

Aplicação do método Observed-MAS para AUML

Neste capítulo é apresentada uma aplicação do método para a linguagem AUML (*Agent UML* (Odell et al, 2000)), a fim de testar sua aplicabilidade para outra linguagem de modelagem e de analisar a dependência do método ao arcabouço conceitual TAO (Silva et al, 2003).

7.1.

Preâmbulo

Como apresentado no Capítulo 4, parte da primeira ontologia (Ont1) proposta pelo método Observed-MAS foi definida com base no arcabouço conceitual TAO. No Capítulo 5 o método foi aplicado para a linguagem de modelagem MAS-ML cujo metamodelo também foi definido com base no TAO. Sendo assim, na primeira aplicação do método não foi possível avaliar os problemas que poderiam surgir com a aplicação do método para uma linguagem de modelagem não baseada no TAO.

A fim de avaliar o quão dependente do TAO é o método Observed-MAS, instanciou-se o mesmo para a linguagem AUML. Diferentemente de MAS-ML, o metamodelo de AUML não foi definido com base nas entidades, propriedades e relacionamentos definidos no TAO. Além da análise da dependência com o TAO, o uso da linguagem de modelagem AUML também auxiliou no teste de viabilidade da instanciação de todas as fases do método, demonstrando como as ontologias e as consultas poderiam ser definidas para uma outra linguagem de modelagem.

Inicialmente, faz-se uma descrição geral da linguagem, das entidades e relacionamentos que ela define, assim como dos seus diagramas. A partir desta descrição passa-se à aplicação do método propriamente dita, com a construção da ontologia Ont1, a definição das propriedades internas aos diagramas e, conseqüentemente, das consultas que violam tais propriedades, bem como a definição de propriedades de boas práticas de modelagem para a construção dos diagramas.

Em paralelo à instanciação da fase F1 do método realizou-se um pequeno estudo de caso, relacionado ao domínio de mercados virtuais, o mesmo realizado para a linguagem MAS-ML e apresentado no Capítulo 6, seção 6.2. Terminada a instanciação da fase F1, passou-se à fase F2 do método, com a definição da ontologia Ont2 e das consultas que violam as propriedades inter-diagramas e que sugerem boas práticas de modelagem considerando o conjunto de diagramas que descrevem o modelo de um SMA.

7.2.

AUML

A linguagem de modelagem AUML estende UML a fim de descrever os aspectos relacionados às especificidades de agentes de software. Esta linguagem tem sido desenvolvida por vários pesquisadores, de forma incremental, inicialmente estendendo os diagramas dinâmicos de UML a fim de descrever protocolos de interação entre agentes (Odell et al, 2000; Bauer et al, 2002; Huget, 2002b), e posteriormente extensões do diagrama de classe (Bauer, 2002; Huget, 2002a). Além da extensão dos diagramas previstos em UML, novas entidades foram introduzidas pela linguagem, como agentes, papéis (Odell et al, 2000; Bauer, 2002), organizações (grupos) e ambiente (Parunak & Odell, 2002). Apesar de introduzidas na linguagem, as entidades papel, organização e ambiente não tiveram suas propriedades internas especificadas.

Como o critério usado para estender UML foi o uso de estereótipos, não foram definidos novos relacionamentos ou diagramas, sendo apenas estendidos os relacionamentos e diagramas previstos em UML para modelar as novas entidades. A seguir serão descritos os diagramas de classe e de sequência definidos na linguagem AUML, pois a aplicação do método considera apenas estes dois diagramas, que são comuns às linguagens AUML e MAS-ML (aplicação feita no Capítulo 5).

7.2.1.

Diagramas de classe AUML

Diagramas de classe foram estendidos para descrever relacionamentos entre papéis; ou entre organizações e papéis; ou relacionamentos entre agentes (Bauer, 2002; Odell et al; 2003). A sintaxe dos diagramas de classe ainda não

está totalmente estabelecida, pois Bauer (2002) e Huget (2002a) trabalharam neste diagrama e apresentam pequenas discordâncias quanto à descrição da classe de agente e dos relacionamentos possíveis entre agentes e entre agentes e objetos. Nesta aplicação será considerada a extensão proposta por Bauer e colaboradores (Bauer, 2002; Odell et al, 2003), pois Huget (2002a) não faz menção às organizações nem aos ambientes.

