente que podem exercer papéis em organizações, podemos associar uma multiplicidade ao ponto que liga o relacionamento *play* e o relacionamento *ownership*. Por outro lado, uma multiplicidade não precisa ser associada à classe de entidade, pois uma instância de papel não pode ser exercida por mais de uma instância de entidade.



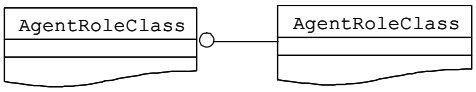
As classes de organização que exercem papéis em outras classes de organização são suborganizações da classe de organização que define o papel. Observe que uma organização não pode exercer papéis em suborganizações que estão exercendo um de seus papéis definidos. Por exemplo, se a classe de organização *OrgA* estiver ligada pelo relacionamento *play* à classe do papel do agente *R* e a classe do papel do agente *R* for definidas pela classe de organização *OrgB*, *OrgB* não poderá exercer papéis em *OrgA*.

4.3.1.9. O Relacionamento *Control*

Esse relacionamento especifica que uma *AgentRoleClass* pode controlar outra *AgentRoleClass*. O relacionamento *control* é mostrado como uma linha simples com um círculo em uma extremidade. O círculo indica a entidade do controlador.

As multiplicidades podem ser associadas ao controlador e às classes de entidades controladas. Uma multiplicidade associada à classe de entidades controladas indica o número de instâncias que podem ser controladas por uma instância do controlador. Por outro lado, uma multiplicidade associada à classe de

entidades do controlador indica o número de controladores que podem controlar uma instância controlada.



4.3.2. Diagrama de Classes

Um diagrama de classes de UML é um gráfico de entidades conectadas por seus relacionamentos estáticos. Observe que um digrama de "classes" também pode conter interfaces, pacote, relacionamentos e outras entidades de modelo. Um diagrama de classes é uma visão gráfica do modelo estrutural/estático.

Nosso principal objetivo com a extensão dos diagramas de classes é modificá-lo a fim de representar os relacionamentos entre as classes e outras entidades do SMA. O diagrama de classes estendido representa os relacionamentos entre as classes e os ambientes, classes e agentes e classes e organizações. Ademais, ele também foi estendido para representar os relacionamentos entre agentes, entre ambientes e entre organizações. As classes de entidade e os relacionamentos representados em diagramas de classes são ilustrados de acordo com os elementos de diagrama apresentados no Seção 4.3.1.

As classes que podem participar desse diagrama são classe do agente, classe de organização, classe de ambiente e outras classes definidas por UML. Os relacionamentos usados nesse diagrama são aqueles definidos por UML, mais os relacionamentos listados a seguir:

1. *inhabit* – usados entre classes e classes de ambiente;
2. *association* – usado entre classes do agente e classes, entre classes e classes de organização e entre classes de ambiente;
3. *specialization* – usado entre classes do agente, entre classes de organização e entre classes de ambiente.

Usando o diagrama de classes estendido, é possível modelar as classes que residem nos ambientes, as classes associadas a agentes e organizações, a associação entre classes de ambiente e a especialização entre agentes, organizações e ambientes.

4.3.3. Diagrama de Organização

O objetivo dos diagramas de organização é modelar todas as organizações de um sistema. Os diagramas de organização são responsáveis por modelar uma organização, ou seja, por modelar as propriedades da organização (objetivos, crenças, planos, ações e axiomas), os papéis definidos pela organização, as entidades (agentes, classes e suborganizações) que exercem esses papéis e o ambiente em que ela reside. Os relacionamentos *specialization* entre organizações são modelados em diagramas de classes.

Nos diagramas de organização, também é importante descrever as propriedades dos papéis (objetivos, crenças, deveres, direitos e protocolos) definidas na organização e as entidades que exercem cada papel. Os relacionamentos entre os papéis e entre os papéis e as classes são modelados nos diagramas de papel. Para cada agente, objeto e suborganização descritos nos diagramas de organização, é necessário definir suas propriedades. Contudo, seus relacionamentos são modelados em diagramas de classes.

As classes que podem ser usadas nos diagramas de organização são classe, classe de agente, classe de papel de agente, classe de papel de objeto, classe de organização e classe de ambiente. Os relacionamentos que podem ser usados são:

1. *ownership* – usado entre classes de organização e as classes de papel definidas pela organização;
2. *play* – usado entre classes de agente e classes de papel do agente, entre classes de suborganização e classes de papel do agente, e entre classes e classes de papel de objeto;
3. *inhabit* – usado entre as classes de ambiente e organização e classes de agente.

4.3.4. Diagrama de Papel

O diagrama de papel é responsável pela ilustração dos relacionamentos entre os papéis do agente e os papéis de objeto identificados nos diagramas de organização. Esse diagrama também identifica as classes acessadas pelos papéis de objeto e papéis do agente. As interações entre os agentes e as organizações do

sistema são descritas com base nos relacionamentos entre os papéis ilustrados nos diagramas de papel.

Um diagrama de papel pode mostrar os relacionamentos entre classe do papel do agente e classe do papel de objeto e entre esses papéis e classes. O conjunto de relacionamentos usado nesse diagrama é:

1. *control* – usado entre classes de papel de agente;
2. *dependency* – usado entre classes de papel de objeto, entre classes de papel de agente e classes de papel de objeto e entre classes de papel de agente;
3. *association* – usado entre classes de papel de objeto, entre classes de papel de agente e classes de papel de objeto, entre classes de papel de agente e entre classes e classes de papel;
4. *aggregation* – usado entre classes de papel de objeto e entre classes de papel de agente;
5. *specialization* – usado entre classes de papel de objeto e entre classes de papel de agente.

