4
A Linguagem de Modelagem LawML

A LawML é uma linguagem visual que tem o objetivo de oferecer uma estrutura semântica e uma notação para a modelagem de leis de regulação de interações entre agentes dentro de sistemas multi-agentes abertos. A estrutura semântica é baseada na abordagem de regulação proposta em (Paes, 2005a; Paes et al., 2005b) e as instâncias de elementos de LawML são mapeadas para elementos descritos em código no formato XMLaw (Paes, 2005a).

Para descrever a semântica de LawML, serão utilizados alguns elementos do metamodelo da UML (Booch et al., 2006), dando-se uma semântica restrita à modelagem de leis de governança, no lugar da modelagem de objetos e classes.

A linguagem deve ajudar os desenvolvedores de leis a compreender, especificar, construir, comunicar, documentar e implementar as leis de governança de SMA. Para isso, será descrito um conjunto de diagramas, que servem para a modelagem das leis e que podem ser transformados automaticamente em código no formato XMLaw. Esses diagramas serão baseados nos diagramas da UML, restritos aos aspectos específicos da abordagem de leis.

4.1.
O Formalismo da Linguagem

Este trabalho não tem por objetivo estender a UML para incorporar à mesma aspectos relativos à definição de leis em SMAs abertos. Ele apenas utiliza o mesmo mecanismo aplicado à linguagem UML: a definição de um metamodelo, para descrever os aspectos sintáticos e semânticos de LawML.

O metamodelo de LawML apresenta duas visões de modelagem: a *visão estrutural* e a *visão comportamental*. Essas visões se completam e são expressas por meio de diagramas, que são formados por elementos de notação que possuem sintaxe e semântica definidas para a linguagem.

A semântica dos elementos de notação da LawML define como as instâncias desses elementos se relacionam entre si para produzir um modelo com significado

próprio. A sintaxe desses elementos define quais construções existem e como essas são graficamente representadas.

A semântica e a sintaxe desses elementos serão apresentadas neste capítulo de maneira textual e através de uma representação gráfica. Como a linguagem é baseada no modelo conceitual da abordagem de leis, e como os modelos instanciados são mapeados diretamente para a linguagem declarativa XMLaw, os conceitos semânticos dos elementos de lei já estão descritos no modelo conceitual da abordagem de leis (Paes, 2005a; Paes et al., 2005b) e foram apresentados no Capítulo 2 desta dissertação.

4.2.
A Estrutura do Meta-modelo de LawML

O metamodelo de LawML apresenta duas visões de modelagem que se completam e que permitem a modelagem dos vários aspectos de uma lei: a *visão estrutural* e a *visão comportamental*.

A *visão estrutural* define a estrutura estática da lei, descrevendo as instâncias dos elementos de lei definidos no metamodelo, os valores de seus atributos e seus relacionamentos estáticos. Essa visão possui o *Diagrama de Instâncias de Elementos de Lei* (DIEL). Esse diagrama permite a descrição estática das instâncias dos elementos que podem ser encontrados na lei, nas cenas e nos protocolos. Esse diagrama é baseado no Diagrama de Objetos da UML, mas restrito à modelagem de leis em XMLaw.

A *visão comportamental* apresenta a estrutura dinâmica da lei. Ela permite a especificação dos protocolos de interação entre agentes e a ativação/desativação de elementos de lei a partir de eventos gerados por outros elementos de lei. Essa visão possui o *Diagrama de Estados de Protocolo* (DEP) e o *Diagrama de Comunicação de Instâncias de Elementos de Lei* (DCIEL), que, respectivamente, são baseados no Diagrama de Estados e Diagrama de Comunicação da UML, mas restritos à modelagem de leis em XMLaw.

4.3.
O Problema Exemplo

A apresentação da LawML será feita através da modelagem passo a passo da lei do módulo de compra de produto de um sistema multi-agente – o Aeroporto (Paes, 2005a). Conforme os conceitos teóricos dos diagramas são apresentados, serão exemplificados através da modelagem desse problema exemplo, descrito a seguir.

Atualmente, os grandes aeroportos fazem mais do que simplesmente servir de lugar para aterrissagem e decolagem de aviões. *Shopping centers* com centenas de lojas, cinemas, hotéis, centros empresariais, e até mesmo centros gastronômicos, são algumas das atrações dos aeroportos atuais. O grande número de usuários em potencial e a variedade de serviços oferecidos em um mesmo local fazem dos aeroportos um bom domínio para o desenvolvimento de aplicações para a compra de produtos e até mesmo para a compra de passagens aéreas.

O cenário adotado nesse problema exemplo simula a situação em que uma pessoa chega a um aeroporto com um dispositivo móvel (Figura 4-1).

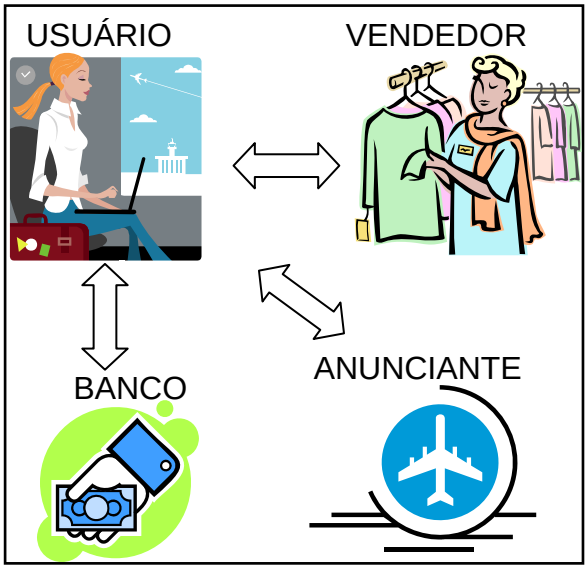


Figura 4-1 – Compra de um Produto em um Aeroporto

O aeroporto é dotado de uma rede de sistemas embarcados que fornece serviços de relacionamentos pessoais, cinema, compras e localização. Ao chegar ao shopping, o usuário requisita quais são os serviços disponíveis naquele momento. O anúncio da lista de serviços disponíveis é feito por um servidor do próprio aeroporto, denominado anunciante. Uma vez que um dos serviços é

escolhido pelo usuário, o anunciante informa ao mesmo quais são as opções para o serviço selecionado. Por exemplo, se o serviço selecionado for de compras, então será exibida uma lista de produtos disponíveis para a compra.

Nesse problema exemplo, o escopo é limitado ao serviço de compra, uma vez que o desenvolvimento dos outros serviços seria realizado de forma análoga e não traria maiores benefícios ao entendimento da linguagem LawML.

Ao selecionar um dos produtos da lista de compras, o usuário recebe uma lista de todos os vendedores que possuem aquele produto. A partir desse momento, o usuário escolhe um dos vendedores e inicia o processo de negociação, que se for bem sucedido será seguido de um processo de pagamento. Para o pagamento, um outro elemento do aeroporto aparece: o banco. O banco também provê serviços de pagamentos de mercadorias via sistema embarcado. Dessa forma, o usuário paga o valor negociado ao banco e apresenta o recibo eletrônico para receber a mercadoria.

O sistema do aeroporto deve inspirar um bom nível de confiança, no sentido em que os clientes devem ser protegidos contra comportamentos maliciosos das empresas situadas no aeroporto e as empresas também devem ser protegidas contra usuários maliciosos. Uma vez que ambas as partes saibam que os elementos do sistema estão interagindo mediante regras bem definidas e públicas, a confiança no sistema tende a crescer.

4.4. Diagramas

4.4.1. Diagrama de Instâncias de Elementos de Lei

O objetivo desse diagrama é a modelagem estrutural de instâncias de elementos de lei, do detalhamento de seus elementos, através da atribuição de valores aos atributos e do relacionamento estrutural entre elementos. Ele é baseado no Diagrama de Objetos da UML. Cada nó representa a instanciação de um elemento de lei definido no metamodelo e são chamados instâncias de elementos de lei. Os arcos do diagrama são instâncias dos relacionamentos encontrados no metamodelo e são chamados de vínculos, representados por linhas contínuas. É permitida a existência de relacionamentos estruturais – associações – entre as instâncias. Não são permitidos relacionamentos de dependência, de realização ou de generalização entre as instâncias. Além disso, é permitida a existência de instâncias de classes de associação[4].

A Figura 4-2, a seguir, apresenta uma visão geral do metamodelo do Diagrama de Instâncias de Elementos de Lei.

Ao especificar um Diagrama de Instâncias de Elementos de Lei, deve-se instanciar os elementos e relacionamentos do metamodelo, atribuindo-se valores aos seus atributos.

De acordo com o modelo conceitual da abordagem de regulação, devem ser instanciados:

- uma *Organização de Lei*;
- os *Papéis* vinculados com a *Organização de Lei*;
- as *Cenas* vinculadas com a *Organização de Lei*;
- as *Ações* vinculadas com uma *Cena* ou com a *Organização de Lei*;
- os *Relógios* vinculados com uma *Cena* ou com a *Organização de Lei*;
- as *Normas* vinculadas com uma *Cena* ou com a *Organização de Lei*. Uma *Norma* pode possuir um *vínculo* com um *Papel* de nome *Assinatura*, relacionado por uma instância de classe de associação *Assinatura*;

[4] Uma classe de associação pode ser vista como uma associação que também tem propriedades de classe e é representada com um símbolo de classe anexado a uma associação por uma linha tracejada (Booch et al., 2006).

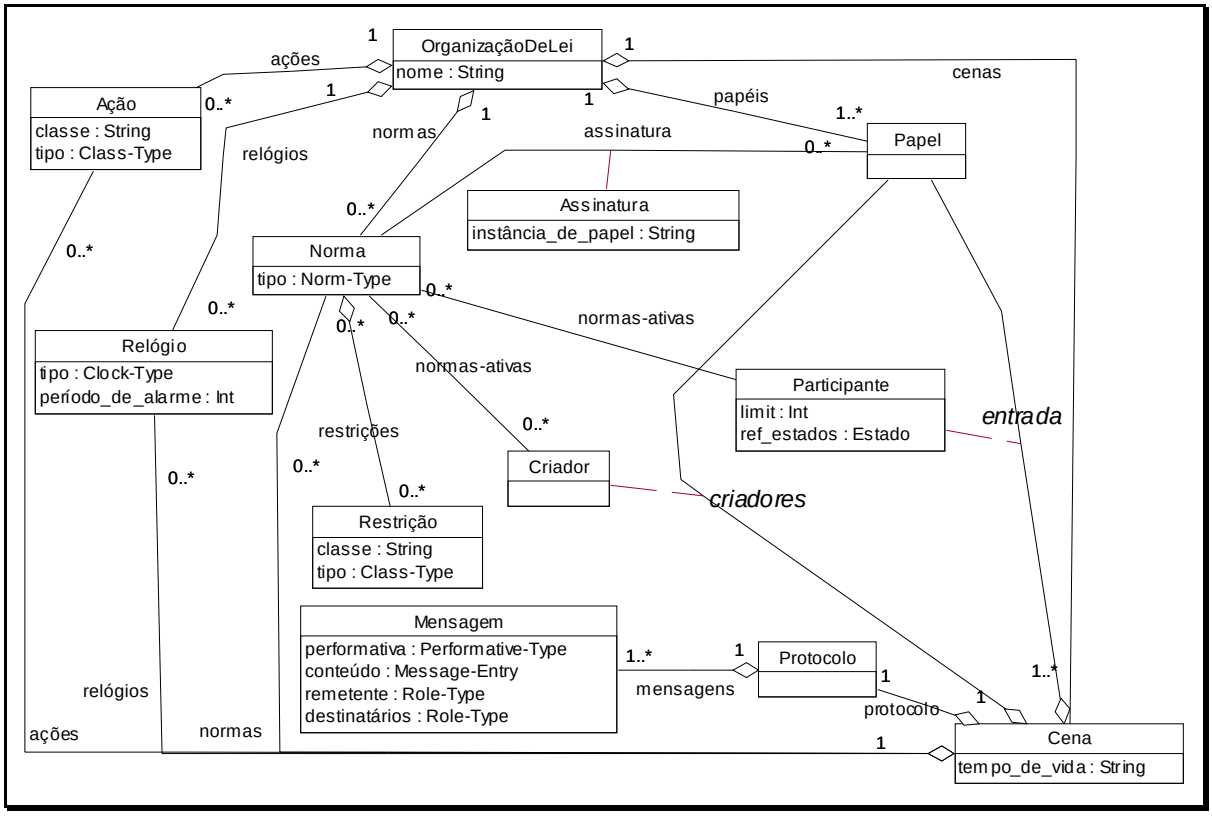


Figura 4-2 - Estrutura principal do metamodelo do Diagrama de Instâncias de Elementos de Lei