Um exemplo de diagrama de classe que descreve o relacionamento entre papéis, agentes e organizações é ilustrado na Figura 126. Para indicar quais papéis estão agrupados numa organização ou quais instâncias de agentes desempenham determinados papéis são usadas raias (*swim lanes*) horizontais e verticais, sendo que as raias verticais indicam o relacionamento de agregação entre papéis e organizações e as raias horizontais indicam a instanciação de agentes desempenhando papéis.

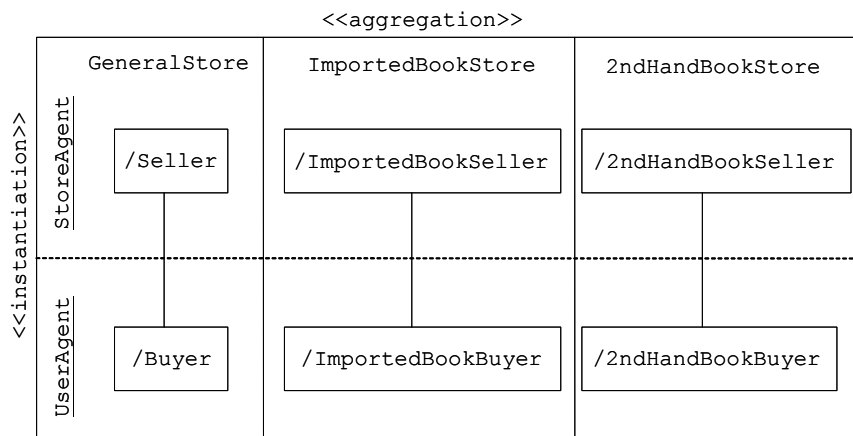


Figura 126. Diagrama de classe com representando organizações, papéis e agentes usando swim lanes.

O diagrama de classe também descreve relacionamentos entre grupos e papéis. Neste caso o relacionamento definido é agregação. A Figura 127 ilustra os relacionamentos de agregação definidos entre a organização *GeneralStore* e os papéis *Buyer* e *Seller*, e o relacionamento de associação entre os papéis *Buyer* e *Seller*, também descritos no diagrama da Figura 126.

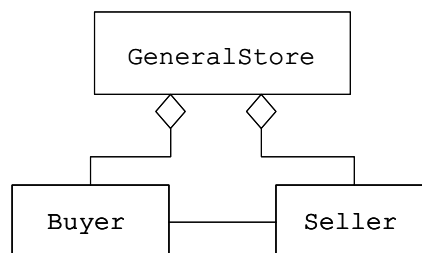


Figura 127. Exemplo de diagrama de classe descrevendo relacionamento entre organização e papéis.

O relacionamento entre agentes também é descrito em diagramas de classe. Neste caso, o relacionamento usado é associação. Classes de agentes podem ser representadas de diferentes formas. Na Figura 128, o primeiro diagrama indica uma classe de agente, sem indicação de papéis que o mesmo desempenha; o segundo indica uma classe de agente cujas instâncias podem desempenhar o conjunto de papéis definidos logo em seguida (*Buyer*, *ImportedBookBuyer*,...), e o terceiro indica uma instância (*Mary*) da classe de agente *UserAgent*, que desempenha os papéis definidos logo após o nome da instância. De acordo com a especificação da linguagem, uma instância de agente não é obrigada a desempenhar papéis (Bauer, 2002).

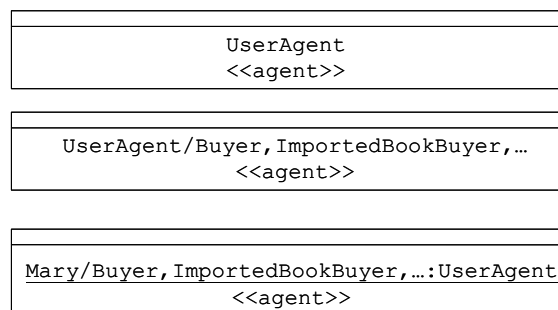


Figura 128. Diferentes formas de descrição de uma classe/instância de agente usando diagrama de classe.

Em diagramas de classe que modelam agentes também são descritas as propriedades internas dos agentes. Neste caso, um agente é considerado como composto de cabeça e corpo. A cabeça refere-se à parte de comunicação do agente e, por conseguinte, é responsável pelos protocolos de interação do agente, os quais, por sua vez, lidam com as metas e crenças do agente (estado mental). Protocolos de interação são compostos por atos de comunicação (*communicative acts* - CA), que englobam o tipo de mensagem trocada, incluindo informações sobre conteúdo, remetente e destinatário. Atos de comunicação também são representados por classes. O corpo do agente refere-se às ações que o agente pode executar.

