

5

Aplicação do método Observed-MAS para a linguagem MAS-ML

Neste capítulo é apresentada uma prova de conceito para o método Observed-MAS. A prova de conceito é definida a partir da aplicação do método para a linguagem de modelagem MAS-ML (*Multi-Agent Systems Modeling Language*) (Silva & Lucena, 2004).

5.1.

Aplicação do Observed-MAS para MAS-ML: visão geral

Durante a descrição geral do Observed-MAS (Capítulo 4) ficou estabelecido que cada fase do método (F1 e F2 – fases um e dois, respectivamente) é composta por três etapas distintas (Figura 44): uma ontologia (Ont1 e Ont2 - ontologias de F1 e F2, respectivamente) e dois conjuntos de consultas, um destinado a descobrir violações a propriedades de especificação dos modelos de SMAs (QV1 e QV2 – consultas às violações nas fases F1 e F2, respectivamente) e outro destinado a sugerir boas práticas de modelagem (QD1 e QD2 – consultas para sugerir boas práticas de modelagem nas fases F1 e F2, respectivamente).

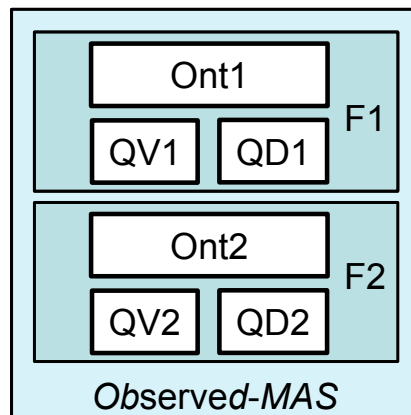


Figura 44 Observed-MAS resumido

Seguindo os passos previstos para a execução do método, a aplicação do Observed-MAS para a linguagem MAS-ML teve início com a instanciação da fase F1 do método. Na primeira etapa da fase F1 foi feita a construção da ontologia Ont1, combinando-se os conceitos que compõem a primeira ontologia usada para descrever as entidades e relacionamentos especificados no TAO (4.3.1 e 4.3.2), com os termos e relacionamentos que compõem o metamodelo da linguagem MAS-ML. Esta combinação, que será apresentada na seção 5.3.1, permite a descrição dos diagramas MAS-ML como instâncias de Ont1, gerando a estruturação formal de cada diagrama, e possibilitando a análise de suas propriedades internas através das consultas QV1 definidas na segunda etapa da fase F1 do Observed-MAS.

A segunda etapa da fase F1 foi instanciada com a definição de consultas (QV1) para apoiar a análise dos diagramas MAS-ML. Esta análise considera propriedades inerentes a SMAs e propriedades específicas a cada diagrama MAS-ML. Assim, nesta etapa acrescentou-se novas consultas àquelas apresentadas na seção 4.4.2, agora relativas a violações às propriedades internas dos diagramas MAS-ML. Estas consultas são apresentadas na seção 5.3.2.

A terceira etapa da fase F1 foi instanciada com a definição de consultas (QD1) que retornam como resposta sugestões de boas práticas de modelagem usando MAS-ML. As consultas QD1 têm como foco sugerir ao desenvolvedor boas práticas de modelagem na construção de cada diagrama e na definição das propriedades internas das entidades. Estas consultas são apresentadas na seção 5.3.3.

A fase F2 foi instanciada depois de finalizada a instanciação da fase F1. Na primeira etapa da fase F2 foi definida a ontologia Ont2, uma extensão conservativa de Ont1, obtida a partir da introdução de axiomas que descrevem as propriedades internas dos diagramas MAS-ML e das entidades que participam de um SMA. Estes axiomas, apresentados na seção 5.4.1, têm como base as consultas QV1 definidas durante a fase F1 do método.

A segunda etapa da fase F2 foi instanciada com a definição de consultas (QV2) para apoiar a análise da modelagem de um SMA, cujos diagramas são descritos em MAS-ML. Diferentemente da análise realizada na segunda etapa da fase F1, esta análise considera propriedades de interdependência entre os diversos diagramas definidos no metamodelo de MAS-ML (seção 5.4.2).

A terceira etapa da fase F2 foi instanciada com a definição de consultas (QD2) que retornam como resposta sugestões de boas práticas de modelagem

usando MAS-ML. As consultas QD2 têm como foco sugerir ao desenvolvedor boas práticas de modelagem a partir de uma visão macro de modelos de SMAs, pois considera todos os diagramas que compõem a modelagem para sugerir pontos que podem ser melhorados. A instanciação da fase F2 do método é apresentada na seção 5.4.

Este capítulo é organizado da seguinte maneira: na seção 5.2 descreve-se o metamodelo de MAS-ML, a linguagem de modelagem escolhida; nas seções 5.3 e 5.4 descreve-se a instanciação das fases F1 e F2, respectivamente. A seção 5.5 apresenta a ferramenta de suporte implementada para automatizar a aplicação do Observed-MAS. Finalmente, a seção 5.6 apresenta uma análise crítica do método, com suas principais contribuições na área de engenharia de SMAs.

5.2.

O Metamodelo de MAS-ML

O metamodelo da linguagem de modelagem MAS-ML é baseado no TAO (Silva et al, 2003), já descrito no Capítulo 4, e no metamodelo de UML (OMG, 2004). A fim de representar as novas entidades e relacionamentos que compõem um SMA, e que foram explicitados no TAO, percebeu-se que a linguagem de modelagem UML não dispunha de elementos suficientes para representá-los, dadas as diferenças estruturais e comportamentais entre agentes e objetos. Assim, observou-se a necessidade da extensão do metamodelo de UML, através da adição de novas metaclasses, pois os outros mecanismos de extensão disponíveis (*stereotype*, *tag definition*, *constraints*) não eram suficientes para fazê-lo. A adição de novas metaclasses ao metamodelo de UML é chamada extensão de primeira classe e, quando utilizada, altera a estrutura do metamodelo base, gerando um novo metamodelo. Em particular, neste caso gerou-se o metamodelo de MAS-ML.

Nas Figura 45 e Figura 46 o metamodelo de MAS-ML é apresentado parcialmente. Na Figura 45 são destacadas as metaclasses referentes as novas entidades, e que foram adicionadas ao metamodelo de UML. Na Figura 46 são destacadas as metaclasses referentes aos novos relacionamentos, e que foram adicionados ao metamodelo de UML.

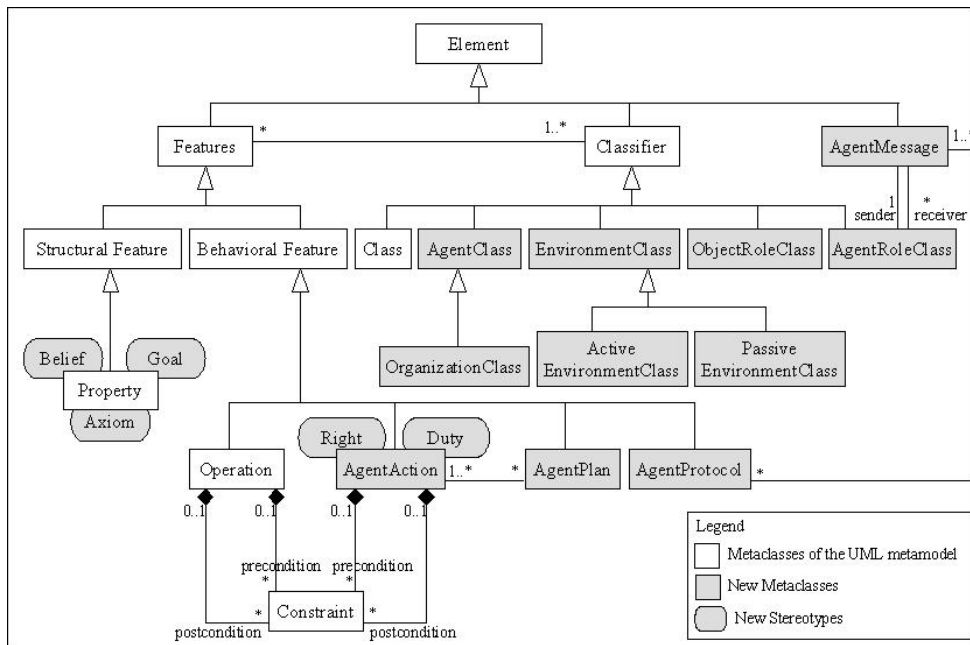


Figura 45. Metamodelo MAS-ML: extensão do metamodelo de UML. (fonte (Silva, 2004))

O metamodelo de MAS-ML é composto pelas entidades e relacionamentos definidos no TAO, pelos diagramas que descrevem visões específicas da forma como as novas entidades podem se relacionar e pelas entidades e relacionamentos previstos no metamodelo de UML, alguns deles estendidos para atender também às novas entidades. *AgentClass*, *OrganizationClass*, *AgentRoleClass*, *ObjectRoleClass* e *EnvironmentClass* são as novas entidades que compõem o metamodelo. A metaclasses *Class*, do metamodelo de UML, foi incorporada ao metamodelo de MAS-ML sem nenhuma alteração, uma vez que ela continua representando objetos. As metaclasses *inhabit*, *ownership*, *play*, *control*, *generalization*, *dependency*, *aggregation* e *association* são os relacionamentos previstos no metamodelo.

Novos diagramas, como os diagramas de organização e de papéis, foram acrescentados aos diagramas de UML e os diagramas de classe e de sequência foram estendidos para acomodar as entidades específicas do domínio de SMAs. O diagrama de classe foi estendido para permitir a visualização do relacionamento entre as classes de objeto com as novas classes de agentes, de organizações e de ambientes. Além destes relacionamentos, também são visualizados neste diagrama relacionamentos entre agentes, entre organizações e entre ambientes (i.e. *specialization*).

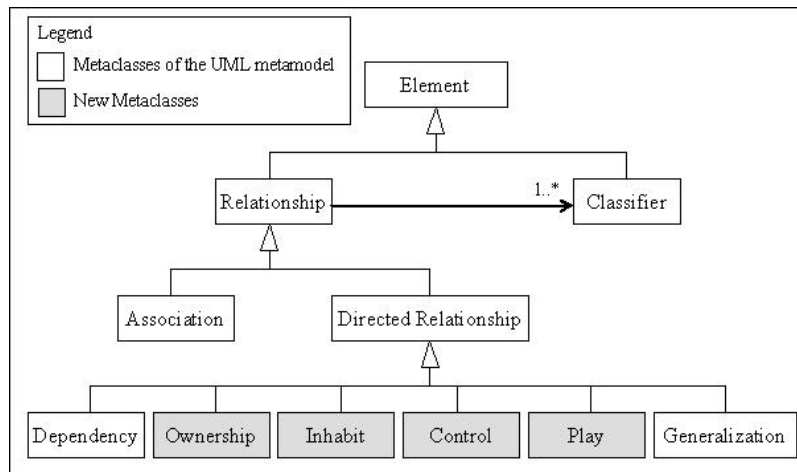


Figura 46. Relacionamentos MAS-ML: extensão do metamodelo de UML. (fonte(Silva, 2004))

Os diagramas de organizações têm por objetivo modelar todas as organizações de um SMA. Cada diagrama de organização é responsável pela modelagem de uma organização, ou seja, suas propriedades internas (crenças, metas, planos, ações e axiomas), os papéis que ela possui, as entidades que desempenham estes papéis e o ambiente que a organização e as entidades que desempenham papéis nesta organização estão inseridas.