- um *Protocolo* vinculado com uma *Cena*. Todas as *Cenas* devem possuir um *vínculo* com um *Protocolo* diferente;

- as *Mensagens* com um *vínculo* com um *Protocolo*;

- os *Criadores* associados aos vínculos *criadores* entre *Cena* e *Papel*. O *Criador* pode ter *vínculos* com *Normas*, pelo atributo *normas ativas*;

- os *Participantes* associados aos vínculos de *entrada* entre *Cena* e *Papel*. O *Participante* pode ter *vínculos* com *Normas*, pelo atributo *normas ativas* e devem referenciar *Estados* através da atribuição de valor ao atributo *ref-estados*. Esse *Estado* deve ser especificado no Diagrama de Estados de Protocolo do *protocolo* relacionado;

- as *Restrições* vinculadas com uma *Norma*;

Os atributos de cada instância de elemento de lei são valorados, conforme o modelo conceitual da Figura 4-2. A valoração dos atributos deve seguir os tipos especificados nesse diagrama. Os novos tipos e os tipos enumerados estão especificados no modelo da Figura 4-3:

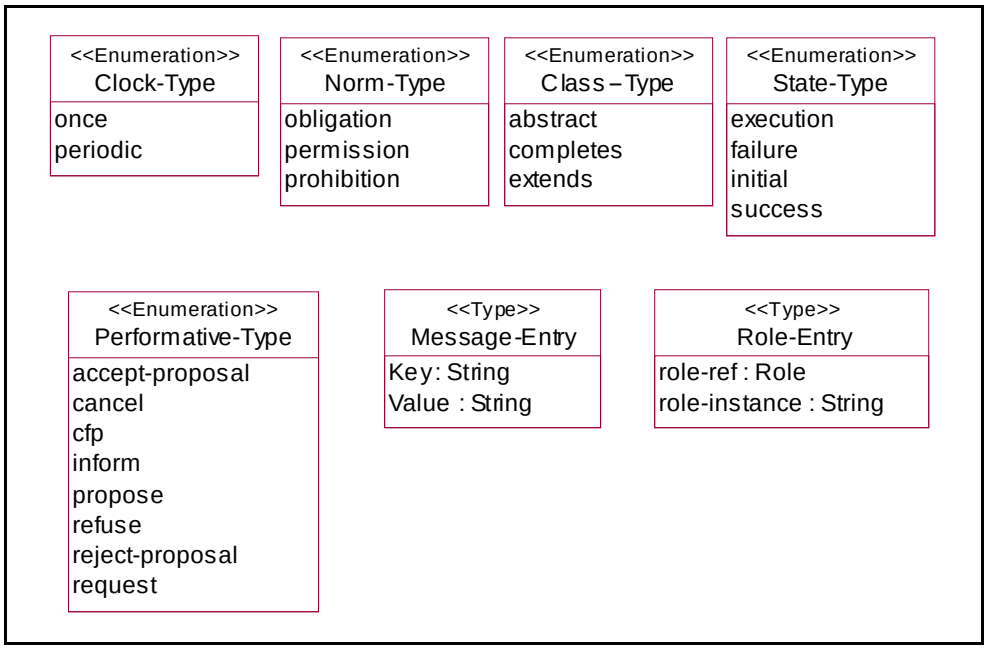


Figura 4-3 - Novos tipos de dados de LawML

A Figura 4-4 complementa o modelo conceitual. Ela apresenta quais os elementos podem ser *nós* nesse diagrama, quais podem ser *geradores de evento*; quais os elementos *ativáveis* e quais os elementos *desativáveis*.

Nó_de_instância são as instâncias de elementos de lei que podem ser nós nesse diagrama e possuem o atributo *id*, que representa o identificador da instância e deve ser único para cada instância dentro de um modelo de lei.

Além dos *vínculos* já citados, outros *vínculos* podem ser criados entre as instâncias, seguindo o modelo conceitual (Figura 4-4): o *vínculo* de *ativação* e o de *desativação.* O *vínculo* de *ativação* representa o relacionamento estático de ativação de instância a partir de evento gerado por outra instância. Ele conecta a instância de elemento *gerador de evento* com a instância *ativável*. De maneira análoga, o *vínculo* de *desativação* representa o relacionamento estático de desativação de instância a partir de evento gerado por outra instância e conecta a instância de elemento *gerador de evento* com a instância *desativável*.

O *vínculo* especifica que a instância ativadora/desativadora pode gerar um evento que vai ser responsável pela ativação/desativação da instância ativável/desativável e deve ter o estereótipo <<ativação>> *ou* <<desativação>>.

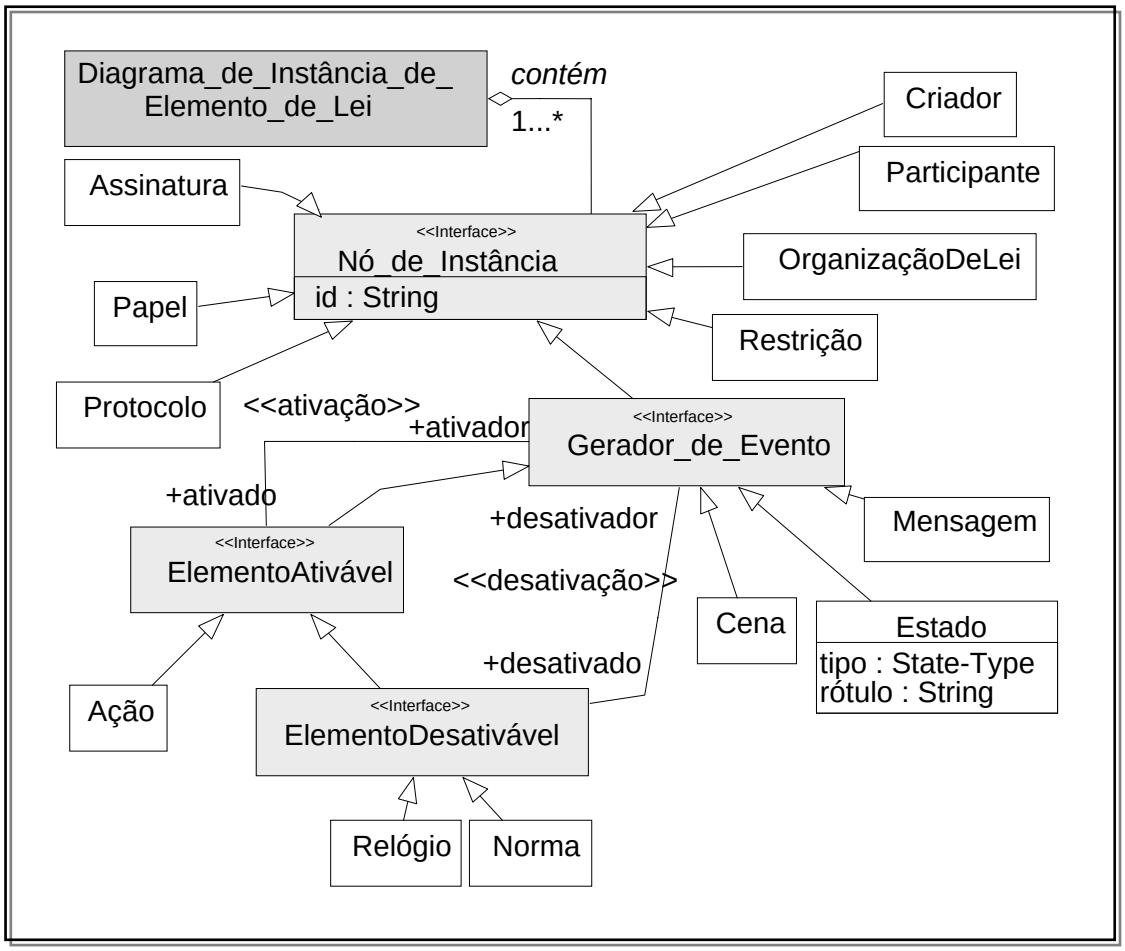


Figura 4-4 - Complemento do metamodelo do Diagrama de Instâncias de Elementos de Lei

Nesse diagrama se representam apenas os vínculos (relacionamentos estáticos). Os eventos relacionados aos vínculos, devem ser especificados no Diagrama de Comunicação de Instâncias de Elementos de Lei.

Observe que o elemento *Estado* também pode ser instanciado nesse diagrama, como instância de elemento *gerador de evento*.

4.4.1.1. Sintaxe

Cada instância de elemento de lei é representada sintaticamente de maneira semelhante ao Diagrama de Objetos da UML: por um retângulo com um rótulo em sublinhado indicando o *Id* da instância, seguido por dois pontos ":" e o elemento que está sendo instanciado, como pode ser visto na Figura 4-5.

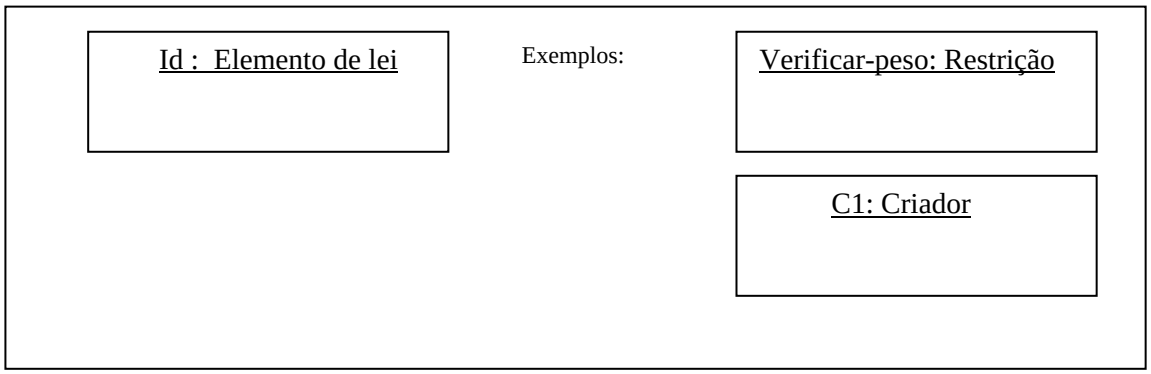


Figura 4-5 - Notação das Instâncias.