7.2.2.

Diagramas de seqüência AUML

As primeiras investigações envolvendo extensão de UML para agentes foram relacionadas à adaptação dos diagramas de seqüência para descrição de protocolos definidos no contexto dos papéis e interações entre agentes desempenhando papéis (Odell et al, 2000). Protocolos são descritos através de

pacotes compostos por seqüências de interações. Além disso, protocolos também são identificados por moldes (*templates*) que especificam como devem ser preenchidas as seqüências de interações definidas num pacote. A Figura 129 ilustra um exemplo de protocolo cujo molde encontra-se no canto superior direito do pacote.

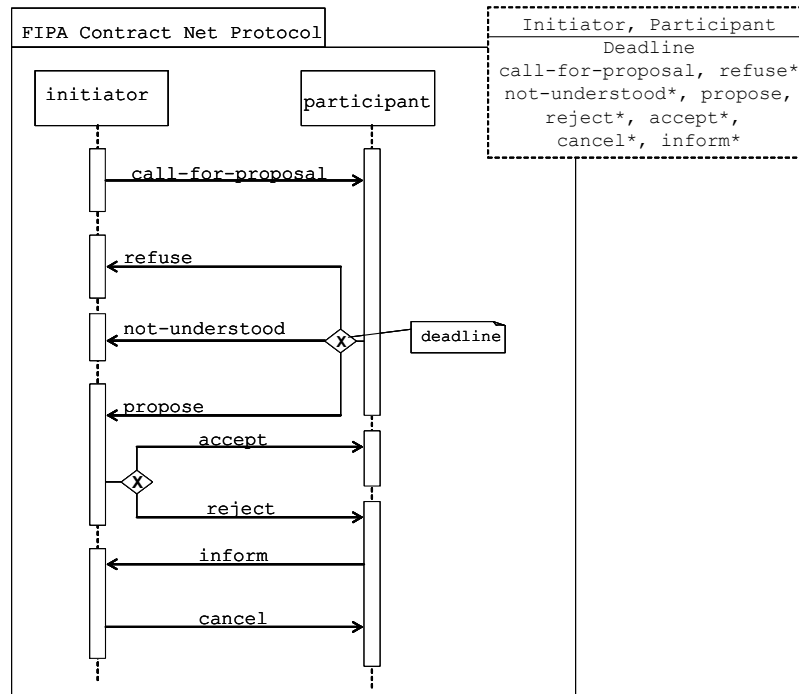


Figura 129. Protocolo de interação descrito usando diagrama de seqüência, pacote e molde.

A partir das informações relacionadas à especificação dos diagramas de classe e de seqüência, passou-se à aplicação do Observed-MAS, com a definição da ontologia Ont1, das consultas QV1 e QD1, e, posteriormente, da ontologia Ont2 e das consultas QV2 e QD2.

7.3.

Fase F1: estruturação e análise das propriedades intra-diagramas AUML

A aplicação do método Observed-MAS para a linguagem AUML teve início com a instanciação da primeira etapa da fase F1. Esta etapa consiste da construção da ontologia Ont1, a partir da integração da primeira ontologia de finida no Capítulo 4, com os termos e relacionamentos que compõem a linguagem AUML. Logo em seguida, são definidos os conjuntos de consultas QV1 e QD1, sobre violações a propriedades intra-diagramas AUML e sobre

sugestões de boas práticas de modelagem destes diagramas, de acordo com as características de SMAs.

7.3.1.

A ontologia Ont1

Alguns dos conceitos relacionados ao domínio de SMAs e definidos na ontologia Ont1 durante a definição do método (Capítulo 4) não são especificados pela linguagem AUML, e, portanto, não são instanciados durante o mapeamento dos modelos AUML para instâncias da ontologia Ont1. A Figura 130 ilustra os conceitos de Ont1 relacionados ao domínio de SMAs, sendo que os conceitos hachurados são os não especificados pela linguagem AUML.