Os diagramas de papéis ilustram os relacionamentos entre as classes de papéis identificadas nos diagramas de organizações. Além de ilustrar estes relacionamentos, que também descrevem interações entre sub-organizações, também são ilustrados relacionamentos envolvendo classes de objetos e classes de papéis.

Os diagramas de seqüência foram definidos para permitir a visualização dos aspectos dinâmicos relacionados às entidades definidas no metamodelo de MAS-ML. Estes aspectos são divididos em dois grupos: os dependentes e os independentes do domínio da aplicação. Os aspectos dinâmicos independentes do domínio da aplicação são aqueles relacionados à criação e destruição de entidades (processos dinâmicos primitivos) e aqueles relacionados a comportamentos derivados dos relacionamentos *inhabit*, *play* e *ownership* (processos dinâmicos complexos). Os aspectos dinâmicos dependentes do domínio da aplicação são aqueles que descrevem a execução interna e as interações entre entidades específicas a um domínio de aplicação, como aqueles derivados dos relacionamentos *association*, *aggregation*, *dependency*, *control* e *specialization*.

Para descrever os processos dinâmicos primitivos, o metamodelo de MAS-ML prevê a utilização de diagramas de seqüência usando os estereótipos

<<create>> e <<destroy>> associados às mensagens trocadas no diagrama. Para descrever os processos dinâmicos complexos derivados da dinâmica dos papéis, novos estereótipos foram criados. São eles: <<role_commitment>>, <<role_cancel>>, <<role_activate>>, <<role_deactivate>> e <<role_change>>. O estereótipo <<role_commitment>> rotula a mensagem de criação de papéis, sejam eles de agentes ou objetos. Estas mensagens são enviadas pela entidade criadora do papel (agente ou sub-organização) e recebidas pela entidade que vai desempenhar o papel criado. O estereótipo <<role_cancel>> rotula a mensagem de destruição de um papel, seja ele de agente ou objeto. Estas mensagens são enviadas pela entidade destruidora do papel (agente ou sub-organização) e recebidas pela entidade que desempenhava o papel destruído (o próprio agente ou sub-organização, no caso de papel de agente, ou o objeto, no caso de papel de objeto).

A dinâmica dos papéis inclui dois tipos de estados, além dos estados “criado” e “destruído”. São os estados “ativo” e “inativo”. A fim de representar a dinâmica destes estados, foram definidos os estereótipos <<role_activate>> e <<role_deactivate>>. O estereótipo <<role_deactivate>> rotula a mensagem de desativação de um papel de agente cujo estado estava ativo. O estereótipo <<role_activate>> rotula a mensagem de ativação de um papel de agente cujo estado estava inativo. Em ambos os casos, a mensagem rotulada é enviada e recebida por um agente ou sub-organização habilitado a desempenhar o papel ativado ou desativado. Ainda considerando as especificidades da dinâmica dos papéis, existe a possibilidade de um agente trocar de papel ao longo de seu ciclo de vida. O estereótipo <<role_change>> foi definido para possibilitar a modelagem desta alteração de papéis. Neste caso, a mensagem rotulada é enviada por um agente ou sub-organização habilitado a desempenhar ambos os papéis.

A partir da descrição do metamodelo de MAS-ML e da descrição do TAO (Capítulo 4) é feita a aplicação do Observed-MAS para MAS-ML.

5.3.

Fase F1: Estruturação e análise das propriedades de SMAs e dos diagramas MAS-ML

A instanciação da fase F1 foi conduzida a partir da definição da ontologia Ont1, apresentada na seção 5.3.1, seguida da definição das consultas (QV1) sobre violações às propriedades internas aos diagramas MAS-ML, apresentadas

na seção 5.3.2 e, finalmente, pela definição das consultas QD1, cujas respostas sugerem boas práticas de modelagem para a construção de diagramas MAS-ML (5.3.3).

5.3.1.

A ontologia Ont1: descrição das entidades de SMAs

Alguns aspectos da ontologia Ont1 foram apresentados durante a descrição geral do Observed-MAS (4.4.1). Na aplicação do método usando MAS-ML, foi feita a integração das ontologias usadas para descrever o TAO com os conceitos e propriedades obtidos através do metamodelo da linguagem MAS-ML. Considerando-se que o metamodelo de MAS-ML foi baseado no TAO (5.2), vários conceitos definidos no metamodelo foram mapeados diretamente para conceitos definidos no Capítulo 4, como entidades ou relacionamentos do domínio de SMAs. Na Figura 47 apresenta-se tais conceitos e relacionamentos, explicitando a hierarquia entre eles.

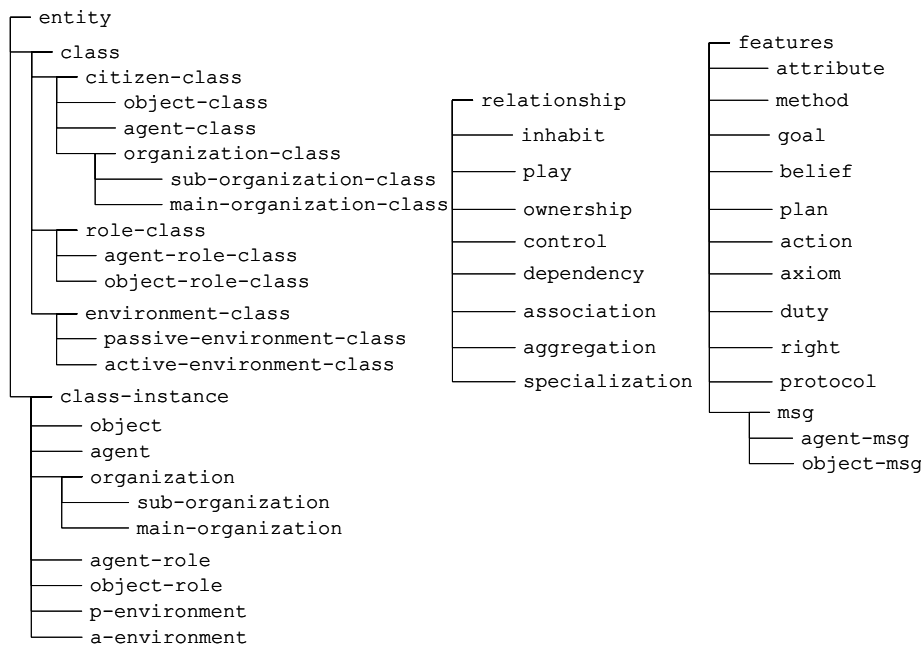


Figura 47. Conceitos comuns ao domínio de SMAs e ao metamodelo MAS-ML

Relativamente aos conceitos específicos das linguagens de modelagens, foi introduzido o conceito `model` para representar os modelos descritos na linguagem de modelagem, o qual foi refinado para acolher os diferentes tipos de diagramas previstos pelo metamodelo de MAS-ML. Assim, foram definidos os sub-conceitos `static-model` e `dynamic-model`. Considerando-se a estrutura

dos diagramas descritos por estes conceitos, foram definidas as propriedades `has-relationship` e `has-class` para permitir a definição de consultas que analisem violações às propriedades internas dos diagramas estáticos. Ambas propriedades possuem `static-model` como domínio e `class` como imagem. Na Figura 48 é apresentada a definição destes conceitos e propriedades na ontologia Ont1.

```
(signature
  :atomic-concepts(... model static-model dynamic-model...)
  :roles(... (has-relationship :domain static-model
                              :range relationship)
            (has-class :domain static-model
                      :range class)...)
  (implies static-model model)
  (implies dynamic-model model)
  (disjoint static-model dynamic-model))
```

Figura 48. Conceitos integrados à primeira ontologia do TAO.

Além destas propriedades, também foram definidas as propriedades `has-path` e `has-msg` para analisar violações às propriedades internas de diagramas dinâmicos. Outros conceitos foram definidos para posterior refinamento, tais como `path`, `stereotyped-msg`, `sequence` e `seq-element`. Estes conceitos e seus refinamentos serão descritos nas subseções seguintes. As subseções 5.3.1.1 e 5.3.1.2 apresentam os conceitos e as propriedades de Ont1 relacionados aos diagramas estáticos e dinâmicos de MAS-ML, respectivamente.

5.3.1.1.

Conceitos relacionados aos diagramas estáticos definidos pelo metamodelo de MAS-ML

Os diagramas estáticos definidos pelo metamodelo de MAS-ML são os diagramas de classe, de organização e de papéis. Na ontologia Ont1 foram definidos os conceitos `class-model`, `organization-model` e `role-model`, todos sub-conceitos de `static-model`. Os outros conceitos relacionados aos diagramas estáticos são oriundos do domínio de SMAs, definidos pelos sub-conceitos do conceito `class` e pelo sub-conceitos do conceito `relationship`.

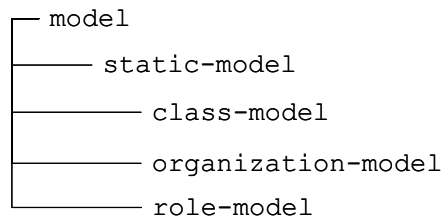


Figura 49. Refinamento do conceito static-model.

5.3.1.2.

Conceitos relacionados aos diagramas dinâmicos previstos no metamodelo de MAS-ML

O metamodelo de MAS-ML descreve, atualmente, dois diagramas dinâmicos: o diagrama de sequência e o diagrama de atividades. Nesta tese considerou-se apenas o diagrama de sequência, pois o diagrama de atividades foi incorporado ao metamodelo recentemente (Silva et al, 2005). O diagrama de sequência é usado para descrever tanto os processos dinâmicos primitivos (criação e destruição das entidades) quanto os processos dinâmicos complexos (dinâmica dos papéis). Além disso, diagramas de sequência também são usados para descrever a estrutura de protocolos e planos.