A atribuição de valores aos atributos dos elementos é feita através da subdivisão do retângulo em dois compartimentos (semelhante à UML). Cada atributo é escrito seguido de um sinal de igual "=" e da atribuição de valor ao atributo, como pode ser visto na Figura 4-6.

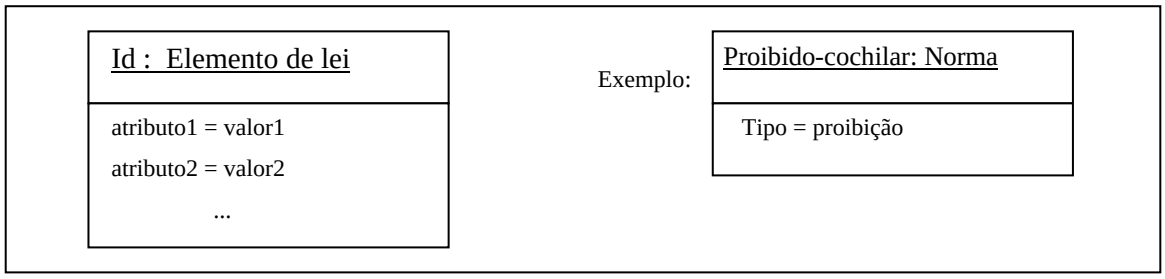


Figura 4-6 – Notação da atribuição de valor aos atributos.

Essa representação é opcional, pois esse compartimento pode ser omitido para facilitar a visualização. Isso não significa que não haja atribuição de valores, apenas que não se deseja mostrá-la no diagrama. O *id* é um exemplo de atributo que não é especificado nesse compartimento, já que ele aparece nomeando a instância[5].

A representação dos atributos da *Mensagem* é feita conforme a seguinte regra: cada elemento do vetor é colocado entre chaves "< >" e separado por vírgulas ","; dentro das chaves separam-se os valores por vírgulas. Veja na Figura 4-7 a especificação da *Mensagem* de *id*= m1, de performativa "inform". Essa mensagem informa o "preço" e o "prazo" de um produto. Ela é enviada por um agente de papel "vendedor" de instância desconhecida, para dois agentes de papel "comprador": "José" e Carlos".

[5] Observe que na UML o nome da instância não precisa ter relação com algum atributo valorado.

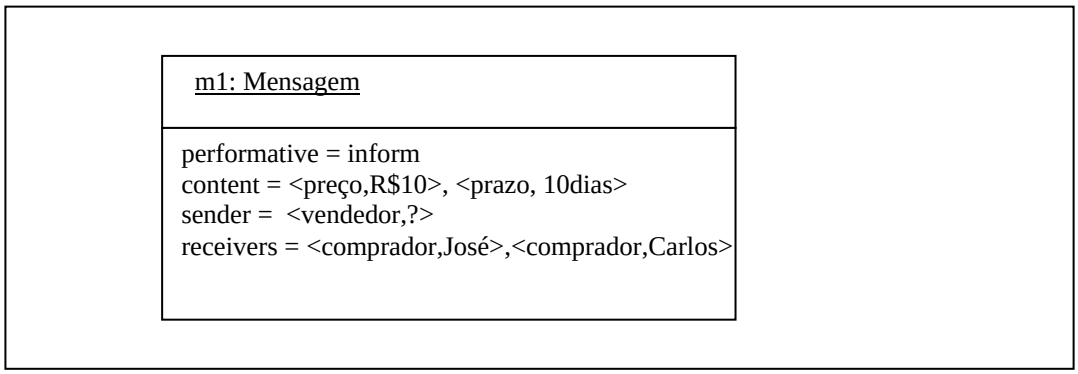


Figura 4-7 - Notação de uma instância de Mensagem

Não serão permitidos outros compartimentos, como o de métodos em UML.

Os vínculos entre as instâncias de elementos de lei são representados como em UML – uma linha conectando as instâncias. Os vínculos devem receber o mesmo nome, estereótipo, papéis e adorno que a associação relacionada, especificada no metamodelo. O nome do vínculo pode ser omitido quando não se fizer necessário para o entendimento do modelo. Veja um exemplo de notação de vínculos na Figura 4-8.

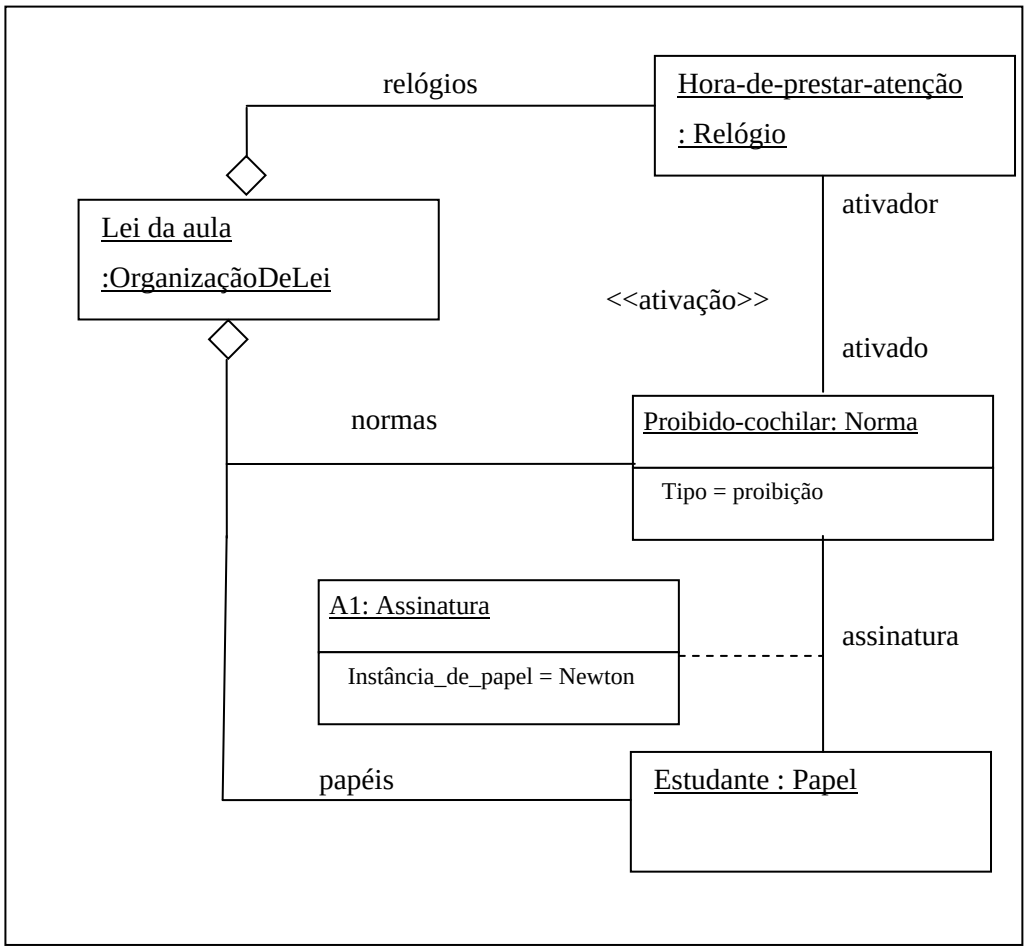


Figura 4-8 - Exemplo de notação de vínculos.

4.4.1.2. Regras de Transformação

O mapeamento dos elementos desse diagrama para código no formato XMLaw é feito seguindo as seguintes regras de transformação:

DIEL01. *Organização de Lei*

- O elemento *Organização de Lei* é mapeado diretamente para o elemento de código *XMLaw LawOrganization*, dentro do elemento *Laws*, raiz do documento XML.

DIEL02. *Papel*

- O *Papel* é mapeado diretamente para o elemento de código *XMLaw Role*, dentro do elemento *XMLaw LawOrganization*, com o qual o *Papel* contém um *vínculo*.

DIEL02.a. *Vínculo* entre *Papel* e *Organização de Lei*

- O *vínculo* do *Papel* com a *Organização de Lei* apenas indica a qual Organização de Lei o Papel pertence, ou seja, onde o elemento de código *XMLaw Role* será criado;

DIEL03. *Cena*

- A *Cena* é mapeada diretamente para o elemento de código *XMLaw Scene*, dentro do elemento *XMLaw LawOrganization*, com o qual a *Cena* contém um *vínculo*.

DIEL03.a. *Vínculo* entre *Cena* e *Organização de Lei*

- O *vínculo* da *Cena* com a *Organização de Lei* apenas indica a qual Organização de Lei a *Cena* pertence, ou seja, onde o elemento de código *XMLaw Scene* será criado;

DIEL04. *Ação*

- A *Ação* é mapeada diretamente para o elemento de código XMLaw *Action*, dentro do elemento *XMLaw LawOrganization* ou do elemento *XMLaw Scene*, com o qual a *Ação* contém um *vínculo*.

DIEL04.a. *Vínculo* entre *Ação* e *Organização de Lei*

- O *vínculo* da *Ação* com a *Organização de Lei* apenas indica a qual *Organização de Lei* a *Ação* pertence, ou seja, onde o elemento de código *XMLaw Action* será criado;

DIEL04.b. *Vínculo* entre *Ação* e *Cena*

- O *vínculo* da *Ação* com a *Cena* apenas indica a qual *Cena* a *Ação* pertence, ou seja, onde o elemento de código *XMLaw Action* será criado;

DIEL05. *Relógio*

- O *Relógio* é mapeado diretamente para o elemento de código *XMLaw Clock*, dentro do elemento *XMLaw LawOrganization* ou do elemento *XMLaw Scene*, com o qual o *Relógio* contém um *vínculo*.

DIEL05.a. *Vínculo* entre *Relógio* e *Organização de Lei*

- O *vínculo* do *Relógio* com a *Organização de Lei* apenas indica a qual *Organização de Lei* o *Relógio* pertence, ou seja, onde o elemento de código *XMLaw Clock* será criado;

DIEL05.b. *Vínculo* entre *Relógio* e *Cena*

- O *vínculo* do *Relógio* com a *Cena* apenas indica a qual *Cena* o *Relógio* pertence, ou seja, onde o elemento de código *XMLaw Clock* será criado;

DIEL06. *Norma*

- A *Norma* é mapeada diretamente para o elemento de código *XMLaw Norm*, dentro do elemento *XMLaw LawOrganization* ou do elemento *XMLaw Scene*, com o qual a *Norma* contém um *vínculo*.