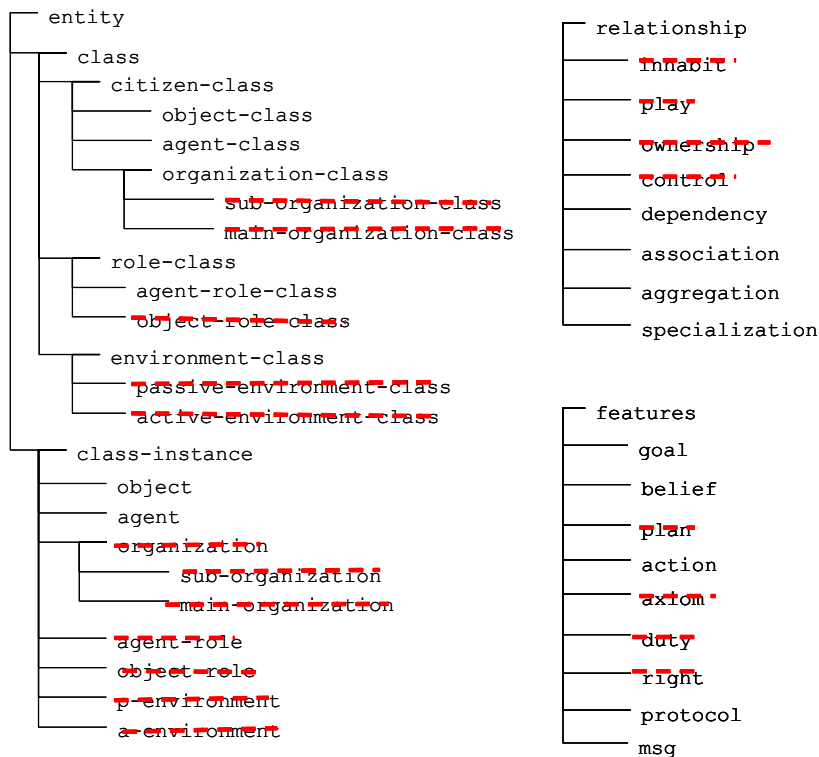


Figura 130. Conceitos relacionados ao domínio de SMA, alguns não especificados em AUML.

Além dos conceitos recepcionados pela linguagem AUML, e indicados sem hachura na Figura 130, foram definidos os conceitos indicados na Figura 131, para descrever todos diagramas dinâmicos previstos em AUML e as entidades que participam destes diagramas.

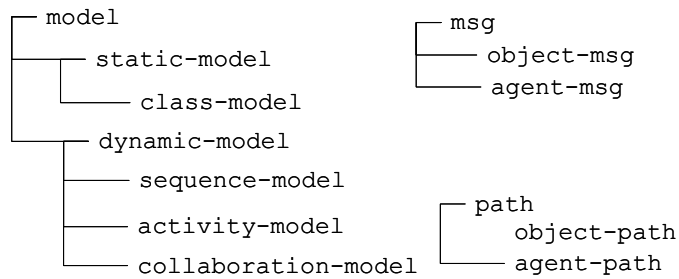


Figura 131. Conceitos relacionados à linguagem AUML.

O código da ontologia Ont1, definida durante a aplicação do método Observed-MAS para AUML encontra-se no Apêndice H.

7.3.2.

Consultas sobre propriedades intra-diagramas AUML (QV1)

A instanciação da segunda etapa da fase F1 consistiu da definição de consultas que devolvem como resposta violações às propriedades internas dos diagramas AUML de classe e de seqüência. Estas propriedades se referem à boa-formação dos diagramas AUML e, conseqüentemente, à consistência das bases de conhecimento geradas a partir do mapeamento destes diagramas para instâncias da ontologia Ont1.

A Propriedade LM. 17 define quais entidades podem participar do relacionamento *dependency*.

Propriedade LM. 17 *O relacionamento `dependency` é permitido apenas entre objetos.*

A consulta ilustrada na Figura 132 devolve como resposta relacionamentos do tipo *dependency* que estão mal estruturados. Observe que, como AUML só define um tipo de diagrama estático (diagrama de classe), não há necessidade de indicar na consulta o diagrama ao qual o relacionamento pertence, bastando indicar o relacionamento e seus extremos.

```

(retrieve (?end1 ?end2 ?dep)
  (and (?dep dependency)
    (?end1 ?dep is-end1)
    (?end2 ?dep is-end2)
    (or (and (?end1 object-class)
              (not (?end2 object-class)))
        (not (?end1 object-class)))))
  
```

Figura 132. Consulta sobre violações a propriedades intra-diagrama de classe AUML (I).

De acordo com a definição de diagramas de classe AUML, organizações são entidades que agregam papéis (Odell et al, 2003), como especificado na Propriedade LM. 18.