4.4. Diagramas Dinâmicos da MAS-ML

Propomos estender o diagrama de seqüência de UML para representar os aspectos dinâmicos de SMAs, ou seja, para representar as interações entre as instâncias do SMA e as intra-ações definidas por cada instância. Primeiro, ao estender o diagrama de seqüência de UML, precisamos definir novos *pathnames* e ícones com o objetivo de representar as instâncias de SMAs (agentes, organizações e ambientes) em diagramas de seqüência (Seção 4.4.1). Segundo, para representar as interações entre agentes, organizações, ambientes e objetos, a definição do conceito chamado *mensagem* usado em UML precisa ser estendida para representar entidades que estão enviando e recebendo mensagens e não estão chamando métodos e outras entidades. Agentes, organizações e ambientes ativos interagem enviando mensagens e não chamando métodos (Seção 4.4.2). Além disso, para representar a criação e a destruição de instâncias de SMAs e para representar a interação entre agentes, organizações, objetos e seus papéis, alguns estereótipos associados a mensagens foram redefinidos e outros foram criados (Seção 4.4.3). Ademais, o diagrama de seqüência foi estendido para representar a

execução de planos e ações enquanto modela as intra-ações relacionadas a agentes, organizações e ambientes (Seção 4.4.4). Finalmente, o diagrama de seqüência de UML também foi estendido para ilustrar os protocolos descritos pelos papéis do agente. Ao usar o diagrama estendido para modelar um protocolo, o diagrama representa os papéis do agente envolvidos no protocolo e a seqüência de mensagens definidas pelo protocolo (Seção 4.4.5).

4.4.1. Elementos de Diagramas Dinâmicos

Em UML, um diagrama de seqüência é representado pelo seu *pathname* e um ícone. UML define a estrutura de nomes a seguir, chamada *pathname*, associada a objetos:

```
simple pathname -> object : class
complete pathname -> object/role : class :: package
```

O *pathname* simples de um objeto descreve o nome do objeto e o nome da classe, separados por uma vírgula. O *pathname* completo descreve, à esquerda da vírgula, o objeto e o nome do papel do objeto na interação. À direita da vírgula, o *pathname* completo descreve o nome da classe e o nome do pacote da classe. O *pathname* associado a uma instância a identifica como uma entidade única no sistema. O *pathname* completo especifica uma instância que está participando de uma interação, porque descreve a instância ao identificar a classe na qual a instância se baseava e sua estrutura completa de pacotes.

4.4.1.1. *Pathname* do Objeto

TAO declara que um objeto pode interagir com outras entidades ao exercer papéis em organizações que residem em um ambiente. Se for o caso de o objeto estar exercendo um papel, para especificar completamente um objeto que esteja participando de uma interação, precisamos mencionar a instância de papel que está exercendo, a instância de organização em que a instância de papel está e a instância de ambiente na qual reside o objeto. Um objeto pode exercer mais de um papel que pode ser instâncias de diferentes classes de papel e mais de um papel que pode ser instâncias da mesma classe do papel. Dessa forma, o *pathname* completo de um objeto pode descrever o papel que está exercendo na interação, a organização em que está exercendo o papel e o ambiente em que reside. O

pathname associado aos objetos em diagramas de seqüência descreve, à esquerda da vírgula, os nomes das instâncias e, à direita da vírgula, os nomes da classe:

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

Imagine um SMA em que haja um objeto *Book* da classe *Item* que está exercendo a instância de papel *Desirable Book* da classe do papel *Desirer*. O objeto *Book* está exercendo o papel *Desirable Book* na instância de organização *Amazon*. *Amazon* é uma instância da classe de organização *Store of Goods*. Além disso, a organização *Store of Goods* é definida no ambiente *Virtual Marketplace*. *Book* reside na instância de ambiente *Place I*. Quando um agente interage com o objeto *Book*, pode ser necessário descrever o papel que o objeto está exercendo, a organização em que o objeto está exercendo esse papel e o ambiente em que o objeto reside.

```
Book/DesirableBook/Amazon/PlaceI :
Item/Desirer/StoreOfGoods/VirtualMarketplace
```

As organizações podem ser suborganizações de outras organizações. No caso de uma suborganização, é necessário representar todas as instâncias de organização envolvidas para especificar completamente um objeto. Nesse caso, o *pathname* do objeto deve ser modificado a fim de representar a hierarquia de organizações e as classes correspondente de cada organização:

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

O *pathname* completo pode ser substituído por um *pathname* simples, quando apropriado. As informações sobre o nome da instância, a instância do papel exercido pelo objeto, sobre o local da organização e o ambiente em que reside podem ser omitidas. O designer pode omitir essas informações quando o contexto da interação é bem conhecido ou irrelevante. Por exemplo, o *pathname* pode não conter as informações sobre o ambiente, se houver apenas uma instância de ambiente na aplicação (consulte (a)) ou pode não conter as informações sobre a organização, se houver apenas uma instância de organização em um determinado ambiente (consulte (b)).

```
a) object/role/org : Class/ObjectRoleClass/Organization
```

```
b) object/role/env : Class/ObjectRoleClass/Environment
```

4.4.1.2. *Pathname* do Agente

O *pathname* completo que representa um agente em um diagrama de seqüência é semelhante àquele que representa um objeto. O *pathname* descreve a instância do agente, a instância de papel que o agente está exercendo, a hierarquia de organização em que o agente está exercendo o papel, o ambiente em que reside e os nomes das classes correspondentes.

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

Suponha que haja uma instância do agente *Mary* da classe do agente *User Agent* que esteja exercendo a instância de papel *Buyer of Books* da classe do papel *Buyer*. O objeto *Mary* está exercendo o papel *Buyer of Books* na instância de organização *Amazon*. Quando a instância do agente *Mary* interage com o objeto *Book*, pode ser necessário descrever o papel que o agente está exercendo, a organização em que está exercendo esse papel e o ambiente em que reside.

```
Mary/BuyerOfBooks/Amazon/PlaceI :
UserAgent/Buyer/StoreOfGoods/VirtualMarketPlace
```

As informações sobre o nome da instância, o papel, as organizações e o ambiente podem ser omitidas; portanto, pode ser usado um *pathname* simples. Por exemplo, se as informações sobre a organização e o ambiente forem bem-conhecidas, o *pathname* do agente pode ser tão simples quanto:

```
agent/role : AgentClass/AgentRoleClass
```

4.4.1.3. *Pathname* da Organização

Uma suborganização pode exercer um papel em organizações e pode interagir com outras entidades. O *pathname* da suborganização descreve a instância da suborganização, o papel que ela está exercendo, a hierarquia completa da organização em que ela está definida, o ambiente em que reside e os nomes das classes correspondentes. Uma suborganização é representada pelo *pathname* a seguir:

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

Suponha que a classe de organização *Store of Goods* defina uma classe do papel do agente *Market of Used Goods*. Além disso, a organização *Store of Goods* define uma classe de suborganização *Second-hand Goods Store* associada à classe do papel *Market of Used Goods*. Isso significa que as instâncias de *Second-hand Goods Store* podem exercer instâncias do papel *Market of Used Goods*. Por exemplo, a instância de organização *Second-hand Bookstore* pode exercer a instância de papel *Market of Used Books*. A fim de descrever a instância de organização que está participando de uma interação, pode ser necessário descrever o *pathname* completo.

```
Second-handBookstore/MarketOfUsedBooks/Amazon/PlaceI :
Second-handGoodsStore/MarketOfUsedGoods/StoreOfGoods/VirtualMarketPlace
```

Um *pathname* simples pode ser usado quando apropriado. As informações sobre o nome da instância, o papel, a hierarquia da organização em que a suborganização é definida e sobre o ambiente podem ser omitidas. Por exemplo, se a suborganização exerce apenas um papel, e a hierarquia da organização e o ambiente forem bem conhecidos, o *pathname* da suborganização pode ser simples como:

```
suborg : OrgClass
```

O *pathname* que descreve as organizações principais é mais simples do que o *pathname* que descreve suborganizações. Como uma organização principal não exerce papéis, seu *pathname* deve descrever a instância da organização principal, o ambiente em que reside e os nomes das classes correspondentes, conforme apresentado a seguir:

```
main-organization/env : OrgClass/EnvClass
```

4.4.1.4. *Pathname* do Ambiente

O *pathname* que especifica completamente o ambiente é o mais simples. Ele descreve o nome da instância de ambiente e o nome da classe. O nome da instância de ambiente pode ser omitido quando apropriado.

```
env : EnvironmentClass
```

A Tabela 1 identifica as instâncias que podem aparecer em diagramas de seqüência, seus ícones e os *pathnames* completos associados. Os objetos são exibidos como retângulos, os agentes são mostrados como retângulos com cantos arredondados e as organizações como losangos. O ambiente é mostrado como um retângulo arredondado se for uma entidade ativa e como um retângulo, se for passiva.

Tabela I – Os elementos do diagrama de seqüência

Instâncias	*Elemento do diagrama*
Objeto	*obj/role/org/env : Class/ObjectRole/Organization/Environment*
Agente	*agent/role/org/env : Agent/AgentRole/Organization/Environment*
Organização	*org/role/org/env : Organization/AgentRole/Organization/Environment*
Ambiente	*Active entity*: *env : Environment*; *Passive entity*: *env : Environment*

4.4.2. Extensão do Conceito de Mensagem

O diagrama de seqüência de UML ilustra um objeto que está enviando mensagens a outros objetos, ou seja, um objeto que está chamando métodos de outros objetos. Uma mensagem é apresentada como uma linha direta do emissor para o destinatário, quase sempre incluindo o nome de sua operação/método. A fim de representar agentes ou organizações que enviam e recebem mensagens, foi necessário estender o conceito de *mensagem* usado no diagrama de seqüência de UML.

No diagrama de seqüência estendido proposto pela MAS-ML, pode ser usada uma seta para representar um método de um objeto ou ambiente passivo que está sendo chamado por outra entidade ou para representar uma mensagem enviada por um agente, organização ou ambiente ativo, conforme ilustrado na Figura 25. As setas com cabeça cheia representam mensagens síncronas, e as setas com cabeça aberta representam mensagens assíncronas. Uma seta de cabeça cheia é usada para ilustrar um método sendo chamado associando-se o nome de um método. A seta indica a entidade que está chamando o método – o emissor – e a

entidade que executará o método – o destinatário. Uma seta com cabeça aberta é usada para ilustrar uma mensagem do agente associando o rótulo da mensagem e a descrição do conteúdo a ela. A linha indica o emissor e o destinatário da mensagem.

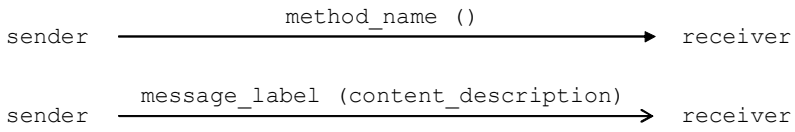


Figura 25 – Um método sendo chamado e uma mensagem de agente sendo enviada.

Uma linha pode ser facilmente identificada observando-se o destinatário da mensagem. Se o destinatário for um objeto ou um ambiente passivo, a linha está sendo usada para representar um método sendo chamado. Se o destinatário for um agente, uma organização ou um ambiente ativo, a linha é usada para indicar uma mensagem de agente sendo enviada.

4.4.3. Os estereótipos Associados a Mensagens

Com base no comportamento independente do domínio descrito na Seção 3.2, será descrito um conjunto de estereótipos associados a mensagens. Usando estereótipos, é possível ilustrar todos os processos dinâmicos primitivos e os processos dinâmicos de alto nível de SMAs, conforme definido no *framework* conceitual TAO.

Os estereótipos `<<create>>` e `<<destroy>>` associados a uma mensagem em UML são usados para representar a criação e a destruição de objetos. Propomos usar os mesmos estereótipos para representar a criação e a destruição de agentes, organizações e ambientes. Ademais, outros estereótipos associados a mensagens foram criados a fim de representar agentes e organizações que estejam se comprometendo com papéis, estejam mudando, ativando, desativando e cancelando papéis.

4.4.3.1. Os Estereótipos <<create>> e <<destroy>>

O estereótipo `<<create>>` associado a uma mensagem é usado quando o designer deseja representar uma entidade que está criando outra. A entidade que envia a mensagem é a criadora e a que recebe é a que está sendo criada.

Esse estereótipo foi especializado a fim de representar a criação de agentes, organizações e ambientes. Além do mais, ele também pode identificar a criação de um papel que será exercido pelo agente, organização ou objeto sendo criado. Conforme definido na Seção 3.2.1.1, quando um agente ou uma organização é criado, também pode ser criado um papel que será associado ao agente ou organização. Portanto, o uso da mensagem `<<create>>` possibilita a identificação da criadora de uma entidade, a entidade sendo criada e o papel associado que será exercido pela entidade, se for o caso. O estereótipo `<<create>>` também pode ser usado para representar a criação de um objeto e a associação de um papel ao objeto, pois os objetos também podem exercer papéis (Seção 3.1.1).

Como o estereótipo `<<create>>` representa a criação de uma entidade, o ciclo de vida da nova entidade começa com o recebimento da mensagem estereotipada como `created`. Se a entidade for um agente, uma organização ou um objeto criado para exercer um papel, o recebimento da mensagem estereotipada como `created` também indicará que uma nova instância de papel foi criada e associada à entidade. O primeiro bloco da Figura 26 ilustra uma organização sendo criada por um agente. Quando a organização *Fruit Store* é criada, uma instância de papel chamada *Fruit Wholesale* também é criada. A organização *Fruit Store* é uma instância da classe de organização *Store*, e o papel *Fruit Wholesale* é uma instância da classe do papel chamada *Wholesale*.