Diagrama de sequência

Um diagrama de sequência é composto por identificadores de entidades (classes e instâncias de classes) e mensagens. No metamodelo de MAS-ML foi definido, para cada entidade, um *pathname*. Um *pathname* é o identificador que distingue uma entidade de outra. Desta forma, ele descreve a entidade pelo nome a ela atribuído e o contexto em que está inserida. Em UML, um *pathname* é composto por duas partes: uma relacionada à classe do objeto e a outra relacionada à instância da classe (Booch et al, 1999). O metamodelo MAS-ML adota, para a definição de *pathname*, o mesmo critério de contextualização usado em UML, aplicado ao domínio de SMAs, onde entidades habitam ambientes, e desempenham papéis em organizações (Figura 50).

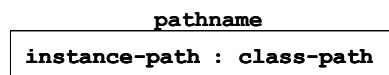


Figura 50. Estrutura de um pathname

Na ontologia Ont1 foi definido o conceito `sequence-model`, para representar os diagramas de sequência, como sub-conceito de `dynamic-model`. O conceito mais abstrato `dynamic-model` foi mantido a fim de permitir a introdução de novos diagramas dinâmicos, como, por exemplo, o diagrama de atividades. Por conseguinte, é possível acrescentar novos sub-conceitos ao conceito `dynamic-model`, sem alterar a estrutura de Ont1.

Além deste conceito, também foram definidos os conceitos `path`, `class-path` e `instance-path`, representando o `pathname`, o `pathname` restrito à identificação de classe e o `pathname` restrito à identificação da instância da classe, respectivamente. Considerando a estrutura de `pathnames` (Figura 50) foi definido o conceito `path` como a união disjunta dos conceitos `class-path` e `instance-path`. A união é disjunta, pois não há interseção entre entidades representadas por classes (e identificadas por `class-path`) e entidades representadas por instâncias de classes. Observe que uma instância de classe pode ser identificada de duas formas: pelo `path` completo (`instance-path: class-path`) ou pelo `instance-path`. A Figura 51 ilustra a estrutura usada para definir a semântica do conceito `path` em Ont1.

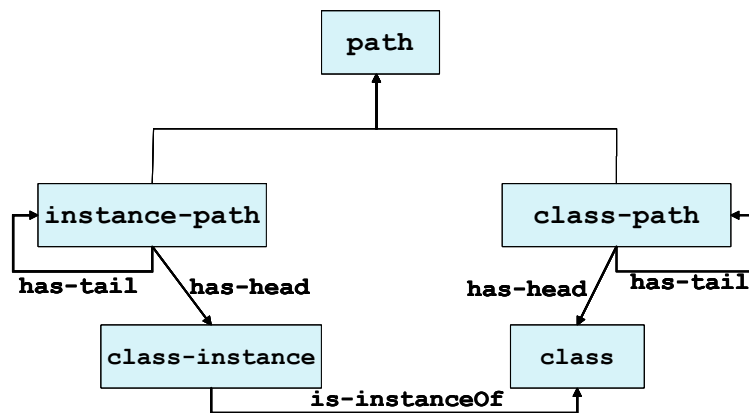


Figura 51. Estrutura usada na definição do conceito `path` em Ont1.

As propriedades `has-head` e `has-tail` (Figura 51) foram definidas para permitir a descrição da semântica do conceito `path`. A propriedade `has-head` indica a `class-instance` ou a `class` à qual a entidade identificada pertence, e a propriedade `has-tail` indica o contexto no qual a entidade está inserida. A Figura 52 ilustra os identificadores definidos na ontologia Ont1, os quais são descritos a seguir.

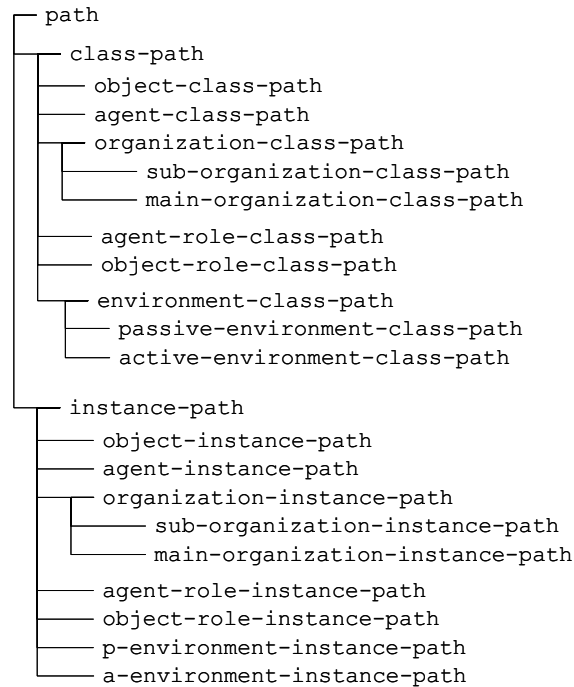


Figura 52. Conceitos relacionados ao *pathname* das entidades descritas pelo metamodelo MAS-ML

Identificadores de objeto

De acordo com o metamodelo de MAS-ML, o identificador completo de um objeto deve descrever o objeto, o papel que ele pode desempenhar, a organização à qual o papel pertence e o ambiente que ele habita. Sua descrição é ilustrada na Figura 53.

```

object/object-role/organization/environment:
object-class/object-role-class/organization-class/environment-class
  
```

Figura 53. Identificador de objetos.

Na ontologia Ont1 os identificadores foram definidos recursivamente. Foram definidos os conceitos *object-class-path* e *object-instance-path* como sub-conceitos de *class-path* e *instance-path*, respectivamente. A semântica do identificador de objetos é descrita na Figura 54.

```

(implies object-class-path class-path)
(implies object-instance-path instance-path)
(implies object-class-path (and (all has-head object-class)
                                (some has-tail object-role-class-path)))
(implies object-instance-path
  (and (all has-head object)
        (some has-tail object-role-instance-path)))
  
```

Figura 54. Identificador de objetos definido na ontologia Ont1.

Identificador de agente

De acordo com o metamodelo de MAS-ML, o identificador completo de um agente deve descrever o agente, o papel que ele desempenha, a organização dona do papel e o ambiente que ele habita. Sua descrição é ilustrada na Figura 55.

```
agent/agent-role/organization/environment:
agent-class/agent-role-class/organization-class/environment-class
```

Figura 55. Identificador de agentes.

Na ontologia Ont1 foram definidos os conceitos `agent-class-path` e `agent-instance-path` como sub-conceitos de `class-path` e `instance-path`, respectivamente. A semântica do identificador de agentes é descrita na Figura 56.

```
(implies agent-class-path class-path)
(implies agent-instance-path instance-path)
(implies agent-class-path (and (all has-head agent-class)
                                (some has-tail agent-role-class-path)))
(implies agent-instance-path (and (all has-head agent)
                                   (some has-tail agent-role-instance-path)))
```

Figura 56. Identificador de agentes definido na ontologia Ont1.

Identificador de organização

De acordo com o metamodelo de MAS-ML, o identificador completo de uma organização deve considerar se a organização é do tipo sub-organização ou organização principal. Se for sub-organização, o identificador completo deve descrever a sub-organização, o papel que ela desempenha, a organização dona do papel e o ambiente que ela habita. Se for organização principal, o identificador deve descrever a organização e o ambiente que ela habita. As descrições de ambos são ilustradas na Figura 57.

```
sub-organization/agent-role/organization/environment:
sub-organization-class/agent-role-class/organization-class/environment-class

main-organization/environment:main-organization-class/environment-class
```

Figura 57. Identificadores de organizações.

Na ontologia Ont1 foram definidos os conceitos `organization-class-path` e `organization-instance-path` como sub-conceitos de `class-path` e `instance-path`, respectivamente. A partir destes conceitos definiu-se os conceitos `sub-organization-class-path` e `sub-organization-instance-path` para representar os identificadores de sub-organizações, e `main-organization-class-path` e `main-organization-instance-path` para representar os identificadores de organizações principais. A semântica dos identificadores de sub-organizações e de organizações principais é descrita na Figura 58.

```
(implies organization-class-path class-path)
(implies organization-instance-path instance-path)

(implies sub-organization-class-path organization-class-path)
(implies sub-organization-instance-path organization-instance-path)

(implies main-organization-class-path organization-class-path)
(implies main-organization-instance-path organization-instance-path)

(implies sub-organization-class-path
  (and (all has-head sub-organization-class)
        (some has-tail agent-role-class-path)))

(implies sub-organization-instance-path
  (and (all has-head sub-organization)
        (some has-tail agent-role-instance-path)))

(implies main-organization-class-path
  (and (all has-head main-organization-class)
        (some has-tail environment-class-path)))

(implies main-organization-instance-path
  (and (all has-head main-organization)
        (some has-tail environment-instance-path)))
```

Figura 58. Identificadores de organizações definidos na ontologia Ont1.

Identificador de papel de agente

De acordo com o metamodelo de MAS-ML, o identificador completo de um papel de agente deve descrever o papel de agente, a organização dona do papel e o ambiente que a organização habita. Sua descrição é ilustrada na Figura 59.

```
agent-role/organization/environment:
agent-role-class/organization-class/environment-class
```

Figura 59. Identificador de papel de agente

Na ontologia Ont1 foram definidos os conceitos `agent-role-class-path` e `agent-role-instance-path` como sub-conceitos de `class-path` e `instance-path`, respectivamente. A semântica do identificador de papéis de agentes é descrita na Figura 60.

```
(implies agent-role-class-path class-path)
(implies agent-role-instance-path instance-path)
(implies agent-role-class-path (and (all has-head agent-role-class)
                                     (some has-tail organization-class-path)))
(implies agent-role-instance-path
  (and (all has-head agent-role)
        (some has-tail organization-instance-path)))
```

Figura 60. Identificador de papéis de agentes definido na ontologia Ont1.

Identificador de papel de objeto

De acordo com o metamodelo de MAS-ML, o identificador completo de um papel de objeto deve descrever o papel do objeto, a organização dona do papel e o ambiente que a organização habita. Sua descrição é ilustrada na Figura 61.

```
object-role/organization/environment:
object-role-class/organization-class/environment-class
```

Figura 61. Identificadores de papel de objeto.

Na ontologia Ont1 foram definidos os conceitos `object-role-class-path` e `object-role-instance-path` como sub-conceitos de `class-path` e `instance-path`, respectivamente. A semântica do identificador de papel de objeto é descrita na Figura 62.

```
(implies object-role-class-path class-path)
(implies object-role-instance-path instance-path)

(implies object-role-class-path (and (all has-head object-role-class)
                                     (some has-tail organization-class-path)))
(implies object-role-instance-path (and (all has-head object-role)
                                       (some has-tail organization-instance-path)))
```

Figura 62. Identificador de papéis de objetos definido na ontologia Ont1.