Propriedade LM. 18 *Organizações são entidades definidas a partir da agregação de papéis de agentes.*

A consulta ilustrada na Figura 133 devolve como resposta indivíduos pertencentes ao conceito `organization-class` que não possuem relacionamento de agregação com nenhum indivíduo pertencente ao conceito `agent-role-class`.

```
(retrieve (?org-without-roles)
  (and (?org-without-roles organization-class)
    (not (?org-without-roles
      (some is-end2 (and aggregation
        (some has-end1 agent-role-class)))))))
```

Figura 133. Consulta sobre violações a propriedade intra-diagrama de classe AUML (II).

Diagramas de sequência descrevem protocolos de interação. A Propriedade LM. 19 especifica que protocolos devem possuir estrutura determinada por um molde (*template*), e descrita como uma sequência de mensagens num diagrama AUML de sequência.

Propriedade LM. 19 *Protocolos ficam completamente descritos através de seus moldes, então, protocolos devem possuir moldes.*

A consulta ilustrada na Figura 134 devolve como resposta protocolos que não possuem moldes. Outras consultas relativas à violação de propriedades intra-diagramas de AUML são encontradas no Apêndice H.

```
(retrieve (?prctl-without-struct)
  (and (?prctl-without-struct protocol)
    (?prctl-without-struct nil has-structure)))
```

Figura 134. Consulta sobre violações a propriedade intra-diagrama de classe AUML (III).

7.3.3.

Consultas sobre sugestões de boas práticas de modelagem (QD1)

A instanciação da terceira etapa da fase F1 consiste da definição de consultas que reflitam sugestões de boas práticas de modelagem de diagramas AUML, baseadas nas características de SMAs. Na literatura relacionada à utilização de AUML para modelar SMAs, não está expressamente especificada a propriedade que diz que agentes devem desempenhar papéis. Porém, em SMAs é comum o uso de papéis de agentes para modelar as interações entre agentes que fazem

parte de um grupo/organização. Assim, pode-se definir uma propriedade de boas práticas de modelagem, como a especificada pela Propriedade BP 5.

Propriedade BP 5. *Agentes devem, em geral, estar associados a pelo menos um papel de agente.*

Observe que uma violação a esta propriedade não caracteriza um erro de modelagem pois, como descrito na Figura 128, um agente não precisa, necessariamente, estar associado a algum papel. A consulta ilustrada na Figura 135 devolve como resposta classes de agentes que não estão associadas a nenhum papel.

```
(retrieve (?agentwithoutrole)
  (and (?agentwithoutrole agent-class)
    (?agentwithoutrole NIL plays-role)))
```

Figura 135. Consulta sobre sugestões de boas práticas de modelagem.

Outras consultas sobre sugestões de boas práticas de modelagem são ilustradas no Apêndice H.

7.4.

Fase F2: estruturação e análise das propriedades de interdependência entre diagramas AUMML

A instanciação da fase F2 para AUMML teve início com a definição da ontologia Ont2, a partir da extensão da ontologia Ont1 com a introdução de axiomas que descrevem as propriedades internas das entidades, relacionamentos e diagramas definidos em Ont1. Após a especificação de Ont1, foram definidos os conjuntos de consultas QV2 e QD2, relativos a violações às propriedades de interdependência entre os diagramas AUMML de classe e de sequência, e às sugestões de boas práticas de modelagem quando considerados os diagramas de classe e de sequência numa modelagem AUMML de um SMA.

7.4.1.

A ontologia Ont2

Durante a definição de Ont2 para a aplicação do método para AUMML, com a extensão de Ont1, observou-se que as propriedades internas de algumas entidades e alguns relacionamentos descritos na primeira ontologia do TAO não foram recepcionados pela linguagem AUMML, pois as entidades e os

relacionamentos não estão especificados pela linguagem (Figura 130). Por outro lado, a descrição das propriedades internas de algumas entidades e relacionamentos, referentes ao domínio de SMAs e recepcionados pela linguagem AUML, precisou ser redefinida. Isto ocorreu na descrição das propriedades internas dos agentes (*agent-class*), das organizações (*organization-class*), dos papéis (*agent-role-class*) e dos ambientes (*environment-class*), assim como dos relacionamentos de especialização (*specialization*), agregação (*aggregation*), dependência (*dependency*) e associação (*association*). A necessidade de redefinir as propriedades internas destas entidades demonstrou, experimentalmente, que a Ont1, em sua parte referente ao domínio de SMAs, era dependente do TAO.