O estereótipo `<<destroy>>` associado a uma mensagem é usado quando o designer deseja representar uma entidade que está destruindo outra. A entidade que envia a mensagem é a destruidora e a que recebe é a que está sendo destruída.

O estereótipo `<<destroy>>` foi especializado a fim de representar a destruição de agentes, organizações e ambientes, e a destruição de todas as instâncias de papel associadas a agentes, organizações e objetos. Conforme definido na Seção 3.2.1.3, quando um agente, uma organização ou um objeto é destruído, todas as instâncias que estão sendo exercidas devem ser destruídas ao mesmo tempo. O ciclo de vida da entidade sendo destruída termina com o recebimento da mensagem estereotipada como `destroy` e recebe a marcação visual de um grande X. O segundo e o terceiro bloco da Figura 26 ilustram a destruição de uma organização por um agente. O segundo bloco representa a organização *Fruit Store* sendo destruída ao exercer o papel *Store*. Quando uma organização é destruída, todos os seus papéis também são destruídos. Para ilustrar

melhor a destruição de todas as instâncias de papel, o designer pode representar a organização sendo destruída sem qualquer instância de papel associada, conforme representado no terceiro bloco da Figura 26.

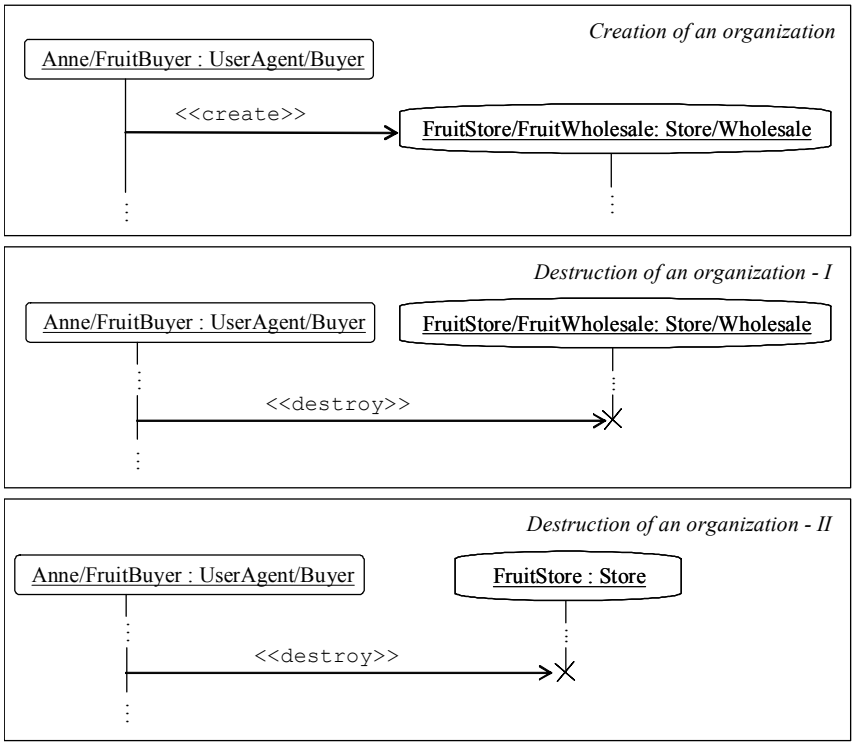


Figura 26 – Criação e destruição de uma organização.

Usando os estereótipos `<<create>>` e `<<destroy>>`, é possível representar os processos dinâmicos primitivos chamados criação e destruição de agentes, organizações, objetos e ambientes. No entanto, não é possível representar a criação de um papel (papel de objeto e papel do agente), quando a entidade que executará o papel já foi criada, tampouco é possível representar a destruição de um papel sem a destruição da entidade que está exercendo o papel. Portanto, foram criados cinco novos estereótipos associados a mensagens a fim de representar um agente, organização ou objeto que esteja se comprometendo com um papel e que esteja cancelando um papel, e para representar um agente ou uma organização que esteja alterando seus papéis e alterando o estado de um papel.

4.4.3.2. O Estereótipo <<role_commitment>>

O estereótipo `<<role_commitment>>` associado a uma mensagem foi criado para representar um agente, organização ou objeto que esteja se comprometendo com um papel. A entidade que envia a mensagem é aquela que está criando o

papel (criando uma instância de papel), e a entidade que recebe a mensagem é a entidade que exercerá o papel. Conforme descrito na Seção 3.2.1.1, quando uma instância de papel é criada, ela é sempre associada a um agente, suborganização ou objeto. Observe que a entidade que envia uma mensagem e aquela que recebe a mensagem podem ser as mesmas.

Quando um agente, organização ou objeto se compromete com um novo papel, eles não param de exercer outros papéis. Uma nova instância de papel é criada e associada à entidade. O ciclo de vida de um papel começa com o recebimento da mensagem estereotipada como `role_commitment`. Isso não indica que uma nova instância de entidade (instância de objeto, agente, suborganização) é criada, e sim, uma nova instância de papel.

O novo papel é uma instância de qualquer classe do papel definida pela classe de organização que reside no mesmo ambiente da entidade, conforme descrito na Seção 3.2.1.1. As entidades não podem exercer diferentes papéis em vários ambientes. Portanto, o estereótipo `<<role_commitment>>` pode representar um agente ou uma organização que está entrando em uma organização para exercer um papel, mas não pode representar um agente ou uma organização que está entrando em um novo ambiente.

Quando o estereótipo `<<role_commitment>>` é usado, está implícito que há uma negociação entre o agente que deseja exercer o papel e a organização que define o papel. Essa negociação está incorporada no estereótipo `<<role_commitment>>`. O designer pode omitir a representação da negociação.