Identificador de ambiente

De acordo com o metamodelo de MAS-ML, o identificador completo de um ambiente é dado apenas pelo nome da instância do ambiente seguido do nome da classe de ambiente que gera a instância. Sua descrição é ilustrada na Figura 63.

```
environment:environment-class
```

Figura 63. Identificadores de ambientes.

Na ontologia Ont1 foram definidos os conceitos `environment-class-path` e `environment-instance-path` como sub-conceitos de `class-path` e `instance-path`, respectivamente, para representar identificadores de ambientes em geral. A fim de diferenciar ambientes ativos de ambientes passivos, foram criados os conceitos `active-environment-class-path` e `a-environment-instance-path` e `passive-environment-class-path` e `p-environment-instance-path`. A semântica dos identificadores de ambientes é descrita na Figura 64.

```
(implies environment-class-path class-path)
(implies environment-instance-path instance-path)

(implies active-environment-class-path environment-class-path)
(implies a-environment-instance-path environment-instance-path)

(implies passive-environment-class-path environment-class-path)
(implies p-environment-instance-path environment-instance-path)

(implies active-environment-class-path
  (and (all has-head active-environment-class)
        (some has-tail *bottom*)))
(implies a-environment-instance-path
```

```
(and (all has-head a-environment)
      (some has-tail *bottom*))

(implies passive-environment-class-path
  (and (all has-head passive-environment-class)
        (some has-tail *bottom*)))

(implies p-environment-instance-path
  (and (all has-head p-environment)
        (some has-tail *bottom*)))
```

Figura 64. Identificadores de ambientes definido na ontologia Ont1.

Além dos identificadores já descritos, diagramas de sequência também possuem mensagens trocadas por entidades que são descritas através dos identificadores. Estas mensagens foram definidas como sub-conceitos do conceito `msg` além dos já definidos `agent-msg` e `object-msg` (Capítulo 4). A seguir são descritos os conceitos associados às mensagens utilizadas em diagramas de sequência, sejam elas relativas aos processos dinâmicos primitivos ou complexos, a especificação de protocolos ou a simples chamada de método.

As mensagens associadas aos processos dinâmicos primitivos e complexos são descritas no metamodelo de MAS-ML fazendo-se uso do mecanismo de estereótipo, previsto no metamodelo de UML. Na ontologia Ont1 estas mensagens são representadas através do conceito `stereotyped-msg`, definido como sub-conceito de `msg`. As mensagens representadas pelo conceito `stereotyped-msg` estão classificadas, pelo metamodelo de MAS-ML, em mensagens relacionadas aos processos de criação e destruição de entidades ou a processos relacionados à dinâmica dos papéis.

Mensagens relacionadas aos processos de criação e destruição de entidades

Com exceção da criação de papéis, a criação das outras entidades previstas no metamodelo de MAS-ML é descrita por uma mensagem rotulada com o estereótipo `<<create>>` e é representada na ontologia Ont1 pelo conceito `create-msg`, sub-conceito de `stereotyped-msg`. Analogamente, a destruição das entidades é definida no metamodelo de MAS-ML pelo estereótipo `<<destroy>>` e é representada na ontologia Ont1 pelo conceito `destroy-msg`, sub-conceito de `stereotyped-msg`.

Mensagens relacionadas à dinâmica dos papéis

A fim de descrever a dinâmica dos papéis, o metamodelo de MAS-ML especifica alguns estereótipos para rotular mensagens decorrentes da dinâmica dos papéis. O estereótipo `<<role_commitment>>`, indica criação de um determinado papel, e é representado na ontologia Ont1 pelo conceito `role-commit-msg`, sub-conceito de `stereotyped-msg`. O estereótipo `<<role_cancel>>` indica a destruição de um papel e é representado na ontologia Ont1 pelo conceito `role-cancel-msg`, sub-conceito de `stereotyped-msg`. O estereótipo `<<role_activate>>` indica a ativação de um papel já criado e é representado na ontologia Ont1 pelo conceito `role-activate-msg`, sub-conceito de `stereotyped-msg`. O estereótipo `<<role_deactivate>>` indica a desativação de um papel que estava ativo e é representado na ontologia Ont1 pelo conceito `role-deactivate-msg`, sub-conceito de `stereotyped-msg`. Finalmente, o estereótipo `<<role_change>>` indica a troca de papéis durante uma interação, e é representado na ontologia Ont1 pelo conceito `role-change-msg`, sub-conceito de `stereotyped-msg`. A Figura 65 apresenta a hierarquia de conceitos definidos a partir do conceito `msg`.

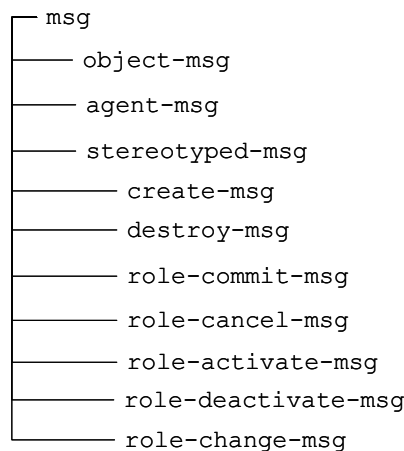


Figura 65. Conceitos relacionados às mensagens descritas no metamodelo de MAS-ML.

5.3.2.

Consultas sobre as propriedades intra-diagramas MAS-ML (QV1)

A instanciação da segunda etapa da fase F1 consistiu da definição de consultas que retornam como resposta violações às propriedades internas dos diagramas MAS-ML. Estas propriedades referem-se a boa formação dos

diagramas e, conseqüentemente, dos modelos por eles gerados. Para que modelos gerados a partir de diagramas estáticos sejam bem formados, as entidades e os relacionamentos previstos no metamodelo da linguagem de descrição do diagrama, e somente eles, devem fazer parte deste modelo.

As subseções 5.3.2.1, 5.3.2.2, 5.3.2.3 e 5.3.2.4 apresentam, para cada tipo de diagrama previsto no metamodelo de MAS-ML, exemplos de propriedades que não podem ser violadas, e as respectivas consultas da etapa QV1.

5.3.2.1.

Consultas sobre propriedades internas de diagramas de classe

As consultas relacionadas às propriedades internas de diagramas de classe referem-se às entidades e relacionamentos previstos para este diagrama pelo metamodelo e as eventuais restrições que algum dos relacionamentos permitidos possam sofrer, enquanto modelados neste diagrama. Como exemplos de propriedades a serem analisadas quanto às suas violações, considere as propriedades Propriedade LM. 1, Propriedade LM. 2 e Propriedade LM. 3, onde LM significa linguagem de modelagem.

Propriedade LM. 1 *As entidades participantes de um diagrama de classe são indivíduos dos conceitos* `object-class`, `agent-class`, `organization-class`, e `environment-class`.

A consulta descrita na Figura 66 ilustra uma violação à Propriedade LM. 1, já que classes do tipo `role-class` não são permitidas neste diagrama.

```
(retrieve (?rolecl ?clmd)
  (and (?rolecl role-class)
    (?clmd class-model)
    (?clmd ?rolecl has-class)))
```

Figura 66. Consulta sobre entidades não permitidas em diagramas de classe

Propriedade LM. 2 *Os relacionamentos permitidos num diagrama de classe são* `inhabit`, `specialization`, `aggregation`, `dependency` e `association`.

A consulta descrita na Figura 67 ilustra uma violação à Propriedade LM. 2, já que relacionamentos dos tipos `play`, `ownership` e `control` não são permitidos neste diagrama.

```
(retrieve (?no-allowed-rel ?clmd)
  (and (?no-allowed-rel (or play ownership control))
    (?clmd class-model)
    (?clmd ?no-allowed-rel has-relationship)))
```

Figura 67. Consulta sobre relacionamento não permitido em diagramas de classe

O metamodelo de MAS-ML também especifica algumas restrições aos relacionamentos permitidos nos diagramas de classe. Como exemplo, cita-se a restrição ao relacionamento *association*, descrita pela Propriedade LM. 3.

Propriedade LM. 3 *Num diagrama de classe, o relacionamento association só é permitido entre indivíduos do conceito object-class e indivíduos do conceito citizen-class; ou entre indivíduos do conceito environment-class.*

A consulta descrita na Figura 68 ilustra uma violação à Propriedade LM. 3.

```
(retrieve (?ass ?clmd)
  (and (?ass association)
    (?clmd class-model)
    (?clmd ?ass has-relationship)
    (?end1 ?ass is-end1)
    (?end2 ?ass is-end2)
    (or (and (?end1 (or agent-class organization-class))
      (?end2 (or agent-class organization-class
        environment-class)))
      (and (?end1 environment-class)
        (?end2 citizen-class))
      (?end1 role-class)
      (?end2 role-class))))
```

Figura 68. Consulta sobre restrição a relacionamento permitido em diagramas de classe

As demais consultas relacionadas às propriedades internas de diagramas de classe são encontradas no Apêndice B.

5.3.2.2.

Consultas sobre propriedades internas de diagramas de organização

As entidades permitidas em diagramas de organização são todos os tipos de classes, logo não há restrições deste tipo. Por outro lado, apenas os relacionamentos *ownership*, *play* e *inhabit* são permitidos neste tipo de diagrama. Um exemplo de propriedade que deve ser analisada neste sentido é a Propriedade LM. 4.

Propriedade LM. 4 *Num diagrama de organização os relacionamentos permitidos são do tipo* `ownership`, `play` *e* `inhabit`.

A consulta descrita na Figura 69 ilustra uma violação à Propriedade LM. 4.

```
(retrieve (?not-allowed-rel ?orgmd)
  (and (?orgmd organization-model)
    (?not-allowed-rel relationship)
    (?orgmd ?not-allowed-rel has-relationship)
    (not(?not-allowed-rel (or ownership inhabit play))))))
```

Figura 69. Consulta sobre relacionamento não permitido em diagramas de organização

O metamodelo de MAS-ML também especifica algumas restrições aos relacionamentos entre entidades participantes dos diagramas de organizações. Uma destas restrições é descrita na Propriedade LM. 5.

Propriedade LM. 5 *Num diagrama de organizações não estão definidos relacionamentos entre indivíduos do conceito* `agent-class`.

A consulta descrita na Figura 70 ilustra uma violação à Propriedade LM. 5.

```
(retrieve (?rel ?orgmd)
  (and (?rel relationship)
    (?orgmd organization-model)
    (?orgmd ?rel has-relationship)
    (?end1 ?rel is-end1)
    (?end2 ?rel is-end2)
    (?end1 agent-class) (?end2 agent-class)))
```

Figura 70. Consulta sobre restrição a relacionamentos entre agentes em diagramas de organizações.