Como as entidades *organization-class*, *agent-role-class*, e *environment-class* não tiveram suas propriedades internas especificadas em AUML, a descrição destas propriedades foi simplesmente excluída da ontologia Ont2. Relativamente à entidade *agent-class*, só fica explícito pela descrição dos diagramas de classe que descrevem agentes (Bauer, 2002), que agentes devem possuir protocolos de interação, mas não necessariamente ações, metas e crenças. Os axiomas que descrevem tal propriedade são ilustrados na Figura 136.

```
(implies agent-class (some has-protocol protocol))
(implies (some has-action action) agent-class)
(implies (some has-goal goal) agent-class)
(implies (some has-belief belief) agent-class)
```

Figura 136. Axiomas sobre propriedades internas dos agentes.

Sobre os relacionamentos recepcionados pela linguagem e que tiveram suas descrições redefinidas, as mesmas são apresentadas na Figura 137. Em negrito estão os axiomas que refletem as consultas apresentadas nas figuras Figura 132 e Figura 133, as quais violam as propriedades Propriedade LM. 17 e Propriedade LM. 18, respectivamente.

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

;; association
(implies association
  (and (some has-end1 citizen-class)
       (some has-end2 citizen-class)))

;; dependency
```

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

;; aggregation
(implies aggregation
  (or
    (and (some has-end1 object-class)
          (all has-end2 object-class))
    (and (some has-end1 agent-class)
          (all has-end2 agent-class))
    (and (some has-end1 (or agent-role-class
                             environment-class))
          (all has-end2 organization-class))))
```

Figura 137. Axiomas sobre estrutura dos relacionamentos em AUML.

A Propriedade LM. 19 especifica que protocolos são descritos através de um molde, que define uma seqüência de mensagens. Os axiomas que refletem esta propriedade são apresentados na Figura 138. O código completo da ontologia Ont2 é apresentado no Apêndice H.

```
(implies protocol (all has-structure sequence))
(implies protocol (all has-msg agent-msg))
```

Figura 138. Axiomas sobre propriedades internas de protocolos.

7.4.2.

Consultas sobre propriedades inter-diagramas AUML (QV2)

A instanciação da segunda etapa da fase F2 para AUML consistiu da definição de consultas que devolvem como resposta violações às propriedades de interdependência dos diagramas AUML de classe e de seqüência, a partir da análise da base de conhecimento gerada pela instância de Ont2 que representa estes diagramas.

A partir da observação cuidadosa dos diagramas AUML de classe e de seqüência, foram definidas duas propriedades de interdependência entre eles, descritas a seguir.

Propriedade LM. 20 *Identificadores de agente modelados em diagramas de seqüência (que descrevem protocolos) devem ter uma entidade do tipo agent-class ou agent-role-class ou agent modeladas em algum diagrama de classe da modelagem.*

A consulta ilustrada na Figura 139 devolve como resposta indivíduos que participam de diagramas de seqüência e violam a Propriedade LM. 20.

```
(retrieve (?participant ?seq-md)
  (and (?participant agent-path)
```

```
(?seq-md sequence-model)
(?seq-md ?participant has-path)
(not (?participant
      (some is-in-static-model class-model))))))
```

Figura 139. Consulta sobre propriedade de interdependência entre diagramas AUMML (I).

Propriedade LM. 21 *O estado interno do(s) agente(s) que desempenha(m) os papéis descritos num diagrama de seqüência, que modela protocolos, deve(m) estar modelado(s) em algum diagrama de classe da modelagem.*

As consultas ilustradas na Figura 140 devolvem como resposta indivíduos que violam a Propriedade LM. 21.

```
(retrieve (?prtcl ?participant ?seq-md)
  (and (?participant agent-class)
        (?seq-md sequence-model)
        (?prtcl protocol)
        (?struct sequence)
        (?class-md class-model)
        (?seq-md ?participant has-path)
        (?seq-md ?struct has-structure)
        (?prtcl ?struct has-structure)
        (not (and (?participant ?prtcl has-protocol)
                  (?class-md ?participant has-entity))))))

(retrieve (?prtcl ?participant ?seq-md)
  (and (?participant agent)
        (?seq-md sequence-model)
        (?prtcl protocol)
        (?struct sequence)
        (?class-md class-model)
        (?seq-md ?participant has-path)
        (?seq-md ?struct has-structure)
        (?prtcl ?struct has-structure)
        (not (and (?participant ?ag-class is-instanceOf)
                  (?ag-class ?prtcl has-protocol)
                  (or (?class-md ?participant has-entity)
                      (?class-md ?ag-class has-entity))))))

(retrieve (?prtcl ?participant ?seq-md)
  (and (?participant agent-role-class)
        (?seq-md sequence-model)
        (?prtcl protocol)
        (?struct sequence)
        (?class-md class-model)
        (?seq-md ?participant has-path)
        (?seq-md ?struct has-structure)
        (?prtcl ?struct has-structure)
        (not (and (?ag-class ?participant plays-role)
                  (?ag-class ?prtcl has-protocol)
                  (or (?class-md ?participant has-entity)
                      (?class-md ?ag-class has-entity))))))
```

Figura 140. Consulta sobre propriedades de interdependência entre diagramas AUMML (II)

7.4.3.