Como o criador de uma instância de papel do agente pode ser o agente ou a organização que exercerá o papel (Seção 3.2.1.1), o primeiro bloco ilustrado na Figura 27 exemplifica um agente que está exercendo o papel *Fruit Buyer* comprometendo-se com o papel *Clothes Buyer*, ou seja, criando a instância de papel *Clothes Buyer* e comprometendo-se com o papel. O segundo bloco ilustrado na Figura 27 exemplifica um agente que está exercendo o papel *Fruit Buyer* criando o papel *Desirer Fruit* e associando esse papel ao objeto *Apple*. Como o criador de um papel de objeto é sempre um agente ou organização, a entidade que envia a mensagem estereotipada como `role_commitment` ao objeto deve ser um agente ou organização.

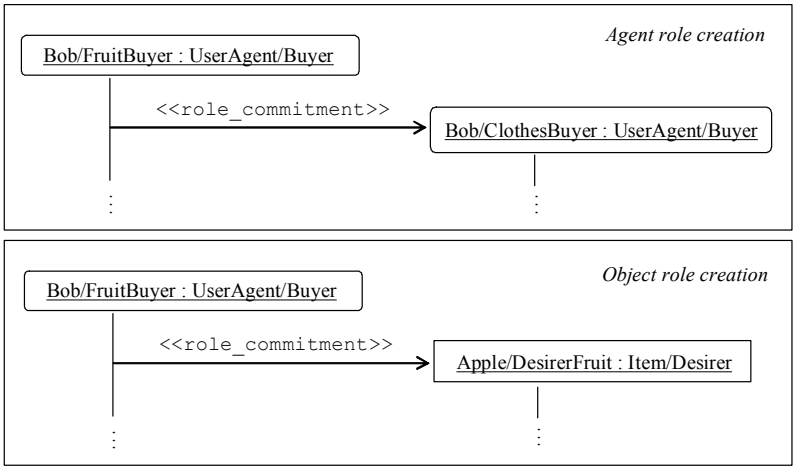


Figura 27 – Comprometendo-se com um papel.

4.4.3.3. O Estereótipo <<role_cancel>>

O estereótipo `<<role_cancel>>` representa um papel sendo cancelado, ou seja, destruído. O compromisso entre a entidade — agente, organização ou objeto — e o papel é cancelado, e a entidade pára de exercer o papel. Os demais papéis associados às entidades não são modificados. O ato de cancelar um papel pode ser executado por um agente ou uma organização se estiver exercendo mais de um papel. Um agente ou uma organização não pode cancelar seu papel exclusivo a menos que esteja mudando de uma organização para outra ou de um ambiente para outro, mudando, assim, seu papel. Um agente e uma organização devem exercer pelo menos um papel em uma organização (ou organização principal). O ciclo de vida do papel que está sendo cancelado termina com o recebimento da mensagem estereotipada como `role_cancel` e recebe a marcação visual de um grande X.

Como o destruidor de um papel do agente é o agente ou a organização que está exercendo o papel (Seção 3.2.1.3), a entidade que envia a mensagem estereotipada como `role_cancel` é a entidade (agente ou organização) que está exercendo o papel. Como o destruidor de um papel de objeto é um agente ou organização (Seção 3.2.1.3), a entidade que envia a mensagem estereotipada como `role_cancel` ao objeto é um agente ou organização. A Figura 28 mostra um agente cancelando um de seus papéis e um agente cancelando o papel de um objeto. Um agente pode cancelar um papel enquanto o exerce ou pode cancelá-lo enquanto está exercendo outro papel, conforme apresentado no primeiro bloco.

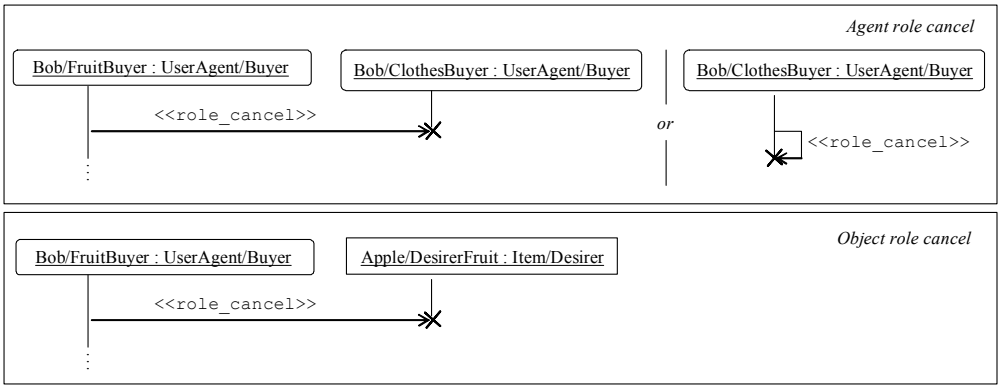


Figura 28 – Cancelando um papel.

É possível ilustrar um agente ou uma organização que está saindo de outra organização usando o estereótipo `<<role_cancel>>` (Seção 3.2.2.1). Um agente que esteja deixando um ambiente não pode ser ilustrado usando esse estereótipo, pois para sair de um ambiente, ele deve parar de exercer todos os papéis nesse ambiente e começar a exercer pelo menos um papel em outro ambiente (Seção 3.2.2.2).

4.4.3.4. Os Estereótipos <<role_deactivate>> e <<role_activate>>

O estereótipo `<<role_deactivate>>` muda o estado de um papel que um agente ou organização está exercendo de ativo para inativo. O ciclo de vida do papel fica inativo com o envio da mensagem estereotipada como `role_deactivate` e recebe a marcação visual de um grande traço – encerrando o estado ativo. O estereótipo `<<role_activate>>` altera o estado de um papel de inativo para ativo. O ciclo de vida do papel fica ativo com o recebimento da mensagem estereotipada como `role_activate` e recebe a marcação visual de um grande traço – iniciando o estado ativo. Os outros papéis que um agente ou organização está exercendo não se alteram quando um determinado papel muda seu estado. A Figura 29 ilustra o uso dos estereótipos `<<role_activate>>` e `<<role_deactivate>>`.