As demais consultas relacionadas às propriedades internas de diagramas de organizações são encontradas no Apêndice B.

5.3.2.3. Consultas sobre propriedades internas de diagramas de papéis

As entidades participantes de um diagrama de papéis são indivíduos dos conceitos `object-class`, `agent-role-class` e `object-role-class`. Um exemplo de propriedade que deve ser analisada é a descrita na Propriedade LM. 6.

Propriedade LM. 6 *Num diagrama de papéis são permitidos apenas indivíduos dos conceitos* `role-class` e `object-class`.

A consulta descrita na Figura 71 ilustra uma violação à Propriedade LM. 6.

```
(retrieve (?not-allowed-cl ?rlmd)
  (and (?not-allowed-cl (or agent-class organization-class
                           environment-class))
    (?rlmd role-model)
    (?rlmd ?not-allowed-cl has-class)))
```

Figura 71. Consulta sobre entidades não permitidas em diagramas de papéis.

Os relacionamentos permitidos num diagrama de papéis estão descritos na Propriedade LM. 7.

Propriedade LM. 7 *Num diagrama de papéis, os relacionamentos permitidos são control, specialization, aggregation, dependency e association.*

A consulta descrita na Figura 72 ilustra uma violação à Propriedade LM. 7.

```
(retrieve (?no-allowed-rel ?rlmd)
  (and (?no-allowed-rel (or ownership play inhabit))
    (?rlmd role-model)
    (?rlmd ?no-allowed-rel has-relationship)))
```

Figura 72. Consulta sobre relacionamento não permitido em diagramas de papéis.

O metamodelo de MAS-ML também especifica algumas restrições aos relacionamentos permitidos nos diagramas de papéis. Um exemplo disto é especificado na Propriedade LM. 8.

Propriedade LM. 8 *Num diagrama de papéis, o relacionamento specialization só é permitido entre indivíduos do conceito object-role-class; ou entre indivíduos do conceito agent-role-class.*

A consulta descrita na Figura 73 ilustra uma violação à Propriedade LM. 8

```
(retrieve (?spec ?end1 ?end2 ?rlmd)
  (and (?spec specialization)
    (?rlmd role-model)
    (?rlmd ?spec has-relationship)
    (?end1 ?spec is-end1)
    (?end2 ?spec is-end2)
    (or (?end1 (or citizen-class environment-class))
      (?end2 (or citizen-class environment-class))
      (and (?end1 agent-role-class)
        (not (?end2 agent-role-class)))
      (and (?end1 object-role-class)
        (not (?end2 object-role-class))))))
```

Figura 73. Consulta sobre restrição a relacionamento permitido em diagramas de papéis.

As consultas relacionadas às propriedades internas de diagramas de papéis são encontradas no Apêndice B.

5.3.2.4.

Consultas sobre propriedades internas de diagramas de sequência

Diagramas de sequência são compostos por `path` e `msg`, e possuem uma estrutura (representada pelo conceito `sequence`), que descreve a ordem na qual as `msg` são trocadas. Um `path` representa o identificador de uma determinada entidade de um diagrama. No `path` estão as informações relacionadas à classe e/ou instância da classe que gerou a entidade, indicando o contexto em que ela está inserida. Uma `msg` representa mensagens trocadas entre as entidades que participam de um SMA.

Como qualquer `path` e qualquer `msg` pode participar de um diagrama de sequência, as consultas sobre violações às propriedades internas deste tipo de diagrama referem-se, principalmente, a restrições relacionadas a remetentes e destinatários das mensagens trocadas e descritas num diagrama de sequência.

Um exemplo de restrição relacionada a remetente e destinatário de mensagens descritas em um diagrama de sequência é descrito pela Propriedade LM. 9.

Propriedade LM. 9 *Num diagrama de sequência um agente não pode receber mensagem enviada por um objeto, isto é, um `path` que identifica um agente não pode receber mensagem enviada por um `path` que identifica um objeto.*

A consulta descrita na Figura 74 ilustra uma violação à Propriedade LM. 9.

```
(retrieve (?msg ?obj-sender ?agent-receiver ?seq-md)
  (and (?seq-md sequence-model)
    (?obj-sender object-instance-path)
    (?agent-receiver agent-instance-path)
    (?seq-md ?msg has-msg)
    (?obj-sender ?msg is-msg-sender)
    (?agent-receiver ?msg is-msg-receiver)))
```

Figura 74. Consulta sobre restrição sobre remetentes de mensagens enviadas para agentes em diagramas de sequência.

Diagramas de sequência também descrevem protocolos, que são responsáveis pela interação entre agentes e/ou sub-organizações. Protocolos possuem `path` do tipo `class-path`, pois estão definidos no contexto de classes de papel de agente, classes de organizações principais ou classes de ambientes ativos. As mensagens que compõem um protocolo são do tipo

agent-msg, pois estas são as mensagens que tornam possível a interação entre agentes. A Propriedade LM. 10 define os path que modelam protocolos.

Propriedade LM. 10 *Num diagrama de seqüência que modela a estrutura de um protocol, tanto o remetente como o destinatário da mensagem são identificados através de agent-role-class-path, ou main-organization-class-path ou active-environment-class-path.*

A consulta descrita na Figura 75 ilustra violação à Propriedade LM. 10.

```
(retrieve (?msg-end ?seq-md ?prtcl)
  (and (?msg agent-msg)
    (?seq-md sequence-model)
    (?prtcl protocol)
    (?struct sequence)
    (?prtcl ?struct has-structure)
    (?seq-md ?struct has-structure)
    (?seq-md ?msg has-msg)
    (?msg-end path)
    (?seq-md ?msg-end has-path)
    (?msg-end ?msg is-msg-end)
    (not (?msg-end (or agent-role-class-path
                      main-organization-class-path
                      active-environment-class-path))))
```

Figura 75. Consulta sobre restrição a remetente de mensagem em diagramas de seqüência.

Outro exemplo de propriedade interna de diagrama de seqüência que modela protocolos é dado pela Propriedade LM. 11.

Propriedade LM. 11 *Num diagrama de seqüência que modela protocolos, as entidades participantes precisam conhecer o mesmo protocolo.*

A consulta descrita na Figura 76 ilustra violação à Propriedade LM. 11.

```
(retrieve (?seqmd ?role-without-prtcl ?prtcl)
  (and (?seqmd sequence-model)
    (?role-without-prtcl agent-role-class)
    (?prtcl protocol)
    (?role-path agent-role-class-path)
    (?seqmd ?role-path has-path)
    (?role-path ?role-without-prtcl has-head)
    (?struct sequence)
    (?seqmd ?struct has-structure)
    (?prtcl ?struct has-structure)
    (not (?role-without-prtcl ?prtcl has-protocol))))
```

Figura 76. Consulta sobre entidades com protocolos distintos em diagrama de seqüência.

As demais consultas relacionadas às propriedades internas de diagramas de seqüência são encontradas no Apêndice B.

5.3.3.

Consultas referentes às sugestões de boas práticas de modelagem usando MAS-ML (QD1)

A instanciação da terceira etapa da fase F1 consistiu da definição de consultas que retornam como resposta sugestões de boas práticas de modelagem de SMAs usando diagramas MAS-ML. Com o objetivo de ilustrar uma consulta QD1, considere a seguinte restrição relativa à interação entre agentes: a interação entre agentes ocorre através de protocolos. O metamodelo de MAS-ML especifica que protocolos estão “embutidos” nos papéis de agentes que cada agente desempenha. Desta forma é interessante fornecer ao desenvolvedor a informação sobre quais classes de papel de agente possuem protocolos comuns e estão modeladas em algum diagrama de papel sem relacionamentos modelados entre elas. Esta informação decorre de uma propriedade de boas práticas de modelagem para diagramas MAS-ML, especificada na Propriedade BP 1.

Propriedade BP 1. *Classes de papel de agente modeladas em diagramas de papel e que possuem protocolos comuns devem, em geral, estar relacionadas através de relacionamentos do tipo control, association, dependency ou aggregation.*

Observe que não há erro em não existir, num diagrama de papéis, um relacionamento entre duas classes de papel de agente que possuem protocolos comuns, porém cabe ao desenvolvedor decidir se agentes que desempenham aqueles papéis vão ou não interagir e, portanto se vai ou não definir o relacionamento entre elas. Esta informação é especialmente útil durante a construção de diagramas de papéis. A combinação das consultas ilustradas na Figura 77 permite ao desenvolvedor analisar, dentre os pares de papéis que possuem protocolos comuns, quais estão relacionados no diagrama de papel. Com estas informações o desenvolvedor pode decidir sobre quais outros pares de papéis ele deseja relacionar.

```
(retrieve (?role-1 ?role-2 ?prtcl ?rlmd)
  (and (?role-1 agent-role-class)
    (?role-2 agent-role-class)
    (?prtcl protocol)
    (?rlmd role-model)
    (?rlmd ?role-1 has-class)
    (?rlmd ?role-2 has-class)
    (?role-1 ?prtcl has-protocol)
```

```

        (?role-2 ?prtcl has-protocol)))
(retrieve (?role-1 ?role-2 ?prtcl ?rlmd)
  (and (?role-1 agent-role-class)
    (?role-2 agent-role-class)
    (?prtcl protocol)
    (?rlmd role-model)
    (?rlmd ?role-1 has-class)
    (?rlmd ?role-2 has-class)
    (?role-1 ?prtcl has-protocol)
    (?role-2 ?prtcl has-protocol)
    (?rel (or control association
      dependency aggregation))
    (?rel ?role-1 has-end1)
    (?rel ?role-2 has-end)))

```

Figura 77. Consulta que indica pares de papéis de agente que possuem protocolos comuns.

Outro exemplo de consulta que gera sugestão de boas práticas de modelagem, durante a construção de diagramas de organizações, refere-se à seguinte característica de agência: agentes são definidos para atingir metas enquanto desempenham papéis. Estes papéis são aqueles que possibilitem ao agente cumprir alguma de suas metas, parcial ou completamente. Sendo assim, dado um diagrama de organizações, é interessante mostrar ao desenvolvedor as classes de agentes e de papéis de agentes que estão definidas no diagrama, possuem metas comuns, mas não estão relacionadas através de um relacionamento *play*.

Esta informação decorre de uma propriedade de boas práticas de modelagem para diagramas MAS-ML, especificada na Propriedade BP 2.