Consulta sobre sugestões de boas práticas de modelagem (QD2)

A finalização da aplicação do método para AUML consistiu da definição de consultas sobre sugestões de boas práticas de modelagem usando AUML. Como a interação entre agentes acontece através de protocolos, os quais são especificados no estado interno do agente e utilizados em diagramas AUML de seqüência, define-se uma propriedade de boas práticas de modelagem usando AUML como especificado na Propriedade BP 6.

Propriedade BP 6. *Protocolos definidos em classes de agente que estão modeladas em algum diagrama de classe devem, em geral, estar modelados num diagrama de seqüência.*

A consulta ilustrada na Figura 141 devolve como resposta protocolos e classes de agentes que violam a Propriedade BP 6.

```
(retrieve (?prtcl ?ag-cl)
  (and (?prtcl protocolo)
    (?ag-cl agent-class)
    (?ag-cl ?prtcl has-protocol)
    (?struct sequence)
    (?prtcl ?struct has-structure)
    (not (?struct
      (some (inv has-structure) ag-class)))))
```

Figura 141. Consulta sobre boas práticas de modelagem usando AUML.

7.5.

Estudo de caso: Mercados virtuais

O estudo de caso feito para testes das consultas definidas foi o estudo de caso sobre mercados virtuais, já descrito no Capítulo 6, seção 6.2, usado para a aplicação do método para MAS-ML. Na aplicação para AUML, foram construídos diagramas AUML de classe e de seqüência, alguns deles ilustrados nas figuras Figura 126, Figura 127 e Figura 129. Também foram introduzidos erros na base de conhecimento, inicialmente para apresentar violações às propriedades intra-diagramas e, posteriormente para analisar violações a propriedades de interdependência entre diagramas. Para exemplificar as respostas devolvidas pelas consultas, ilustra-se na Figura 142 os erros introduzidos na base de conhecimento, referentes a violações às propriedades intra-diagramas descritas pela Propriedade LM. 17 e pela Propriedade LM. 18.

```
(instance dep-1 dependency)
(related dep-1 user-agent has-end1)
(related dep-1 book has-end2)
...
(instance imported-bookstore organization-class)
(instance imported-book-buyer agent-role-class)
(instance imported-book-seller agent-role-class)

;(instance agg-3 aggregation)
;(related agg-3 imported-book-buyer has-end1)
;(related agg-3 imported-bookstore has-end2)

;(instance agg-4 aggregation)
;(related agg-4 imported-book-seller has-end1)
;(related agg-4 imported-bookstore has-end2)
```

Figura 142. Introdução de erros na base de conhecimento (I).

Na Figura 143 ilustra-se as respostas obtidas pelas consultas descritas nas figuras Figura 132 e Figura 133. O sinal (;) antes da linha de código, na Figura 142, indica que a linha foi comentada e não é analisada pelo sistema.

```
(RETRIEVE (?END1 ?END2 ?DEP) (AND (?DEP DEPENDENCY) (?END1
?DEP IS-END1) (?END2 ?DEP IS-END2) (OR (AND (?END1 OBJECT-
CLASS) (NOT (?END2 OBJECT-CLASS)) (NOT (?END1 OBJECT-
CLASS)))) --> ((?END1 USER-AGENT) (?END2 BOOK) (?DEP DEP-
1)))
...
(RETRIEVE (?ORG-WITHOUT-ROLES) (AND (?ORG-WITHOUT-ROLES
ORGANIZATION-CLASS) (NOT (?ORG-WITHOUT-ROLES (SOME IS-END2
(AND AGGREGATION (SOME HAS-END1 AGENT-ROLE-CLASS)))))) -->
((?ORG-WITHOUT-ROLES IMPORTED-BOOKSTORE))
```

Figura 143. Respostas a consultas sobre propriedades intra-diagramas de classe AUML.