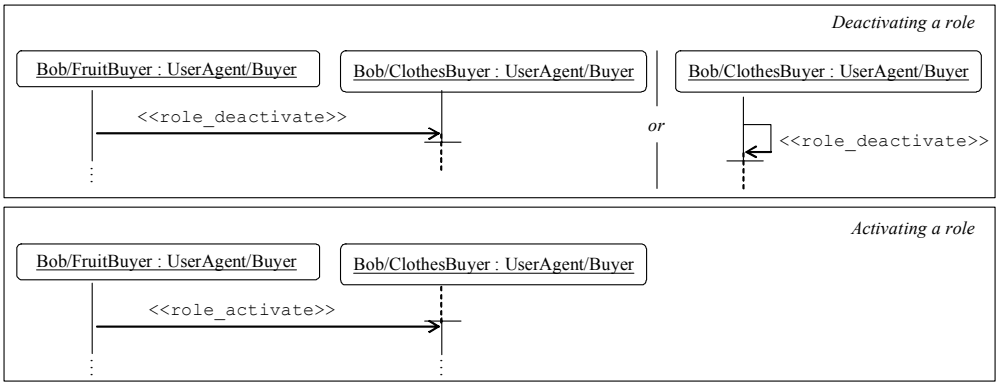


Figura 29 – Ativando e desativando um papel.

4.4.3.5. Os Estereótipos <<role_change>>

O estereótipo `<<role_change>>` representa um agente ou uma organização que está mudando seu papel. Um objeto não muda de um papel para outro, porque não tem autonomia para escolher seus papéis. Um agente ou uma organização escolhe o papel de um objeto. Os agentes e as organizações criam papéis e os associam a objetos (compromisso) ou cancelam os papéis dos objetos.

A entidade que altera seus papéis pára de exercer um papel e começa a exercer outro. A entidade pode criar um novo papel ou pode ativar um de seus papéis desativados. Por outro lado, o papel que a entidade estava exercendo pode ser cancelado ou se tornar inativo. O recebimento da mensagem estereotipada como `role_change` pode indicar que um novo papel foi criado ou que um papel foi reativado. Ademais, a mensagem também indica que o papel original foi cancelado ou que ficou inativo. A Figura 30 demonstra essas quatro possibilidades. No primeiro bloco, *Bob* está cancelando o papel *Fruit Buyer* e se comprometendo com um novo papel chamado *Clothes Buyer*. No segundo bloco, *Bob* está cancelando o papel *Fruit Buyer* e ativando o papel *Clothes Buyer*. O terceiro bloco ilustra *Bob* desativando o papel *Fruit Buyer* e se comprometendo com o papel *Clothes Buyer*. Finalmente, no quarto bloco, *Bob* está desativando o papel *Fruit Buyer* e ativando o papel *Clothes Buyer*.

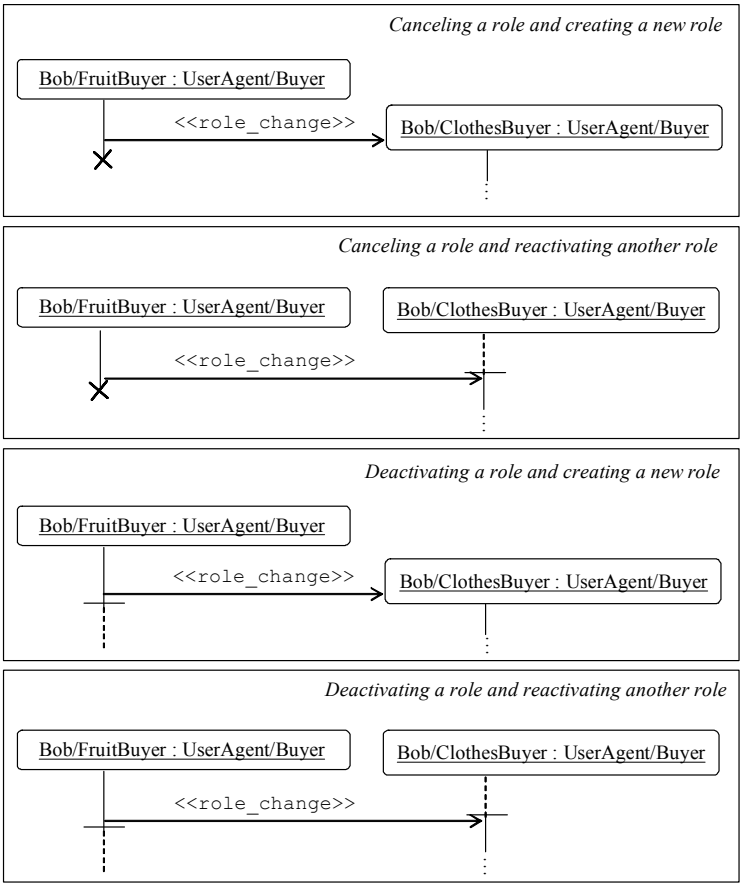


Figura 30 – Alterando papéis.

4.4.3.6. Usando os Estereótipos para Modelar os Aspectos Dinâmicos Independentes do Domínio

Usando a extensão dos estereótipos `<<create>>` e `<<destroy>>` e os cinco novos estereótipos apresentados nesta Seção, é possível ilustrar todos os processos dinâmicos primitivos e os processos dinâmicos de alto nível de um SMA. A criação de entidades e a associação de papéis com agentes, organizações e objetos são representadas usando o estereótipo `<<create>>`. A destruição de entidades e a destruição de todos os papéis exercidos por agentes, organizações e objetos são modeladas usando o estereótipo `<<destroy>>`. Quase todos os processos dinâmicos primitivos são modelados usando esses dois estereótipos.

O processo dinâmico de um agente ou suborganização que está entrando em uma organização pode ser ilustrado usando os estereótipos `<<role_commitment>>` e `<<role_activate>>` porque esses estereótipos indicam que um papel está sendo associado a um agente ou uma suborganização. Pode ser criado um novo papel ou

pode ser reativado um papel antigo em uma organização diferente. Além do mais, o comportamento descrito por um agente ou organização que está deixando uma organização pode ser representado usando os estereótipos <<role_cancel>> e <<role_deactivate>>. Um agente pode cancelar todos os papéis que está exercendo em uma organização ou pode desativá-los. O estereótipo <<role_change>> pode ser usado para ilustrar um agente ou organização que está deixando uma organização e entrando em outra.

O processo dinâmico de um agente ou suborganização que está entrando em um ambiente não pode ser ilustrado usando os estereótipos <<role_commitment>> ou <<role_activate>> porque um agente ou uma organização que está exercendo um papel em um ambiente não pode se comprometer com um papel ou reativar um papel em outro ambiente. Agentes e organizações não podem exercer papéis em diferentes ambientes. Ademais, os estereótipos <<role_cancel>> e <<role_deactivate>> não podem ilustrar um agente ou uma organização que esteja saindo de um ambiente. O uso desses estereótipos a fim de ilustrar um agente que está saindo de um ambiente pressuporia que o agente ou a organização estava exercendo papéis em diferentes ambientes. O designer deve usar o estereótipo <<role_change>> para modelar um agente ou uma organização que esteja saindo de um ambiente, enfatizando que o agente ou a organização pára de exercer um papel em um ambiente e começa a exercer um papel em outro. A entidade se move de um ambiente para outro mudando seus papéis.