Propriedade BP 2. *Classes de agente e de papel de agente modeladas em diagramas de organização e que possuem metas comuns devem, em geral, estar relacionadas através de relacionamentos do tipo *play*.*

Na Figura 78 é apresentada uma consulta que retorna como resposta esta sugestão. Outras consultas relacionadas às sugestões de boas práticas de modelagem são encontradas no Apêndice E.

```

(retrieve (?ag-cl ?ag-rl-cl ?org-md)
  (and (?ag-cl agent-class)
    (?ag-rl-cl agent-role-class)
    (?org-md organization-model)
    (?org-md ?ag-rl-cl has-class)
    (?org-md ?ag-cl has-class)
    (?ag-rl-cl ?goal has-goal)
    (?ag-cl ?goal has-goal)
    (?pl play)

```

```
(not (and (?pl ?ag-cl has-end1)
           (?pl ?ag-rl-cl has-end2))))
```

Figura 78. Consulta que indica agentes e papéis de agente que possuem mesmas metas.

5.4.

FASE F2: Estruturação e análise das propriedades de interdependência entre diagramas MAS-ML

A instanciação da fase F2 foi conduzida a partir da definição da ontologia Ont2, apresentada na seção 5.4.1, seguida da definição das consultas (QV2) sobre violações às propriedades de interdependência entre os diagramas MAS-ML, apresentadas na seção 5.4.2 e, finalmente, pela definição das consultas QD2, cujas respostas sugerem boas práticas de modelagem a fim de melhorar sua qualidade considerando-se todos os diagramas que a compõe a modelagem de um SMA (5.4.3).

5.4.1.

A ontologia Ont2

Alguns aspectos da ontologia Ont2 foram descritos no Capítulo 4, quando foi feita a descrição geral do Observed-MAS. Estes aspectos descrevem as propriedades do TAO. Na aplicação do método para MAS-ML, foram recepcionados os axiomas que descrevem as propriedades do domínio de SMAs e acrescentados axiomas referentes às propriedades internas dos diagramas MAS-ML. Estes axiomas refletem o complementar do conteúdo analisado pelas consultas QV1.

Os diagramas de classe são compostos pelas entidades representadas por indivíduos dos conceitos `object-class`, `agent-class`, `organization-class` e `environment-class` (Propriedade LM. 1). Os relacionamentos permitidos para este diagrama são `inhabit`, `association`, `specialization` e `aggregation` (Propriedade LM. 2). Estas propriedades são analisadas, respectivamente, pelas consultas ilustradas na Figura 66, e na Figura 67. Os axiomas que as descreve são ilustrados na Figura 79, juntamente com as restrições impostas a cada relacionamento, quando participante deste diagrama.

As restrições impostas aos relacionamentos são: o relacionamento `inhabit` é restrito a entidades que representam indivíduos do conceito `object-class`, o relacionamento `association` ocorre entre as entidades que representam indivíduos do conceito `object-class` e indivíduos dos conceitos `citizen-class`; ou `environment-class`. O relacionamento `specialization` ocorre entre entidades de mesmo tipo, e os relacionamentos `aggregation` e `dependency` são restritos às entidades `object-class`.

```
(implies class-model
  (all has-class (or citizen-class environment-class)))

(implies class-model
  (some has-relationship
    (and inhabit (all has-end1 object-class)))))

(implies class-model (some has-relationship
  (and association
    (or (and (some has-end1
      (or citizen-class environment-class))
      (some has-end2 object-class))
      (and (some has-end1 object-class)
      (some has-end2
        (or citizen-class environment-class)))))))

(implies class-model (some has-relationship
  (or (and specialization
      (some has-end1 organization-class)
      (all has-end2 organization-class))
      (and specialization (some has-end1 agent-class)
      (all has-end2 agent-class))
      (and specialization (some has-end1 object-class)
      (all has-end2 object-class))
      (and specialization
      (some has-end1 environment-class)
      (all has-end2 environment-class))))))

(implies class-model (some has-relationship
  (and aggregation (some has-end1 object-class)
    (all has-end2 object-class))))

(implies class-model (some has-relationship
  (and dependency (some has-end1 object-class)
    (all has-end2 object-class))))
```

Figura 79. Axiomas referentes a diagramas de classe.

Os diagramas de organizações são compostos por entidades representadas por todos os tipos de classes e os relacionamentos permitidos são `play`, `ownership` e `inhabit` (Propriedade LM. 4). Esta propriedade é analisada pela consulta ilustrada na Figura 69. Não há restrições sobre os relacionamentos, pois os mesmos são definidos no contexto de diagramas de

organizações, com exceção do relacionamento `inhabit`. Os axiomas que descrevem a estrutura de diagramas de organizações são apresentados na Figura 80.

```
(implies organization-model (all has-class class))
(implies organization-model (all has-relationship
                             (or ownership play inhabit)))
```

Figura 80. Axiomas referentes a diagramas de organizações.

Os diagramas de papéis são compostos pelas entidades `object-role-class`, `agent-role-class` e `object-class` (Propriedade LM. 6). Os relacionamentos permitidos são `control`, `association`, `dependency`, `aggregation` e `specialization` (Propriedade LM. 7). O relacionamento `dependency` é permitido entre entidades `agent-role-class` e entidades `role-class`; e entre entidades `object-role-class`. Os relacionamentos `aggregation` e `specialization` são permitidos entre entidades `agent-role-class`, ou entre entidades `object-role-class` (Propriedade LM. 8). O relacionamento `association` é permitido entre entidades `object-class` e `role-class`; e entre entidades `role-class`. As consultas que analisam parte destas propriedades são descritas na Figura 71, na Figura 72 e na Figura 73. Os axiomas que descrevem a estrutura de diagramas de papéis são apresentados na Figura 81.

```
(implies role-model
  (all has-class (or object-class role-class)))
(implies role-model (some has-relationship control))
(implies role-model (some has-relationship
  (or (and dependency
        (some has-end1 agent-role-class)
        (all has-end2 role-class))
      (and dependency
        (some has-end1 object-role-class)
        (all has-end2 object-role-class)))))

(implies role-model (some has-relationship
  (or (and aggregation
        (some has-end1 agent-role-class)
        (all has-end2 agent-role-class))
      (and aggregation
        (some has-end1 object-role-class)
        (all has-end2 object-role-class)))))

(implies role-model
  (some has-relationship
    (or (and specialization
          (some has-end1 agent-role-class)
          (all has-end2 agent-role-class))
        (and specialization
          (some has-end1 object-role-class)
```

```

                (all has-end2 object-role-class))))))
(implies role-model (some has-relationship
  (and association
    (or (and (some has-end1 role-class)
              (some has-end2 object-class))
        (and (some has-end1 object-class)
              (some has-end2 role-class))
        (and (some has-end1 role-class)
              (some has-end2 role-class)))))

```

Figura 81. Axiomas referentes a diagramas de papéis.

Diagramas de sequência são compostos por identificadores de entidades, descritos na ontologia Ont1 pelo conceito `path`, por mensagens representadas pelo conceito `msg` e possuem uma estrutura representada pelo conceito `sequence`. Os axiomas que descrevem a estrutura de diagramas de sequência são apresentados na Figura 82.

```

(implies sequence-model (some has-path path))
(implies sequence-model (some has-msg msg))
(implies sequence-model (some has-structure sequence))

```

Figura 82. Axiomas referentes a diagramas de sequência.

A troca de mensagens também é modelada através de diagramas de sequência, porém, as restrições relativas aos remetentes e destinatários de uma mensagem são independentes do diagrama. Na Propriedade LM. 9 foi descrita uma restrição a mensagens recebidas por agentes, e a consulta que analisa esta propriedade está ilustrada na Figura 74. A Figura 83 apresenta os axiomas com as restrições relativas a remetentes e destinatários de mensagens `agent-msg`.

```

(implies agent-msg
  (or (and (some has-msg-sender
    (or agent-role-class-path
      agent-role-instance-path
      main-organization-class-path
      main-organization-instance-path
      active-environment-class-path
      a-environment-instance-path))
    (all has-msg-receiver
      (or agent-role-class-path
        agent-role-instance-path
        main-organization-class-path
        main-organization-instance-path
        environment-class-path
        environment-instance-path))))))

```

Figura 83 Axiomas sobre restrições de remetentes e destinatários de `agent-msg`.

5.4.2.

Consultas sobre propriedades inter-diagramas MAS-ML (QV2)

A instanciação da segunda etapa de F2 consiste na definição de consultas que analisam a base de conhecimento gerada pelos diagramas que compõem a modelagem de um SMA e retornam como resposta as violações às propriedades inter-diagramas previstas pelo metamodelo de MAS-ML. Durante a definição das consultas, foram consideradas propriedades de interdependência entre diagramas estáticos e interdependências entre diagramas estáticos e dinâmicos. A seguir descreve-se algumas destas interdependências.

5.4.2.1.

Interdependências entre diagramas estáticos

O metamodelo de MAS-ML especifica algumas propriedades de interdependência entre diagramas estáticos. A Propriedade LM. 12 indica interdependência entre diagramas de organizações e diagramas de classe.

Propriedade LM. 12 *Toda classe de agente ou de organização modelada num diagrama de classes precisa, necessariamente, estar modelada em algum diagrama de organização.*

Observe que a recíproca da Propriedade LM. 12 não é verdadeira. De fato, uma classe de agente ou organização modelada num diagrama de organização não é, necessariamente, especialização (ou generalização) de outra classe de agente ou organização, ou está relacionada a uma classe de objeto. Assim, como o relacionamento previsto entre agentes, ou entre organizações, num diagrama de classes, é o relacionamento *specialization*, e o relacionamento previsto entre agentes e objetos ou entre organizações e objetos é o relacionamento *association*, nem sempre estas classes precisam estar modeladas em diagramas de classe. A Figura 84 ilustra uma consulta que retorna como resposta violações à Propriedade LM. 12.

```
(retrieve (?agent ?clmd)
  (and (?agent agent-class)
    (?clmd class-model)
    (?clmd ?agent has-class)
    (not (?agent (some is-in-static-model
      organization-model))))))
```

Figura 84. Consulta sobre interdependência entre diagramas de classe e de organizações.

Durante a realização do estudo de caso (Capítulo 5, seção 5.4.2), observou-se que o aparato dedutivo proporcionado pelo sistema RACER não era capaz de responder à consulta descrita na Figura 84 para bases de conhecimento de grande porte. Assim, decidiu-se pela composição de consultas, ilustradas na Figura 85 cujas respostas são facilmente manipuladas com o uso de um programa simples, incluído na ferramenta de suporte à utilização do Observed-MAS instanciado para MAS-ML (seção 5.5), e que retorna o resultado esperado.

```
(retrieve (?agent ?clmd)
  (and (?agent agent-class)
    (?clmd class-model)
    (?orgmd organization-model)
    (?clmd ?agent has-class)
    (not (?orgmd ?agent has-class))))

(retrieve (?orgmd) (?orgmd organization-model))
```

Figura 85. Composição de consultas para contornar limitações computacionais.