Relativamente às consultas sobre as propriedades de interdependência entre diagramas de classe e de seqüência, foram introduzidos erros na base de conhecimento, referentes aos diagramas apresentados nas figuras Figura 126 e Figura 129. A introdução destes erros tinha como finalidade testar as consultas QV2, e estão ilustrados na Figura 144.

```
(instance class-diag-1 class-model)
(instance class-diag-2 class-model)
(instance class-diag-3 class-model)
...
(instance negotiate-prtcl-struct sequence)
(related seq-diag-1 negotiate-prtcl-struct has-structure)
(related negotiate-prtcl negotiate-prtcl-struct
has-structure)
...
(instance user-agent agent-class)
(instance mary agent)
(related mary user-agent is-instanceOf)
(related user-agent buyer plays-role)
(related user-agent seller plays-role)
(related user-agent negotiate-prtcl has-protocol)
```

```

...
;(related class-diag-1 buyer has-entity)
(related class-diag-1 seller has-entity)
;(related class-diag-2 mary has-entity)
;(related class-diag-3 user-agent has-entity)
...
(instance seq-diag-1 sequence-model)
(instance seq-diag-2 sequence-model)
(instance seq-diag-3 sequence-model)
...
(related seq-diag-1 buyer has-path)
(related seq-diag-1 seller has-path)
(related seq-diag-2 mary has-path)
(related seq-diag-3 user-agent has-path)

```

Figura 144. Introdução de erros na base de conhecimento (II).

Na Figura 145 ilustra-se as respostas obtidas pelas consultas descritas nas figuras Figura 139 e Figura 140.

```

--> (((?PARTICIPANT BUYER) (?SEQ-MD SEQ-DIAG-1)))

--> (((?PRTCL NEGOTIATE-PRTCL) (?PARTICIPANT USER-AGENT)
(?SEQ-MD SEQ-DIAG-3)))

--> (((?PRTCL NEGOTIATE-PRTCL) (?PARTICIPANT MARY) (?SEQ-MD
SEQ-DIAG-2)))

--> (((?PRTCL NEGOTIATE-PRTCL) (?PARTICIPANT BUYER) (?SEQ-MD
SEQ-DIAG-1)) ((?PRTCL NEGOTIATE-PRTCL) (?PARTICIPANT SELLER)
(?SEQ-MD SEQ-DIAG-1)))

```

Figura 145. Respostas a consultas sobre propriedades inter-diagramas AUML.

7.6.

Discussão

Neste capítulo foi apresentada a aplicação do método Observed-MAS para a linguagem AUML, a fim de analisar a dependência do método ao arcabouço conceitual TAO e testar a aplicação do método para outra linguagem de modelagem.

Durante a construção da ontologia Ont1, na fase F1 do método, observou-se que alguns conceitos definidos no Capítulo 4 como pertencentes ao domínio de SMAs, seguindo especificação do TAO, não foram recepcionados pela ontologia Ont1 (Figura 130). Além destes conceitos, outros conceitos como *organization-class*, *agent-role-class* e *environment-class* foram redefinidos, pois não estão especificadas propriedades internas para indivíduos representados por estes conceitos. Estas alterações na ontologia Ont1 impactaram sobremaneira a ontologia Ont2, especialmente na parte composta

pelos axiomas que descrevem a estrutura dos relacionamentos definidos pelo domínio de SMAs.

A partir deste fato, inferiu-se que o TAO descreve o domínio de SMAs com rigor de detalhes que não são compartilhados pela linguagem AUML e, eventualmente, por outras linguagens de modelagem orientadas a agentes. Assim, pode-se entender que a versão atual do Observed-MAS é dependente do TAO, porém esta dependência não invalida o método, pois é facilmente contornável. Para tanto, deve-se utilizar uma versão mais “relaxada” do TAO, definindo como conceitos da ontologia Ont1, na sua parte referente ao domínio de SMAs, apenas os termos e relacionamentos descritos no TAO, e deixando para o metamodelo da linguagem de modelagem a definição das propriedades internas das entidades e da estrutura dos relacionamentos.

Devido ao objetivo inicial desta segunda aplicação do Observed-MAS, de avaliar a dependência do método ao TAO, a definição das consultas QV1, QD1, QV2 e QD2 não foram feitas de forma exaustiva, sendo deixada para trabalhos futuros uma investigação mais cuidadosa sobre propriedades intra-diagramas e inter-diagramas AUML, assim como a definição dos demais diagramas dinâmicos previstos na linguagem AUML.