Ao modelar um dos processos dinâmicos de alto nível descritos na Seção 3.2.2, o designer tem duas opções: modelar todo o processo dinâmico ou omitir os detalhes, uma vez que o processo dinâmico é um processo bem definido. Os processos dinâmicos de se mover de um ambiente para outro, de sair de uma organização e entrar em uma organização são compostos por um conjunto de interações entre agentes, organizações e ambientes. Os estereótipos podem ser usados para omitir todas as interações ao modelar esses processos dinâmicos. A complexidade de um agente ou suborganização que muda de um ambiente para outro e que envolve a escolha de outro ambiente, a escolha de outra organização e a escolha de outro papel (Seção 3.2.2.2) pode ser encapsulada pelo estereótipo <<role_change>>. O estereótipo <<role_commitment>> ou <<role_activate>> pode ser usado para encapsular a complexidade de um agente ou suborganização que está entrando na organização. O estereótipo <<role_cancel>> ou

<<role_deactivate>> pode ser usado para encapsular a complexidade de um agente ou suborganização que está deixando uma organização.

4.4.4. Modelando Planos e Ações

Embora as intra-ações de objetos (e ambientes passivos) sejam definidas pelos métodos que executam, as intra-ações de agentes, organizações e ambientes ativos são definidas pelos planos e ações que executam. Ao executar métodos, objetos (ou ambientes passivos) podem executar algoritmos, podem chamar seus próprios métodos e métodos de outros objetos, e podem mudar seu estado interno, por exemplo. Ao executar as ações associadas a seus planos, agentes (organização ou ambientes ativos) podem, por exemplo, enviar e receber mensagens, podem chamar métodos de objetos (ou ambientes passivos) e podem alterar seu estado interno. Agentes e organizações também podem se comprometer com um papel e alterar seus papéis, por exemplo.

Os planos executados por um agente (ou organização) são selecionados com base no estado interno do agente e nas características dos papéis que o agente está exercendo. Agentes executam planos a fim de alcançar seus objetivos e os objetivos dos papéis que estão exercendo. Cada plano predefine a seqüência de ações que devem ser executadas por um agente enquanto executa o plano. A seleção dos planos de um ambiente ativo é semelhante à seleção dos planos do agente, mas os ambientes não exercem papéis.

A fim de modelar as intra-ações de agentes, as organizações e os ambientes ativos, um diagrama de seqüência deve ser usado para modelar a seleção de planos, para modelar os planos e para modelar a execução das ações. Dependendo da complexidade dos planos e das ações, deve ser usado um diagrama de seqüência a fim de modelar cada plano e cada ação dos planos.

A execução de um plano ou uma ação é representada por uma seta que começa no agente e termina nele, ao começar um outro foco de controle (ou barra de ativação). O foco de controle de um plano define a seqüência de ações que serão executadas pela entidade no contexto do plano. O foco de controle de uma ação define o que a entidade fará ao executar a ação. A chamada de um plano ou uma ação recebe o nome do plano ou da ação a fim de indicar o que será executado. A Figura 31 ilustra a execução do plano chamado “creating seller”. O

plano verifica se a organização recebeu uma mensagem para criar um vendedor e executa as ações “create seller” e “inform buyer of seller”. A ação “create seller” cria um agente de usuário para exercer o papel de vendedor, e a ação “inform buyer of seller” informa o comprador que pediu ao vendedor a identificação do novo vendedor. Observe que na Figura 31 o nome das instâncias foi omitido.

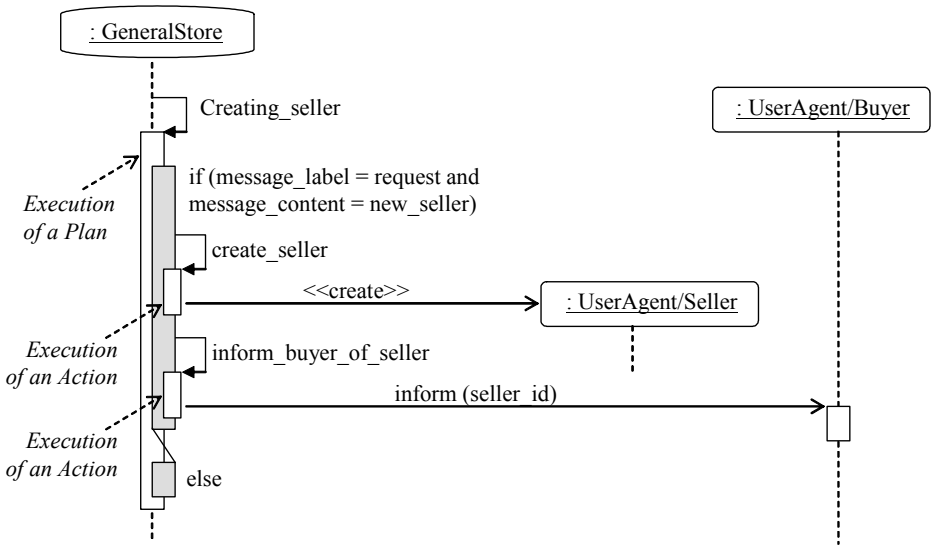


Figura 31 – Modelando a execução de um plano e suas ações.

4.4.5. Modelando Protocolos

Conforme mencionado anteriormente, um protocolo define o conjunto de mensagens que os agentes e as suborganizações podem enviar e as mensagens que podem receber enquanto estão exercendo papéis. Os protocolos definem os padrões de comunicação ao identificar as mensagens e os papéis (Odell et al., 2000). A fim de usar os diagramas de seqüência para modelar protocolos, é necessário representar os papéis envolvidos nos protocolos e as mensagens enviadas e recebidas pelos papéis. As mensagens descritas em protocolos são mensagens de agentes e estão ilustradas seguindo a representação descrita na Seção 4.4.2.