DIEL06.a. *Vínculo* entre *Norma* e *Organização de Lei*

- O *vínculo* da *Norma* com a *Organização de Lei* apenas indica a qual Organização de Lei a *Norma* pertence, ou seja, onde o elemento de código *XMLaw Norm* será criado;

DIEL06.b. *Vínculo* entre *Norma* e *Cena*

- O *vínculo* da *Norma* com a *Cena* apenas indica a qual *Cena* a *Norma* pertence, ou seja, onde o elemento de código *XMLaw Norm* será criado;

DIEL06.c. *Vínculo assinatura* entre *Norma* e *Papel* e instância de classe de associação *Assinatura*

- O *vínculo assinatura* da *Norma* com o Papel e instância de classe de associação *Assinatura* indicam que dentro do elemento de

código *XMLaw Norm* será criado um elemento de código *XMLaw Assignee*, com o atributo *role-ref* recebendo a referência ao *Papel* (o *id* do *Papel).*

DIEL07. *Protocolo*

- O *Protocolo* é mapeado diretamente para o elemento de código *XMLaw Protocol*, dentro do elemento *XMLaw Scene*, com o qual o *Protocolo* contém um *vínculo*.

DIEL07.a. *Vínculo* entre *Protocolo* e *Cena*

- O *vínculo* do *Protocolo* com a *Cena* apenas indica a qual Cena o *Protocolo* pertence, ou seja, onde o elemento de código *XMLaw Protocol* será criado;

DIEL08. *Mensagem*

- A *Mensagem* é mapeada diretamente para o elemento de código *XMLaw Message*, dentro do elemento *XMLaw Protocol*, com o qual a *Mensagem* contém um *vínculo*.

DIEL08.a. *Vínculo* entre *Mensagem* e *Protocolo*

- O *vínculo* da *Mensagem* com o *Protocolo* apenas indica a qual *Protocolo* a *Mensagem* pertence, ou seja, onde o elemento de código *XMLaw Message* será criado;

DIEL09. Instância de classe de associação de *Criador* e *vínculo* entre *Cena* e *Papel*

- O *Criador* é mapeado diretamente para o elemento de código *XMLaw Creator*, dentro do elemento *XMLaw Creators*, por sua vez dentro do elemento *XMLaw Scene*, com o qual o *Papel* contém um *vínculo*.

- O *Papel* indica o valor do atributo *XMLaw role-ref* do *Criador*, que é uma referência ao *Papel* (o *id* do *Papel);*

DIEL09.a. *Vínculo normas ativas* entre *Criador* e *Norma*

- O *vínculo normas ativas* do *Criador* com a *Norma* indica que dentro do elemento de código *XMLaw Creator*, deverá existir um elemento de código *XMLaw ActiveNorms* contendo um elemento de código *XMLaw Norm;* o atributo *ref* desse elemento *Norm* recebe uma referência a *Norma* (o *id* da *Norma);*

DIEL10. Instância de classe de associação de *Participante* e *vínculo* entre *Cena* e *Papel*

- O *Participante* é mapeado diretamente para o elemento de código *XMLaw Participant*, dentro do elemento *XMLaw Entrance*, por sua vez dentro do elemento *XMLaw Scene*, com o qual o *Papel* contém um *vínculo*.

- O *Papel* indica o valor do atributo *XMLaw role-ref* do *Participante*, que é uma referência ao *Papel* (o *id* do *Papel);*

DIEL10.a. *Vínculo normas ativas* entre *Participante* e *Norma*

- O *vínculo normas ativas* do *Participante* com a *Norma* indica que dentro do elemento de código *XMLaw Participant*, deverá existir um elemento de código *XMLaw ActiveNorms* contendo um elemento de código *XMLaw Norm;* o atributo *ref* desse elemento *Norm* recebe uma referência à *Norma* (o *id* da *Norma);*

DIEL10.b. *Atributo ref_estados* do *Participante*

- O *atributo ref_estados* do *Participante* contém uma lista de referência a uma ou mais instâncias de *Estado;* para cada uma, deve existir um elemento de código *XMLaw State* com o atributo *ref* contendo a referência ao *Estado* (o *id* do *Estado).*

DIEL11. *Restrição*

- A *Restrição* é mapeada diretamente para o elemento de código *XMLaw Constraint*, dentro do elemento *XMLaw Constraints*, por sua vez dentro do elemento *XMLaw Norm*, com o qual a *Restrição* contém um *vínculo*.

DIEL11.a. *Vínculo* entre *Restrição e Norma*

- O *vínculo* da *Restrição* com a *Norma* apenas indica a qual *Norma* a *Restrição* pertence, ou seja, onde o elemento de código *XMLaw Constraint* será criado;

DIEL12. Atributos

- Os atributos de cada instância são mapeados diretamente para atributos do elemento de código XMLaw da instância relacionada;

- O nome (*id*) de cada instância nomeada é mapeado para o atributo *id* da instância, caso ela possua atributo *id* em *XMLaw*;

DIEL13. Atributos da *Mensagem*

- O atributo *performativa* é mapeado como em **DIEL12**;

- Dentro do elemento de código *XMLaw Message* é criado elementos de código *XMLaw Content, Sender* e *Receivers;*

DIEL13.a. Atributo *Conteúdo* da *Mensagem*

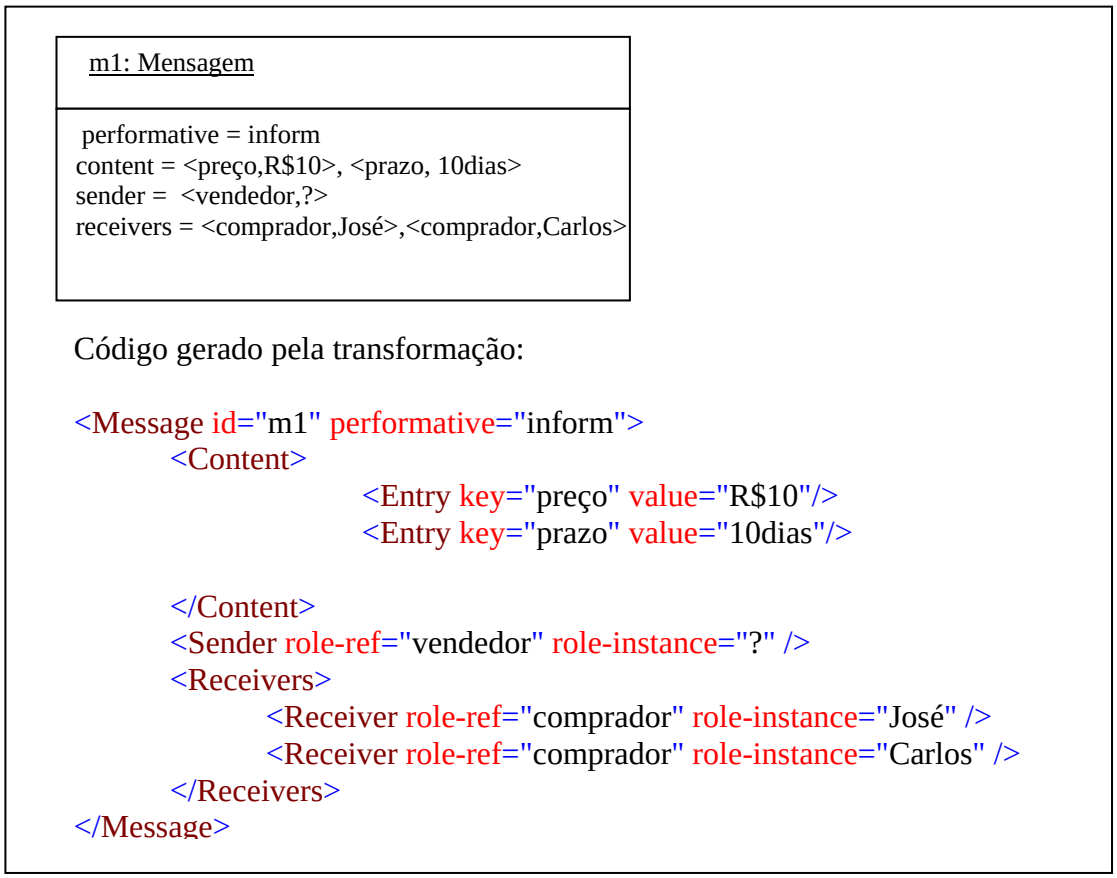
- Para cada *Message-entry* do atributo *Conteúdo*, é criado um elemento de código *XMLaw Entry* dentro do elemento *XMLaw Content*. Os atributos *chave* (key) e *valor* (value) são mapeados diretamente de *Message-entry* para o elemento de código *XMLaw Entry* relacionado;

DIEL13.b. Atributo *Remetente* da *Mensagem*

- O atributo *Remetente* da *Mensagem* é mapeado diretamente para o elemento de código *XMLaw Sender*, assim como os valores de *role-ref* (o *id* do *Papel* referenciado) e *role-instance*;

DIEL13.c. Atributo *Destinatários* da *Mensagem*

- Para cada *Role-entry* do atributo *Destinatários*, é criado um elemento de código *XMLaw Receiver* dentro do elemento *XMLaw Receivers*. Os atributos *role-ref* (o *id* do *Papel* referenciado) e *role-instance* são mapeados diretamente de *Role-entry* para o elemento de código *XMLaw Receiver* relacionado. Veja o exemplo da Figura 4-9.



Código gerado pela transformação:

```
<Message id="m1" performative="inform">
        <Content>
                        <Entry key="preço" value="R$10"/>
                        <Entry key="prazo" value="10dias"/>

        </Content>
        <Sender role-ref="vendedor" role-instance="?" />
        <Receivers>
                <Receiver role-ref="comprador" role-instance="José" />
                <Receiver role-ref="comprador" role-instance="Carlos" />
        </Receivers>
</Message>
```

Figura 4-9 - Exemplo de transformação de notação para código no formato XMLaw

4.4.1.3. Considerações

Para evitar um diagrama com um número ilegível de elementos, recomenda-se a especificação estática do sistema em diversos Diagramas de Instâncias de Elementos de Lei:

- um diagrama para a *Organização de Lei* e seus relacionamentos com outras instâncias;

- um diagrama para cada *Cena* e seus relacionamentos com outras instâncias;

- um diagrama para cada *Protocolo* e seus relacionamentos com outras instâncias ou um diagrama Contendo todos os *Protocolos* e seus relacionamentos.

Deve ser criado um Diagrama de Estados de Protocolo (veja a seção 4.4.2) para cada *Protocolo* para a sua especificação completa;

As ativações/desativações de instâncias de *Norma*, *Relógio* e *Ação* devem ser especificadas no Diagrama de Estados de Protocolo ou no Diagrama de Comunicação de Instâncias de Elementos de Lei (veja a seção 4.4.3).

4.4.1.4.
Problema do Aeroporto: Estrutura da Lei

4.4.1.4.1.
Especificação Inicial da Organização de Lei

A *Organização de Lei* desse exemplo terá o *id= "aeroporto"* e o "*nome = Santos-Dummont*" (Figura 4-10).