A Propriedade LM. 13 descreve uma interdependência entre diagramas de papel e diagramas de organização.

Propriedade LM. 13 *Toda classe de papel de agente ou de papel de objeto definida num diagrama de papel precisa estar definida em algum diagrama de organização.*

Observe que a recíproca da Propriedade LM. 13 não é verdadeira. De fato, classes de papel de agente ou de papel de objeto modeladas num diagrama de organização não precisam estar, necessariamente, relacionadas entre si. Diagramas de papel mostram apenas relacionamentos entre classes de papéis e entre classes de papéis e classes de objetos. Assim, é possível que um papel modelado num diagrama de organização não esteja relacionado a nenhum outro papel e, portanto, não há motivo para modelá-lo num diagrama de papel.

A Figura 86 ilustra uma consulta que retorna como resposta violações à Propriedade LM. 13.

```
(retrieve (?agrl ?rlmd)
  (and (?agrl agent-role-class)
    (?rlmd role-model)
    (?rlmd ?agrl has-class)
    (not (?agrl (some is-in-static-model
      organization-model)))))
```

Figura 86. Consulta sobre interdependência entre diagramas de papéis e de organizações.

O mesmo problema de limitações computacionais, para bases de conhecimento de grande porte, observado para a consulta sobre interdependência entre diagramas de classe e de organizações, também foi

observado para esta consulta, pois ambas seguem o mesmo padrão de descrição. Desta forma foi adotada a mesma solução para contornar tais limitações (Figura 87).

```
(retrieve (?agrl ?rlmd)
  (and (?agrl agent-role-class)
    (?rlmd role-model)
    (?orgmd organization-model)
    (?rlmd ?agrl has-class)
    (not (?orgmd ?agrl has-class))))

(retrieve (?orgmd) (?orgmd organization-model))
```

Figura 87. Composição de consultas (II) para contornar limitações computacionais.

Um exemplo de interdependência entre diagramas de classe e diagramas de papéis é descrito pela Propriedade LM. 14.

Propriedade LM. 14 *Toda classe de objeto modelada num diagrama de papel precisa estar modelada em algum diagrama de classe.*

A consulta ilustrada na Figura 88 retorna como resposta violações à Propriedade LM. 14.

```
(retrieve (?object ?rlmd)
  (and (?object object-class)
    (?rlmd role-model)
    (?rlmd ?object has-class)
    (not (?object (some is-in-static-model
      class-model)))))
```

Figura 88. Consulta sobre interdependência entre diagramas de classe e de papéis.

Analogamente ao ocorrido nas consultas anteriores, também houve a necessidade de composição de consultas para contornar limitações computacionais quando a modelagem gera bases de conhecimento de grande porte. Desta forma foi adotada a mesma solução para contornar tais limitações (Figura 89).

```
(retrieve (?obj ?rlmd)
  (and (?obj object-class)
    (?rlmd role-model)
    (?clmd class-model)
    (?rlmd ?obj has-class)
    (not (?clmd ?obj has-class))))

(retrieve (?obj) (?obj class-model))
```

Figura 89. Composição de consultas (III) para contornar limitações computacionais.

A seguir apresentam-se algumas consultas relacionadas aos processos dinâmicos primitivos e complexos, a fim de exemplificar o conjunto de

propriedades a serem analisadas para descobrir violações inter-diagramas dinâmicos e estáticos.

5.4.2.2.

Interdependência entre diagramas dinâmicos e estáticos

As interdependências entre diagramas dinâmicos e estáticos são, em geral, relacionadas ao tipo de mensagem trocada em diagramas de seqüência. Porém existe uma de caráter geral, descrita pela Propriedade LM. 15.

Propriedade LM. 15 *Toda instância de classe descrita num diagrama de seqüência precisa ser instância de uma classe definida em algum diagrama estático da modelagem.*

Em (Straeten et al, 2003), a violação à Propriedade LM. 15, no contexto de diagramas UML de classe e de seqüência, foi chamada de *classless instance*. O metamodelo de MAS-ML especifica que toda instância de classe possui um identificador, definido em Ont1 (e Ont2) pelo conceito `path` (Figura 51). Um `path` possui referências à instância e à classe que originou a instância que ele identifica, no contexto do SMA. Assim, é possível, a partir do `path`, recuperar informações sobre a classe que originou a instância que ele identifica. A Figura 90 ilustra consulta que retorna como resposta violações à Propriedade LM. 15.

```
(retrieve (?ipath ?seqmd ?head)
  (and (?ipath instance-path)
    (?seqmd sequence-model)
    (?seqmd ?ipath has-path)
    (?ipath ?head has-head)
    (?head nil is-instanceOf)))
```

Figura 90. Consulta sobre interdependência entre diagramas dinâmicos e estáticos (*classless*).

Nos caso das propriedades dos processos dinâmicos primitivos (criação e destruição de entidades), os exemplos de consultas que violam propriedades de interdependência são casos particulares de *classless*. De fato, um diagrama que descreve a criação de uma entidade deve considerar que a classe da qual a entidade criada é uma instância deve ter sido definida em algum diagrama estático. Além desta violação, diagramas dinâmicos que descrevem a criação de agentes e sub-organizações possuem outro tipo de interdependência com diagramas de organizações, pois ambos precisam ser associados a algum papel de agente no momento da criação (Propriedade LM. 16).

Propriedade LM. 16 *Na instancição de uma classe de agente (ou sub-organização) é necessário que haja um relacionamento `play` entre a classe a ser instanciada e a classe de papel de agente que é referenciada no momento da instancição. Este relacionamento deve estar descrito em um diagrama de organização.*

A Figura 91 ilustra uma consulta que retorna como resposta violações à Propriedade LM. 16.

```
(retrieve (?seqmd ?agent-ipath ?create ?role)
  (and (?seqmd sequence-model)
    (?agent-ipath agent-instance-path)
    (?seqmd ?agent-ipath has-path)
    (?create create-msg)
    (?seqmd ?create has-msg)
    (?agent-ipath ?create is-msg-receiver)
    (?agent-ipath ?role-ipath has-tail)
    (?role agent-role)
    (?role-ipath ?role has-head)
    (?role nil is-instanceOf)))
```

Figura 91. Caso particular de *classless* durante a criação de agentes.

Nos processos dinâmicos complexos, as propriedades de interdependência entre diagramas dinâmicos e estáticos são referentes ao uso dos estereótipos relacionados à dinâmica de papéis. Novamente, as interdependências entre diagramas dinâmicos, que descrevem estes processos, e diagramas estáticos, são casos particulares de *classless*. No Apêndice D são apresentadas as consultas QV2.

5.4.3.

Consultas sobre sugestões de boas práticas de modelagem usando MAS-ML (QD2)

A instancição da terceira etapa de F2 consiste na definição de consultas (QD2) que analisam a base de conhecimento gerada a partir de diagramas MAS-ML e retornam como resposta sugestões de boas práticas de modelagem usando MAS-ML.

Como dito na seção 5.3.3, a interação entre agentes ocorre através de protocolos. O metamodelo de MAS-ML especifica que protocolos são definidos no contexto de papéis de agentes, logo, ao modelar a interação entre agentes e/ou sub-organizações usando MAS-ML, deve-se observar que ela se dá através dos papéis que os mesmos desempenham. Além disso, os papéis são definidos

no contexto de diagramas de organizações, porém os relacionamentos entre papéis são modelados em diagramas de papéis. Esta especificidade da linguagem sugere que papéis de agentes, descritos em diagramas de organizações e não descritos em diagramas de papéis, sejam desempenhados por agentes que não interagem com outros agentes ou sub-organizações no sistema. Agentes que não interagem são, em geral, não desejáveis num SMA.

Assim, pode-se definir uma propriedade de boas práticas de modelagem considerando-se os diagramas MAS-ML de organização e de papel, especificada na Propriedade BP 3.

Propriedade BP 3. *Classes de papel de agente modeladas em diagramas de organização devem, em geral, estar modeladas em diagramas de papéis.*

Observe que a violação a esta propriedade não caracteriza um erro de modelagem, mas apenas o fato de que agentes ou sub-organizações que desempenham o papel em questão não podem interagir com outros agentes ou sub-organizações. A Figura 92 ilustra uma consulta que retorna como resposta os papéis de agente que possuem esta característica.

```
(retrieve (?agrl ?orgmd)
  (and (?agrl agent-role-class)
    (?orgmd organization-model)
    (?orgmd ?agrl has-class)
    (not (?agrl (some is-in-static-model role-model)))))
```

Figura 92. Consulta que retorna como resposta sugestão de modelagem (I).

Observe que a Figura 92 segue o padrão das consultas sobre interdependência entre diagramas estáticos, cujas respostas estão sujeitas ao tamanho da base de conhecimento analisada. A mesma solução usada naquela ocasião também foi usada neste caso, para o par de consultas ilustrado na Figura 93.

```
(retrieve (?agrl ?orgmd)
  (and (?agrl agent-role-class)
    (?rlmd role-model)
    (?orgmd organization-model)
    (?orgmd ?agrl has-class)
    (not (?rolemd ?agrl has-class))))

(retrieve (?rolemd) (?rolemd role-model))
```

Figura 93. Combinação de consultas que retornam como resposta sugestão de modelagem.

Ainda considerando a questão da interação entre agentes, ela se dá através de protocolos definidos em papéis de agentes. Papéis de agentes relacionados em diagramas de papéis possuem protocolos comuns, e estes

protocolos devem, em geral, ser modelados em diagramas de seqüência. Assim, pode-se definir uma propriedade de boas práticas de modelagem considerando-se os diagramas MAS-ML de papel e de seqüência, especificada na Propriedade BP 4.

Propriedade BP 4 *Protocolos comuns a classes relacionadas de papéis de agente modeladas em um diagrama de papel devem, em geral, ser modelados em um diagrama de seqüência.*

```
(retrieve (?prtcl-not-modeled)
  (and (?rel (or association dependency aggregation control))
    (?rlmd role-model)
    (?rlmd ?rel has-relationship)
    (?role-class1 agent-role-class)
    (?role-class2 agent-role-class)
    (?rel ?role-class1 has-end1)
    (?rel ?role-class2 has-end2)
    (?prtcl-not-modeled protocol)
    (?role-class1 ?prtcl-not-modeled has-protocol)
    (?role-class2 ?prtcl-not-modeled has-protocol)
    (?struct sequence)
    (?prtcl-not-modeled ?struct has-structure)
    (?seqmd sequence-model)
    (not (?seqmd ?struct has-structure))))
```

Figura 94. Consulta que retorna sugestão de modelagem (II).