Para representar papéis em diagramas de seqüência, foi definido outro elemento de diagrama. Ao descrever o elemento do diagrama de papel, é necessário definir o *pathname* que identifica o papel e o ícone que ilustra o papel. O *pathname* completo que representa uma instância de papel em um diagrama de seqüência identifica o nome da instância de papel e o nome da classe do papel. O nome da instância de papel pode ser omitido. O ícone usado para modelar a

instância de papel em diagramas de seqüência é um retângulo sólido com uma curva na parte inferior. Esse ícone é semelhante ao ícone que representa um papel em diagramas estruturais. A Figura 32 ilustra um protocolo que define a interação entre compradores e vendedores ao negociar um item.

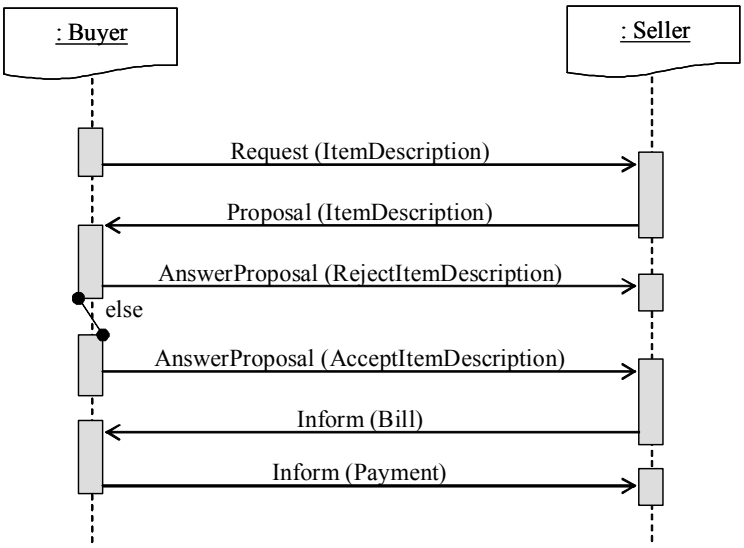


Figura 32 – O protocolo de negociação.

4.4.6. Modelando Concorrência e Distribuição

Duas características importantes de sistemas multiagentes podem ser modeladas usando os diagramas de seqüência de MAS-ML: concorrência (paralelismo) e distribuição. Conforme mencionado anteriormente, agentes e suborganizações podem exercer vários papéis ao mesmo tempo e, portanto, podem executar diferentes ações ao exercer os papéis. Ademais, as entidades de SMAs podem executar em diferentes ambientes.

A concorrência e o paralelismo podem ser modelados em UML representando as interações entre objetos ativos e representando mensagens assíncronas, ou seja, representando métodos que são chamados de forma assíncrona. A MAS-ML estende o diagrama de seqüência de UML a fim de representar a interação entre diferentes entidades autônomas e a fim de representar mensagens assíncronas enviadas por elas. Usando o diagrama de seqüência proposto pela MAS-ML, é possível modelar diferentes entidades que estejam executando em paralelo e modelar a mesma entidade (agente ou suborganização) que está exercendo diferentes papéis e que está atualizando o estado mental da

entidade de forma concorrente. Duas entidades que estejam executando em paralelo interagem enviando e recebendo mensagens assíncronas. A concorrência pode ser ilustrada quando a mesma entidade está exercendo dois papéis. Uma ação que atualiza o estado mental da entidade pode ser executada ao mesmo tempo pela entidade, enquanto ela exerce dois papéis.

O diagrama de seqüência de UML não oferece uma característica de design para modelar a característica de distribuição de uma aplicação. A característica de distribuição é modelada por meio de diagramas de implantação. Em (Gomaa et al., 2000), os autores propõem usar estereótipos para identificar objetos internos do sistema e objetos externos do sistema.

A MAS-ML estende o diagrama de seqüência de UML a fim de modelar as entidades de SMAs que estão trabalhando em *hosts* diferentes e distribuídos por uma rede. Na MAS-ML, o *pathname* que identifica uma entidade em um diagrama de seqüência descreve as instâncias de ambiente nas quais as entidades residem. Portanto, é possível modelar as entidades que residem em diferentes ambientes no mesmo diagrama. Ademais, usando o estereótipo `<<role_change>>`, é possível modelar um agente que está se movendo de um ambiente para outro.

4.5. Discussão

Três diagramas estruturais e um diagrama dinâmico compõem a linguagem de modelagem MAS-ML. Os diagramas estruturais (diagramas de classes, organização e papel) modelam os aspectos estruturais de uma aplicação. Usando esses diagramas, é possível modelar todas as entidades, suas propriedades e o relacionamento entre entidades definidos em uma aplicação. Todas as classes de um sistema são modeladas em diagramas de organização quando exercem os papéis, em diagrama de papel quando um papel interage com uma classe e em diagramas de classes quando a classe interage com outras classes, agentes e organizações.

Usando o diagrama de organização, é possível modelar todas as classes de organização, todas as classes do agente, todas as classes de ambiente e todas as classes de papel. Como todas as classes de papel são definidas pelas classes de organização e os diagramas de organização modelam todas as organizações, todas as classes de papel também são modeladas. Além disso, como os agentes e as

suborganizações exercem pelo menos um papel em uma organização, e todas as classes de papel são definidas nos diagramas de organização, o conjunto de diagramas de organização descreve todas as classes do agente e as classes de suborganização definidas em um sistema e as associa a classes de papel. Além disso, como pelo menos uma organização reside em um ambiente e todas as classes de organização são modeladas no conjunto de diagramas de organização, todas as classes de ambiente são modeladas no conjunto de diagramas de organização.

Usando o diagrama de seqüência dinâmico, é possível modelar (i) a interação entre agentes, organizações, ambientes e objetos, (ii) a execução de planos e ações associados a agentes, organizações e ambientes ativos e (iii) modelar protocolos definidos por papéis. As entidades que são modeladas nos diagramas de seqüência são instâncias das classes de entidades modeladas nos diagramas estruturais. Os métodos, planos e ações modelados nos diagramas de seqüência também são definidos nos diagramas estruturais associados às respectivas classes de entidades.

Como um diagrama de seqüência pode modelar diferentes entidades que executam vários planos e ações ao mesmo tempo e também pode modelar a mesma entidade que está exercendo mais de um papel, esse diagrama pode expressar concorrência e paralelismo nos modelos de design. Além do mais, como um diagrama de seqüência pode modelar diferentes entidades que estão executando em vários ambientes e uma entidade que está se movendo de um ambiente para outro, o diagrama de seqüência pode expressar a distribuição nos modelos de design.