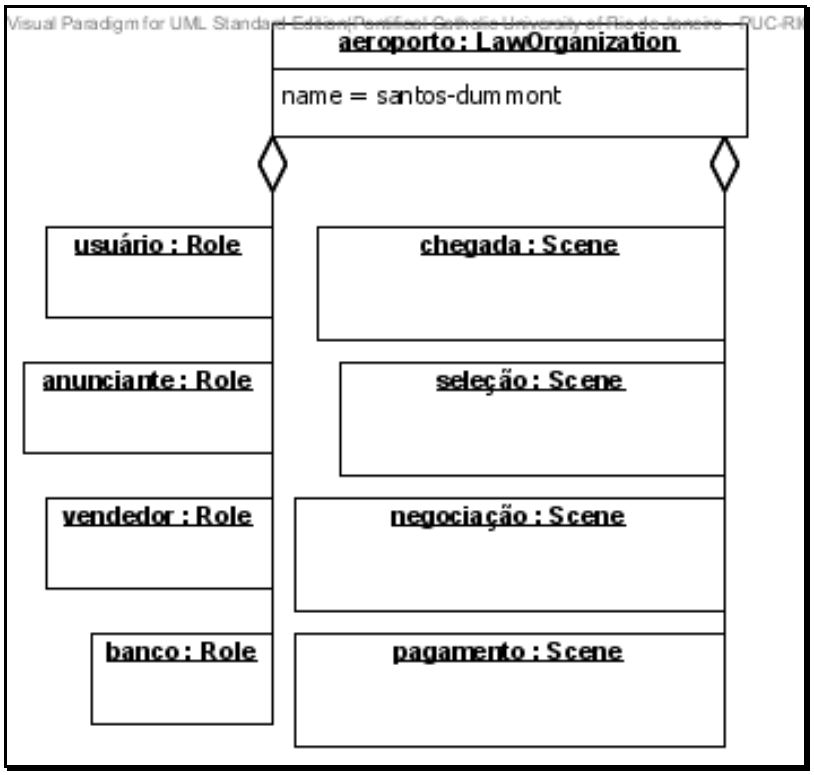


Figura 4-10 - Diagrama de Instâncias de Elementos de Lei do Aeroporto – versão inicial.

Quatro *Cenas* foram identificadas vinculadas à *Organização de Lei*: *chegada, seleção, negociação* e *pagamento.* Na primeira cena o usuário descobre quais os serviços estão disponíveis. Na segunda, ele seleciona um dos serviços disponíveis, que nesse estudo de caso será o serviço de compra, recebe uma lista de todos os produtos que estão disponíveis para a venda, escolhe um dos produtos e recebe uma lista de todos os vendedores daquele produto. Na etapa de negociação, o usuário escolhe um dos vendedores e negocia um preço para o produto. E, finalmente, na última etapa o valor negociado do produto é pago ao banco.

Para essas cenas, foram identificados quatro *Papéis* de agentes no sistema: *usuário, anunciante, vendedor* e *banco.*

A princípio, não foram identificados nenhum *Relógio*, *Norma* ou *Ação*. Mais tarde, caso sejam identificados, devem ser acrescentados no Diagrama de Instâncias de Elementos de Lei do Aeroporto, assim como os atributos das cenas.

4.4.1.4.2. Especificação Inicial das Cenas

Visando facilitar o trabalho de modelagem, para cada *Cena* deve ser criado um Diagrama de Instâncias de Elementos de Lei. Portanto devem ser criados mais quatro Diagramas Instâncias de Elementos de Lei.

Cada *Cena* deve especificar o *Protocolo* de interação que os agentes devem seguir, quais são os *Criadores* permitidos e os agentes *Participantes*.

4.4.1.4.2.1. A Cena Chegada

Nessa *Cena*, considera-se arbitrariamente que todo o processo não deve durar mais do que 10 segundos (atributo *time-to-live*). Veja a Figura 4-11.

Especifica-se o *Protocolo* com o id *protocolo-de-chegada*.

A *Cena* pode ser instanciada por qualquer agente desempenhando qualquer *Papel* válido. Portanto, deve ser especificada uma instância de *Criador* para cada *Papel* de agente.

Apenas agentes desempenhando *papéis* de *usuário* e *anunciante* podem participar das interações. Após a modelagem do *Protocolo* dessa *cena* – com o Diagrama de Instâncias de Elementos de Lei do *Protocolo* e o Diagrama de Estados de Protocolo, será possível especificar em que estado cada *Participante* pode entrar.

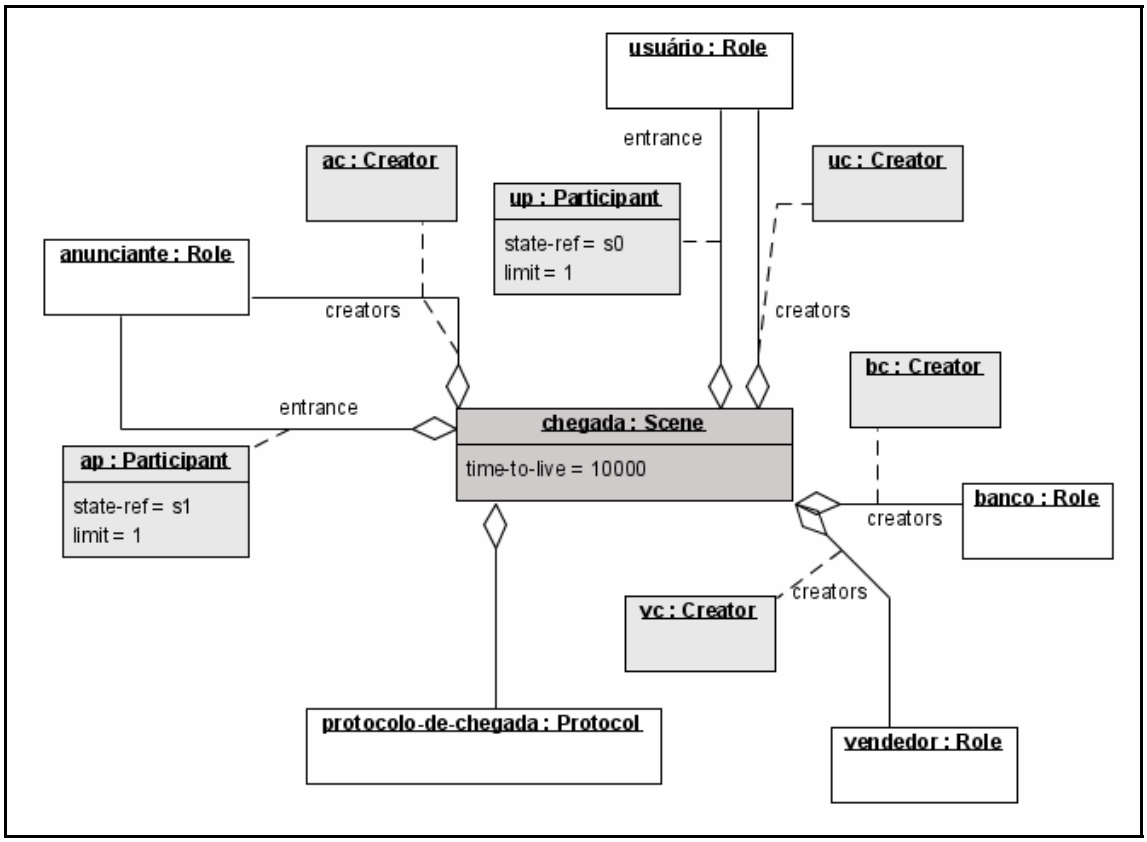


Figura 4-11 - Diagrama de Instâncias de Elementos de Lei da Cena Chegada – versão inicial.

A especificação de possíveis outros elementos – *Relógio*, por exemplo, será permitida após a especificação do *Protocolo*.

4.4.1.4.2.2. A Cena Seleção

Essa cena tem duração de 5 minutos, que é o tempo dado ao usuário (agente usuário) para escolher entre os serviços e entre os produtos. Esse valor (5min = 5* 60*1000 =300000 ms) é atribuído ao atributo *time-to-live* da cena no Diagrama de Instâncias de Elementos de Lei.

O *Protocolo* da cena terá id *"protocolo-de-seleção"*. Qualquer agente pode criar a cena. Um agente "usuário" e um agente "anunciante" podem participar da conversação.

O Diagrama de Instâncias de Elementos de Lei dessa cena será omitido, pois a modelagem é análoga à da *Cena* "*Chegada*".

4.4.1.4.2.3.
A Cena Negociação

Essa cena não tem duração estimada, portanto especifica-se que a duração é infinita. Para tal, o valor *infinity* deve ser atribuído ao atributo *time-to-live* da cena no Diagrama de Instâncias de Elementos de Lei.

O *Protocolo* da cena terá id *"protocolo-de-negociação"*. Qualquer agente pode criar a cena. Um agente "usuário" e um agente "vendedor" podem participar da conversação.

Como na cena anterior, o Diagrama de Instâncias de Elementos de Lei será omitido, pois a modelagem é análoga à da *Cena* "*Chegada*".

4.4.1.4.2.4.
A Cena Pagamento

Essa cena não tem duração estimada, portanto especifica-se que a duração é infinita. Para tal, o valor *infinity* deve ser atribuído ao atributo *time-to-live* da cena no Diagrama de Instâncias de Elementos de Lei.

O *Protocolo* da cena terá id *"protocolo-de-pagamento"*. A criação de uma cena de pagamento só pode ser feita por um agente desempenhando o papel de usuário. Um agente "usuário" e um agente "banco" podem participar da conversação.

Como nas cenas anteriores, o Diagrama de Instâncias de Elementos de Lei será omitido, pois a modelagem é análoga à da *Cena* "*Chegada*".

4.4.1.4.3. Especificação dos Protocolos

Para cada *Protocolo* será criado um Diagrama de Instâncias de Elementos de Lei e um Diagrama de Estados de Protocolo, já que todas as cenas contêm envio de *Mensagens*. Portanto, teremos mais quatro Diagramas de Instâncias de Elementos de Lei e quatro Diagramas de Estados de Protocolo.

4.4.1.4.3.1. O Protocolo da Cena Chegada

As interações dessa cena contêm apenas duas mensagens (Figura 4.12), sendo uma do agente *usuário* para o agente *anunciante*, avisando que chegou ao aeroporto (mensagem *m1*), e a resposta do agente *anunciante* através da lista de serviços disponíveis no aeroporto (mensagem *m2*).

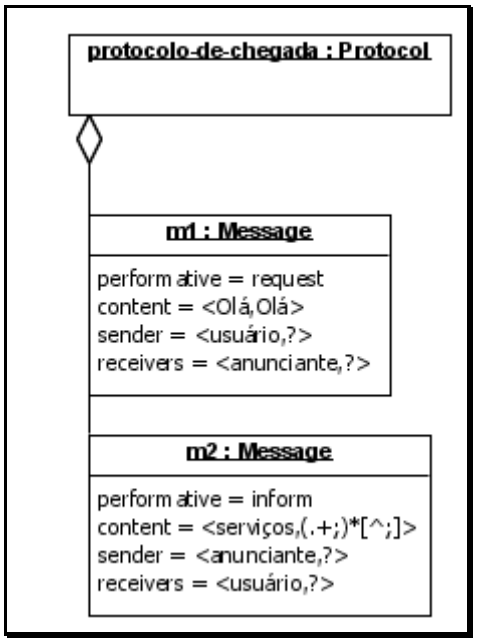


Figura 4-12 - Diagrama de Instâncias de Elementos de Lei do Protocolo *protocolo-de-chegada*

4.4.1.4.3.2. O Protocolo da Cena Seleção

A primeira mensagem (*m3*) é enviada pelo *usuário* ao *anunciante*, informando qual serviço foi escolhido. A resposta do *anunciante* contém a lista de produtos disponíveis para o serviço requisitado (mensagem *m4*), lembrando que o único serviço disponível nesse estudo de caso é o serviço de compras. O

usuário escolhe um dos produtos e requisita a lista de vendedores do produto (mensagem *m5*). Finalmente, a lista de vendedores é informada (mensagem *m6*).