5.5.

Especificação de uma ferramenta para automatização do Observed-MAS aplicado a MAS-ML

Para auxiliar na prova de conceito do método Observed-MAS foi implementada uma ferramenta que automatiza parte do processo de estruturação e análise de diagramas MAS-ML. A ferramenta foi chamada MAS-DCheck (*Multi-Agent Systems Design Checker*).

A MAS-DCheck foi especificada para receber como dados de entrada o mapeamento dos diagramas MAS-ML para instâncias das ontologias Ont1 e Ont2 que descrevem os diagramas referentes às fases F1 e F2 do método e fornecer como saída os resultados das consultas solicitadas pelo usuário em forma de arquivo texto.

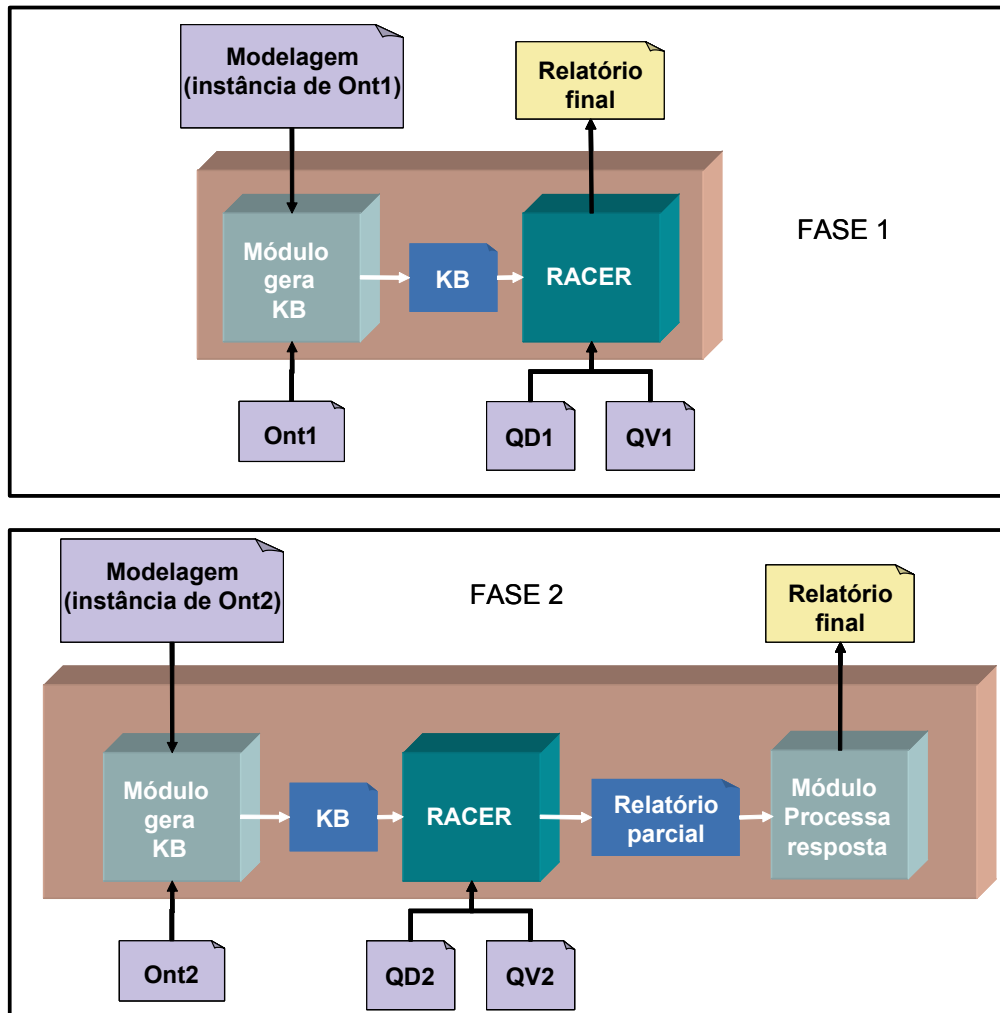


Figura 95. Descrição da arquitetura da MAS-DCheck.

A arquitetura da MAS-DCheck é descrita na Figura 95, onde pode ser observado que as ontologias e os conjuntos de consultas definidos pelo Observed-MAS são módulos utilizados pela ferramenta, assim como o sistema de representação de conhecimento.

Internamente, para cada fase do método, a ferramenta processa o arquivo de entrada, combinando-o com a ontologia correspondente para gerar a base de conhecimento. Gerada a base, e escolhido o conjunto de consultas a ser executado, é solicitada a execução do sistema RACER tendo como dados de entrada a base de conhecimento e o conjunto de consultas. Até este ponto a ferramenta se comporta igualmente para ambas as fases.

Na fase F1, a partir da resposta retornada pelo sistema RACER, a MAS-DCheck simplesmente retorna para o usuário o relatório fornecido pelo RACER. Na fase F2, a MAS-DCheck processa a resposta retornada pelo RACER, a fim

de combinar as respostas das consultas duplas efetuadas para contornar as limitações computacionais encontradas durante a análise inter-diagramas.

Para implementação da MAS-DCheck foi usada a linguagem PERL⁵, pois é uma linguagem desenvolvida para programação destinada à manipulação e edição de textos. Como os arquivos manipulados pelo sistema RACER são, essencialmente, arquivos de texto, esta pareceu uma escolha adequada.

Ainda não foi desenvolvida uma interface gráfica para a MAS-DCheck e, portanto, o usuário indica, em linha de comando, o arquivo que contém a instância da ontologia e o conjunto de consultas que deseja executar, sejam elas referentes à fase F1 ou F2. MAS-DCheck é programada para escolher a ontologia adequada para estruturar a base de conhecimento gerada a partir dos dados de entrada e da análise solicitada. Desta forma, a ferramenta escolhe a ontologia Ont1 se o usuário solicitar uma análise intra-diagrama (QV1) ou sugestões de modelagem intra-diagrama (QD1); ou escolhe a ontologia Ont2 se o usuário solicitar uma análise inter-diagramas (QV2) ou sugestões de modelagem inter-diagramas (QD2).

A saída é apresentada ao usuário em arquivos do tipo texto e seu estilo de edição depende da necessidade ou não de composição de resultados de consultas. Se não há composição de resultados, a saída é o arquivo texto que o RACER define como saída. Havendo composição de respostas de consultas, a saída ainda é um arquivo texto, porém com formato diferente do apresentado pelo RACER.

5.6.

Discussão

Neste capítulo foi descrita uma prova de conceito para o método Observed-MAS através da sua aplicação para a linguagem de modelagem MAS-ML. Para tanto, foram construídas as ontologias Ont1 e Ont2 que apoiam a estruturação das bases de conhecimento geradas a partir do mapeamento de diagramas MAS-ML para instâncias das ontologias. Assim, foram definidas consultas cujas respostas provêm suporte à análise destes diagramas ou guiam o desenvolvedor na utilização de boas práticas de modelagem, usando a linguagem MAS-ML. Além disso, para automatizar a execução do método instanciado para MAS-ML, foi desenvolvida uma ferramenta que usa o aparato

⁵ www.perl.com

dedutivo disponível pelo sistema RACER para, dado entradas que são instâncias das ontologias, retornar como saída as respostas às consultas solicitadas pelo usuário. Esta ferramenta foi projetada para servir de back-end para uma ferramenta gráfica de edição de modelos MAS-ML (Maria et al, 2005).

Versões anteriores a definição final do método, consideraram uma abordagem de fase única (Brandão et al, 2005a) e posteriormente uma abordagem trifásica (Brandão et al, 2005b) para estruturação e análise dos diagramas MAS-ML.

A abordagem de fase única limitou o alcance do suporte que o método instanciado poderia prover durante as atividades de modelagem de SMAs. Esta limitação era decorrente da estruturação feita apenas por uma ontologia mais restritiva, descrita no Observed-MAS como Ont2. Assim, a existência de um diagrama mal estruturado geraria uma base de conhecimento inconsistente, impedindo o uso de consultas para descobrir qual propriedade estava sendo violada e onde estava sendo violada.

Por outro lado, a abordagem trifásica protelava para a terceira fase as sugestões de boas práticas de modelagem, e definia três ontologias, cada uma mais restritiva que a outra. A terceira ontologia incorporava axiomas que descreviam as propriedades inter-diagramas, o que acarretava a introdução de ciclos na ontologia, inviabilizando parte das consultas referentes às boas práticas de modelagem. Além disso, como as consultas referentes às boas práticas de modelagem estavam numa fase onde o desenvolvedor já possuía a informação de que sua modelagem estava de acordo com a especificação do metamodelo da linguagem, seu uso seria, fatalmente, desestimulado. Assim, foram adotadas duas fases de análise na versão final do método e incluídas em cada fase um conjunto de consultas sobre sugestões de boas práticas de modelagem.

Alguns problemas referentes a limitações computacionais foram observados para bases de conhecimento de grande porte. Estes problemas foram contornados usando-se duas estratégias: manter as ontologias Ont1 e Ont2 livres de ciclos, por construção; e combinar consultas simples a fim de obter a mesma resposta que deveria ser retornada por consultas complexas e que não são suportadas pelo aparato dedutivo escolhido, utilizando-se um simples algoritmo de contagem.

Como prova de conceito a aplicação cumpriu seu papel, porém é interessante que o método seja aplicado para outras linguagens de modelagem para SMAs, como AUML (Bauer et al, 2001) e AORML (Wagner, 2003) para confirmar sua viabilidade. A aplicação para AUML é feita parcialmente no

Capítulo 7, onde são analisadas as principais diferenças entre os diagramas comuns a ambas as linguagens.

A ferramenta implementada é parcialmente dependente da linguagem de modelagem adotada, pois o algoritmo que manipula as respostas das consultas combinadas é dependente da forma como são definidas as consultas, as quais foram definidas para diagramas da linguagem MAS-ML. Esta é a única dependência relativa à linguagem. Quanto ao sistema de representação de conhecimento usado pela MAS-DCheck, ele também pode ser trocado, desde que tanto as ontologias quanto as consultas estejam representadas na linguagem implementada pelo sistema.

As principais contribuições apresentadas neste capítulo foram: a definição de ontologias que descrevem parcialmente o metamodelo de MAS-ML, proporcionando a análise de seus modelos através da aplicação do método Observed-MAS; a definição de consultas cujo interesse é analisar a boa formação dos diagramas MAS-ML; a definição de consultas que identificam boas práticas de modelagem no domínio de SMAs usando MAS-ML e a especificação e implementação de uma ferramenta que provê suporte para a aplicação do método Observed-MAS para MAS-ML.