Como o detalhamento de cada mensagem é análogo ao feito no *Protocolo* da *Cena Chegada*, serão omitidos, nesse problema-exemplo, os compartimentos de atributos das *Mensagens* desse e dos próximos *Protocolos*, a fim de reduzir o escopo e facilitar a visualização. Além disso, a especificação estrutural desses *Protocolos* será omitida, pois é análoga à da *Cena* "*Chegada*".

4.4.1.4.3.3. O Protocolo da Cena Negociação

O *Protocolo* de interação utilizado para a cena de negociação é baseado no FIPA-CONTRACT-NET (FIPA, 2002). A primeira mensagem (*m7*) é enviada pelo *usuário* para o *vendedor* requisitando uma proposta para o produto selecionado (cfp, ou call-for-proposal). O *vendedor* pode responder com uma proposta de compra (mensagem *m8*) ou pode negar enviando a mensagem *m11*.

Se o *usuário* aceitar a proposta (mensagem *m9*), o *vendedor* informa onde o pagamento deve ser feito (mensagem *m10*). O agente *usuário* pode, por outro lado, rejeitar a proposta (mensagem *m12*). Caso ele demore em responder, conforme será especificado no Diagrama de Estados de Protocolo desse *Protocolo*, o *vendedor* pode cancelar a proposta feita (mensagem *m13*).

Como no *Protocolo* da cena *Seleção*, o detalhamento de cada mensagem (compartimento de atributos) e a modelagem serão omitidos.

4.4.1.4.3.4. O Protocolo da Cena Pagamento

Nessa cena, o *usuário* solicita ao *banco* (mensagem *m14*) o serviço de pagamento em nome do *vendedor* do *Protocolo* anterior, e o *banco* informa ao *usuário* (mensagem *m15*) que recebeu o pagamento.

Como no *Protocolo* das cenas *anteriores*, o detalhamento de cada mensagem (compartimento de atributos) e a modelagem serão omitidos.

4.4.2. Diagrama de Estados de Protocolo

O objetivo desse diagrama é a modelagem comportamental dos elementos dinâmicos que compõem o *Protocolo.* Esse diagrama nada mais é do que o Diagrama de Estados da UML (Booch et all, 2006), com algumas restrições, que permite modelar uma máquina de transição de estados do *Protocolo* de interação entre os agentes. Cada Diagrama de Estados de Protocolo deve referenciar um *Protocolo* especificado no Diagrama de Instâncias de Elementos de Lei.

A semântica dos elementos desse diagrama é semelhante à da UML, com algumas diferenças. O *Estado* possui atributos a serem especificados: *id* e *rótulo*. Além disso, o *Estado Final* pode ser *de Sucesso* ou *de Falha*.

A ativação de uma *Transição* a partir de um evento é especificada analogamente a como é feito pela UML, através do *evento de ativação*.

Cada elemento pode gerar eventos de tipos específicos. A Tabela 4-1 apresenta o resumo dos eventos cada elemento pode gerar.

Elemento	Tipo de Evento
Ação	*action_activation*
Cena	*time_to_live_elapsed, failure_scene_completion, successful_scene_completion*
Estado	*final_state_reached*
Mensagem	*message_arrival*
Norma	*norm_activation, norm_deactivation*
Relógio	*clock_activation, clock_deactivation, clock_tick, clock_timeout*

Tabela 4-1 - Tipos de eventos gerados pelos elementos de lei.

A ativação de uma *Transição* pode ter como efeito a *ativação* de uma (ou mais) *Norma, Relógio* ou *Ação* (elementos de lei), e ainda pode ter como efeito a *desativação* de uma (ou mais) *Norma* ou *Relógio*. Analogamente à UML, que permite especificar uma ação (computação atômica executável, em UML) associada à ativação de uma transição (especificamente a ação de envio de sinal a um objeto), em LawML pode-se especificar a *ativação ou desativação de elementos de lei* (locais ou globais) como efeito da ativação de uma *Transição*.

LawML oferece dois tipos de *condições de guarda*: as *Normas Ativas* e as *Restrições***.** Se estiver desativada pelo menos uma das *Normas* (elementos do modelo, locais ou globais) que foram especificadas como *Normas Ativas* da *Transição,* essa *Transição* não é ativada e o *Protocolo* permanece no mesmo *Estado*. Da mesma maneira, o código das *Restrições* deve ser executado para avaliar se a *Transição* pode ser ativada. Essas condições de proteção são relacionadas à *Transição* que se pretende restringir. Em UML, é necessário se especificar uma expressão *booleana* para se validar se a *Transição* será ativada. No entanto, em LawML, basta se especificar as *Normas* e as *Restrições*.

As *Normas Ativas* referenciam *Normas* especificadas no Diagrama de Instâncias de Elementos de Lei e as *Restrições* devem ser especificadas no Diagrama de Instâncias de Elementos de Lei, como nós desconectados.

Diferentemente da UML, a LawML não permite a especificação de transições internas, de subestados (estados aninhados), de ações de entrada/saída, de eventos adiados, de efeitos de entrada/saída, de atividades, de estados de histórico, de bifurcação ou de união. Não é permitido também especificar outros efeitos das transições além dos relatados nesta seção. Caso, com a evolução da linguagem, verifique-se a necessidade desses elementos, a linguagem XMLaw e o modelo conceitual deverão ser modificados para atender aos novos requisitos.

4.4.2.1. Sintaxe

Sintaticamente, cada instância de *Estado* é representada da mesma maneira que na UML.

- Se for *Estado Inicial,* usa-se um círculo preenchido, como pode ser visto na Figura 4-13;

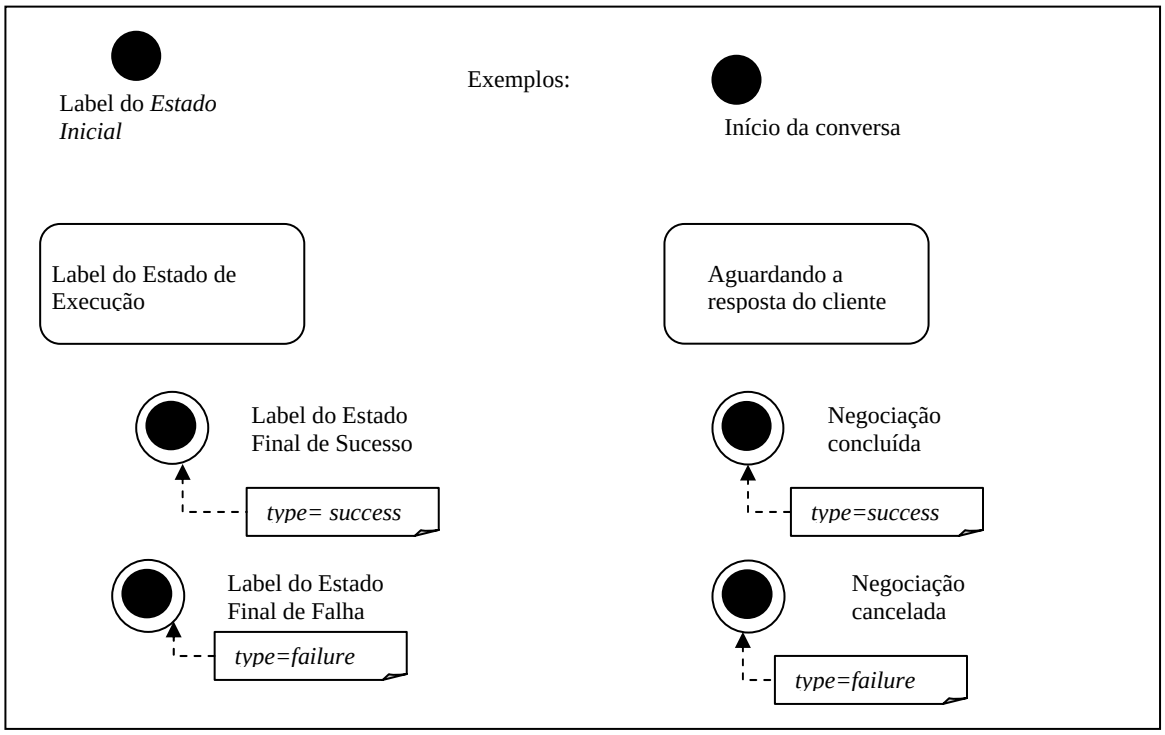


Figura 4-13 - Notação de Estado.

- Se for *Estado de Execução*, usa-se um retângulo com as bordas abauladas;

- Se *Estado Final (de Falha* ou *Sucesso)*, usa-se um *"olho de boi"*. Para o *Estado Final de Falha* acrescenta-se o texto "*type=failure*" à documentação interna do estado e para o *Estado Final de Sucesso*, o texto "*type=success*".

- Pode-se exibir um rótulo contendo o atributo "*Rótulo*", para dar semântica adicional à figura do estado.

- O atributo *id* do *Estado* deve ser acrescentado à documentação interna do estado, com o texto "*id=id_do_estado*".

- Cada instância de *Transição* é representada como uma seta fechada preenchida, com um rótulo para seu *id*, de maneira análoga a como é feito na UML, como pode ser visto na Figura 4-14.

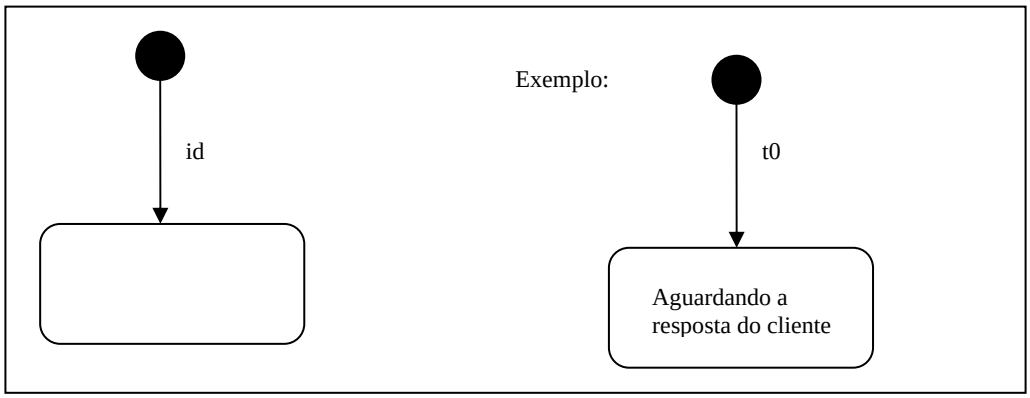


Figura 4-14 - Notação de Transição

Em UML, utilizam-se rótulos para se especificar o evento de ativação, uma ação e uma condição de guarda. Em LawML, também são utilizados rótulos, conforme as seguintes regras:

- Especifica-se o *evento de ativação* após o *id* da *Transição* com um rótulo contendo "dois pontos", o *id da instância* que gerou o evento, seguido de um "ponto" e o *tipo de evento* gerado. Veja a Figura 4-15;

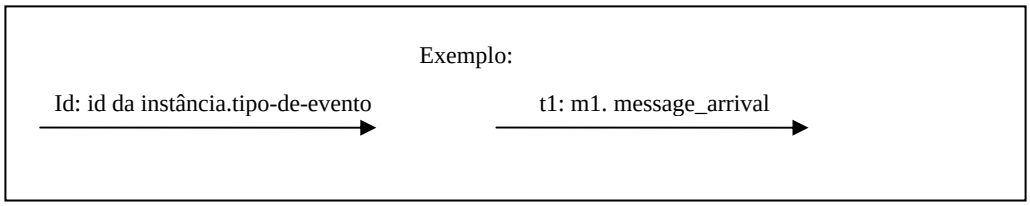


Figura 4-15 - Notação de evento de ativação de uma Transição

- Especifica-se o efeito de *ativação ou desativação de elementos de lei* após o *evento de ativação* da *Transição* (separados por uma "barra") com um rótulo contendo a *referência* à instância que será ativada (ou desativada), seguido de um ponto e um rótulo com o *tipo de evento* gerado como efeito. Essa notação é semelhante à especificação de um dos efeitos das transições em UML – a ação de envio de sinal a um objeto. Veja a Figura 4-16.

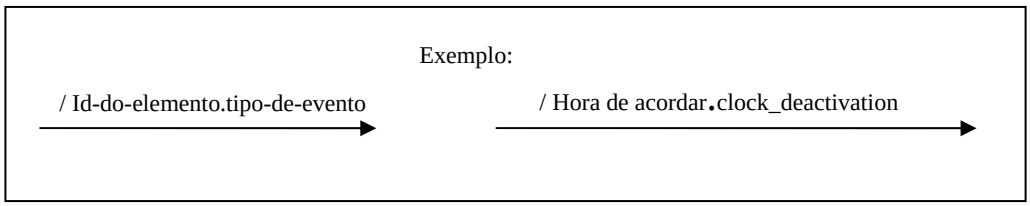


Figura 4-16 - Notação de ativação/desativação de elemento como resultado da ativação da Transição.

- Especifica-se as *condições de proteção* com um rótulo formado por um par de chaves "[...]" informando se são *Normas Ativas* ou *Restrições*, dois pontos e os *elementos* (*referências* às Normas ou instâncias de *Restrição*) separados por vírgulas. Veja a Figura 4-17.

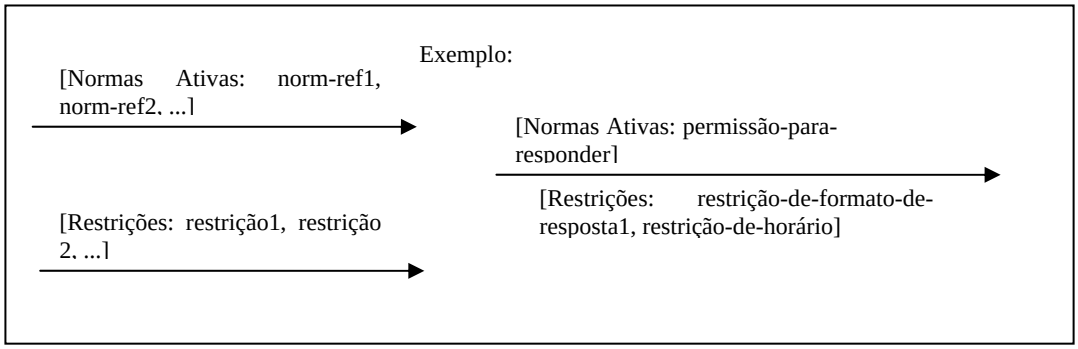


Figura 4-17 - Notação de condição de proteção.

Cada instância de *Restrição* e de *Norma* devem ser representadas no Diagrama de Instâncias de Elementos de Lei.

Cada Diagrama de Estados de Protocolo deve conter uma nota que referencia um Protocolo especificado no Diagrama de Instâncias de Elementos de Lei, como na Figura 4-18.

4.4.2.2. Regras de Transformação

Para o melhor entendimento das regras de transformação será apresentada a transformação de um exemplo simples (Figura 4-18). Esse exemplo apresenta o *Protocolo* "*conversa"*, formado pelos *Estados* "*si*", "*s_pergunta_feita*" e "*sf*". Quando a *Mensagem* "*pergunta*" é enviada, a *Restrição* "*validar_pergunta*" é testada; em caso de sucesso, a *Transição* "*t0*" é ativada e o *Protocolo* passa para o *Estado* de *rótulo* "*Pergunta feita*". A partir desse *Estado*, quando a *Mensagem* "*resposta*" é enviada, a *Transição* "*t1*" é ativada, a ativação dessa *Transição* aciona a ativação da *Ação* "*guardar_resposta*" e o *Protocolo* passa para o *Estado final de sucesso* "*sf*".

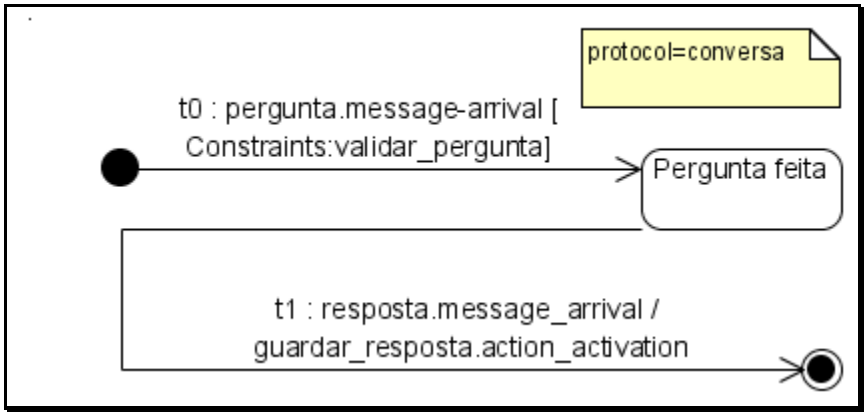


Figura 4-18 – Exemplo de Diagrama de Estados de Protocolo

O mapeamento dos elementos do Diagrama de Estados de Protocolo para código no formato XMLaw é feito seguindo as seguintes regras de transformação:

DEP01. Estado

- O elemento *Estado* é mapeado diretamente para o elemento de código *XMLaw State*, dentro do elemento *XMLaw States*, dentro do *Protocolo* que está associado ao diagrama (na documentação interna do mesmo).

DEP01.a. Atributos Id e *Rótulo* de Estado

- Os atributos *id* (na documentação interna do estado) e *rótulo* são mapeados diretamente para os seus respectivos atributos do elemento de código *XMLaw State*.

DEP01.b. Tipos de Estado

- Se o *Estado* for *Estado Inicial*, seu atributo *type* recebe o valor *"initial"*; se for *Estado de Execução*, seu atributo *type* recebe o valor *"execution"*; se for *Estado Final de Sucesso*, seu atributo *type* recebe o valor *"success"*; e se for *Estado Final de Falha*, seu atributo *type* recebe o valor *"failure"*.

DEP02. Transição

- O *elemento Transição* é mapeado diretamente para elemento de código *XMLaw Transition*, a nível interior do elemento *XMLaw Transitions*, dentro do *Protocolo* que está associado ao diagrama e seus atributos *from* e *to* referenciam respectivamente os *ids* dos *Estados* de saída e de entrada da *Transição*. Seu atributo *id* também é mapeado diretamente para o seu respectivo atributo do elemento de código *XMLaw Transition.*

DEP03. Evento de ativação

- O *tipo de evento* é mapeado para o atributo *event-type* da *Transição*, e o parâmetro do evento (a *referência ao elemento do modelo*) é mapeado para o atributo *ref* da *Transição.*

DEP04. Efeito de ativação/desativação de elemento de lei

- A *referência* ao *elemento do modelo* que será ativado (ou desativado) indica o elemento do modelo onde será inserido o elemento de código *XMLaw* "*Element"*;

- O *tipo de evento* determina se o elemento será inserido dentro de *XMLaw "Ativations"* ou *XMLaw "Deativations"* (no caso de *Norma* ou *Relógio*)*;*

- O atributo *XMLaw ref* de *Element* guardará o *Id* da *Transição*;

- O atributo *XMLaw event-type* de *Element* guardará o valor "*transition-activation*".

DEP05. Condição de proteção

- Se a *Transição* tiver alguma *Norma Ativa*, será criado um elemento *XMLaw ActiveNorms* dentro da *Transição.* Cada referência a uma *Norma* será mapeada para um elemento *XMLaw Norm*, dentro de *XMLaw ActiveNorms*, com atributo *ref* recebendo o *Id* da *Norma* referenciada.

- Se a *Transição* tiver alguma *Restrição*, será criado um elemento *XMLaw Constraints* dentro da *Transição*. Cada *Restrição* será mapeada para um elemento *XMLaw Constraint*, dentro de *XMLaw Constraint*, com atributos mapeados diretamente.

Veja a transformação do exemplo anterior na Figura 4-19.

Código gerado pela transformação:

```
<Protocol id="conversa">
    <Messages>
        ...
    </Messages>

    <States>
        <State id="si" label="" type="initial"/>
        <State id="s_pergunta_feita" label="Pergunta feita"
                type="execution"/>
        <State id="sf" label="" type="success"/>
    </States>

    <Transitions>
        <Transition id="t0" ref="pergunta" event-type="message_arrival"
                    from="si" to="s_pergunta_feita">
            <Constraints>
                <Constraint id="validar_pergunta" class="……"/>
            </Constraints>
        </Transition>
        <Transition id="t1" ref="resposta" event-type="message_arrival"
                    from="s_pergunta_feita" to="sf"/>
    </Transitions>
</Protocol>

<Action id="guardar_resposta" class="……">
    <Element event-type="transition-activation" ref="t1"/>
</Action>
```

Figura 4-19 - Exemplo de código gerado pela transformação de Diagrama de Estados de Protocolo

4.4.2.3. Considerações

Todas as instâncias de elementos de lei que forem modelados nesse diagrama, com exceção do *Estado* e da *Transição,* devem ser especificadas no Diagrama de Instâncias de Elementos de Lei.

4.4.2.4.
Problema do Aeroporto: Comportamento dos Protocolos

A especificação desse diagrama pode e deve ser feita em paralelo à especificação do Diagrama de Instâncias de Elementos de Lei, já que um deve influenciar o outro.

Em UML, especifica-se uma *transição automática* a partir do *Estado Inicial*, isso é, sem *evento de ativação*. Seguindo essa mesma semântica, em LawML cria-se sempre um *Estado Inicial* com a *Transição* automática para um *Estado de Execução* de *Protocolo*.

Cada *Protocolo* desse problema-exemplo possui um *Estado Inicial* que se conecta ao próximo *Estado* do *Protocolo* por meio de uma *Transição automática*. Apesar de essa restrição sempre popular todos os diagramas de *Protocolo* com um estado e uma transição a mais, ela permite seguir os padrões já consolidados pela UML para especificação de máquinas de transição de estados, que considera o *Estado Inicial* como pseudo-estado.

4.4.2.4.1.
O Protocolo da Cena Chegada

O *Estado Inicial* desse *Protocolo* recebe o rótulo *"s-ini-chegada"*. A partir desse *estado* deve sair uma *transição automática* "*t-ini-chegada*" conectada ao próximo *estado* do *Protocolo.*

O *estado de execução* que recebe uma *transição* (*t-ini-chegada*) do *Estado Inicial* (*s-ini-chegada*) será rotulado por "Entrada de Usuário" (*id*= "s0"). Quando o agente *usuário* enviar para o agente *anunciante* a mensagem *m1*, avisando que chegou ao aeroporto, a *transição t1* é ativada, e o *Protocolo* passa para o *estado* de rótulo "Olá enviado", de *execução* (*id*= "s1"). Um *relógio* é ativado toda vez que um agente envia uma mensagem requisitando os serviços. A finalidade desse relógio é manter um bom tempo de resposta no sistema, pois uma vez que o relógio contar cinco segundos, uma ação de recuperação automática do sistema (*anunciante-fora-do-ar*) é ativada.

A ativação da transição *t1* tem como efeito a *ativação* do *relógio* de id="*tempo-para-responder-olá*", que deverá ser criado no Diagrama de Instâncias de Elementos de Lei da *Cena* Chegada. A *ativação* da *ação* de id= "*anunciante-*

fora-do-ar" não é especificada nesse diagrama, e sim, no Diagrama de Comunicação de Instâncias de Elemento da *Cena* Chegada, como será visto na Seção 4.4.3.4.1.

Com a resposta do agente *anunciante* através da lista de serviços disponíveis no aeroporto (mensagem *m2*), a *transição t2* é ativada, e sua ativação tem como efeito a *desativação* do *relógio* "*tempo-para-responder-olá*" e o *Protocolo* passa para o *estado* de rótulo "*Olá respondido*" *(estado final de sucesso,* de *id = s2).* Veja a Figura 4-20.

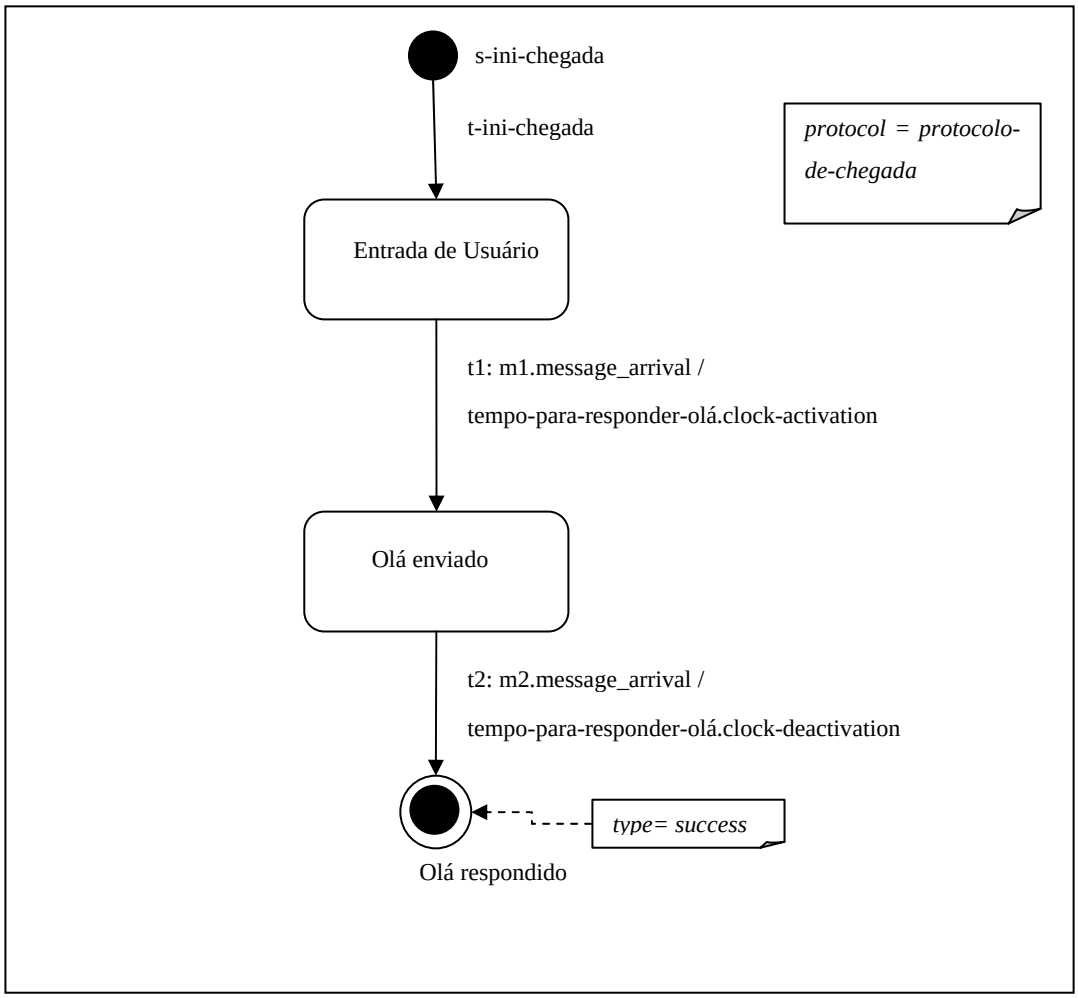


Figura 4-20 - Diagrama de Estados de Protocolo de *protocolo-de-Chegada*

Após a especificação desse *Protocolo*, pode-se voltar ao Diagrama de Instâncias de Elementos de Lei da Cena Chegada (Figura 4-21) e especificar o *Relógio* "*tempo-para-responder-olá"* (e seus atributos) com um *vínculo* com a *Cena*, a *Ação* de id= "*anunciante-fora-do-ar*" (e seu atributo) com um *vínculo* com a *Cena* e com o *Relógio*. Esse *vínculo*, de estereótipo <<*ativação*>>, deve

ter nome nos papéis[6] de *ativador* (na extremidade do *Relógio)* e de *ativado* (na extremidade da *Ação).*

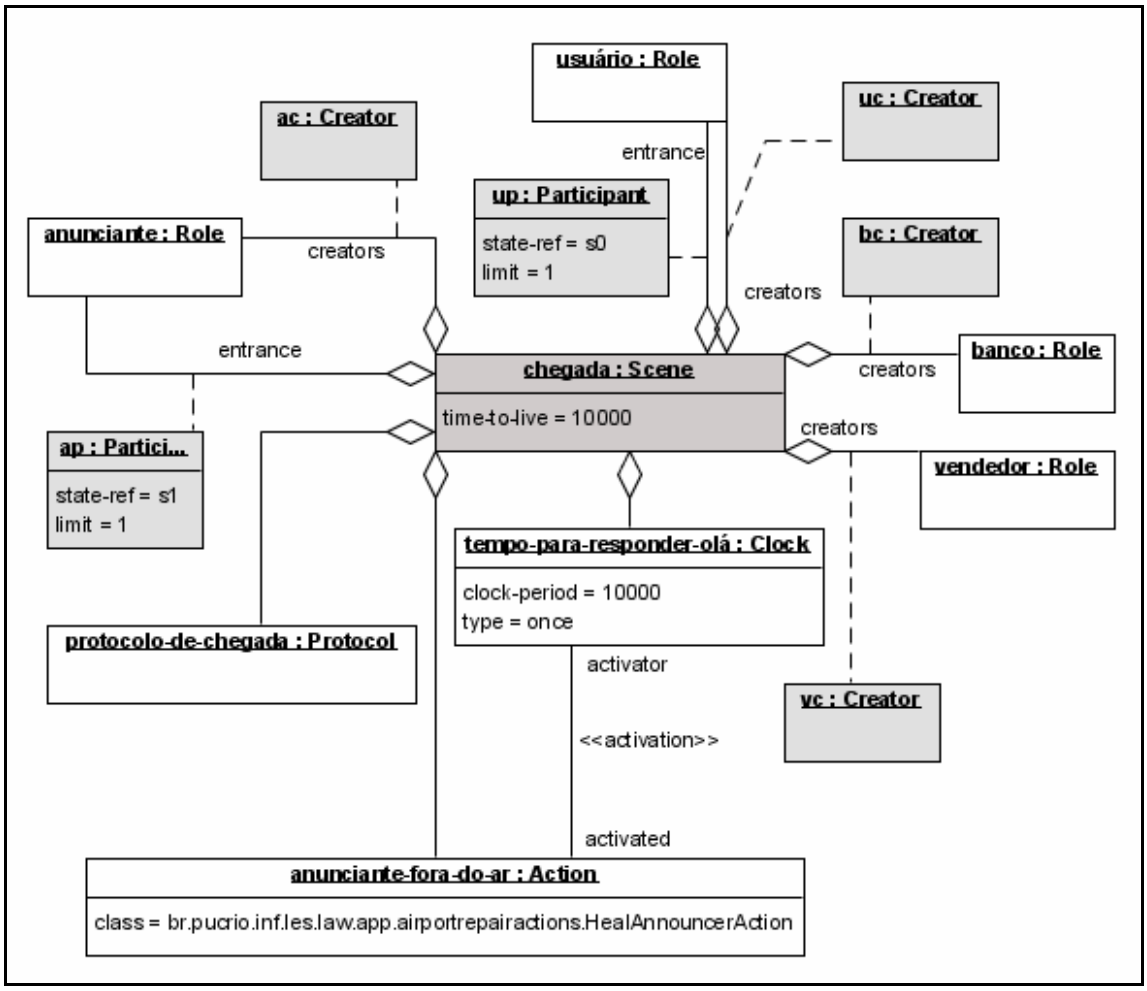


Figura 4-21 - Diagrama de Instâncias de Elementos de Lei da Cena Chegada

E, por fim, o *Estado* em que cada *Participante* pode entrar na *Cena*. Eles só podem participar se o estado de execução do *Protocolo* estiver no *estado s0* (rotulado por "Entrada de Usuário") para a entrada do *usuário* e no *estado s1* (de rótulo "*Olá enviado*") para a entrada do *anunciante*.

[6] Não confundir com o elemento de lei *Papel*. Trata-se do papel que cada instância exerce no vínculo, como em UML.

4.4.2.4.2. O Protocolo da Cena Seleção

O *Estado Inicial* desse *Protocolo* recebe o rótulo "*s-ini-seleção"*. A partir desse *estado* deve sair uma *transição automática* conectada ao próximo *estado* do *Protocolo*.

O *estado de execução* que recebe uma *transição* do *Estado Inicial* (*s-ini-seleção*) será identificado por *s3, de rótulo "Pronto para solicitar opções"*. Quando o agente *usuário* enviar para o agente *anunciante* a mensagem *m3,* informando qual serviço foi escolhido, a *transição t3* é ativada, e o *Protocolo* passa para o *estado s4*, de *rótulo* "*Lista de produtos solicitada*".

A resposta do anunciante contém a lista de produtos disponíveis para o serviço de compra requisitado (mensagem m4), a *transição t4* é ativada, e o *Protocolo* passa para o *estado s5*, de *rótulo* "*Lista de produtos informada*".

O usuário escolhe um dos produtos e requisita a lista de vendedores do produto (mensagem m5), a *transição t5* é ativada, e o *Protocolo* passa para o *estado s6*, de *rótulo* "*Lista de vendedores solicitada*".

Finalmente, a lista de vendedores é informada (mensagem m6), o *Protocolo* alcança o *estado final de sucesso* s7, de *rótulo "Lista de vendedores informada"*, o *Protocolo* e a *Cena* são encerrados. Veja a Figura 4-22.

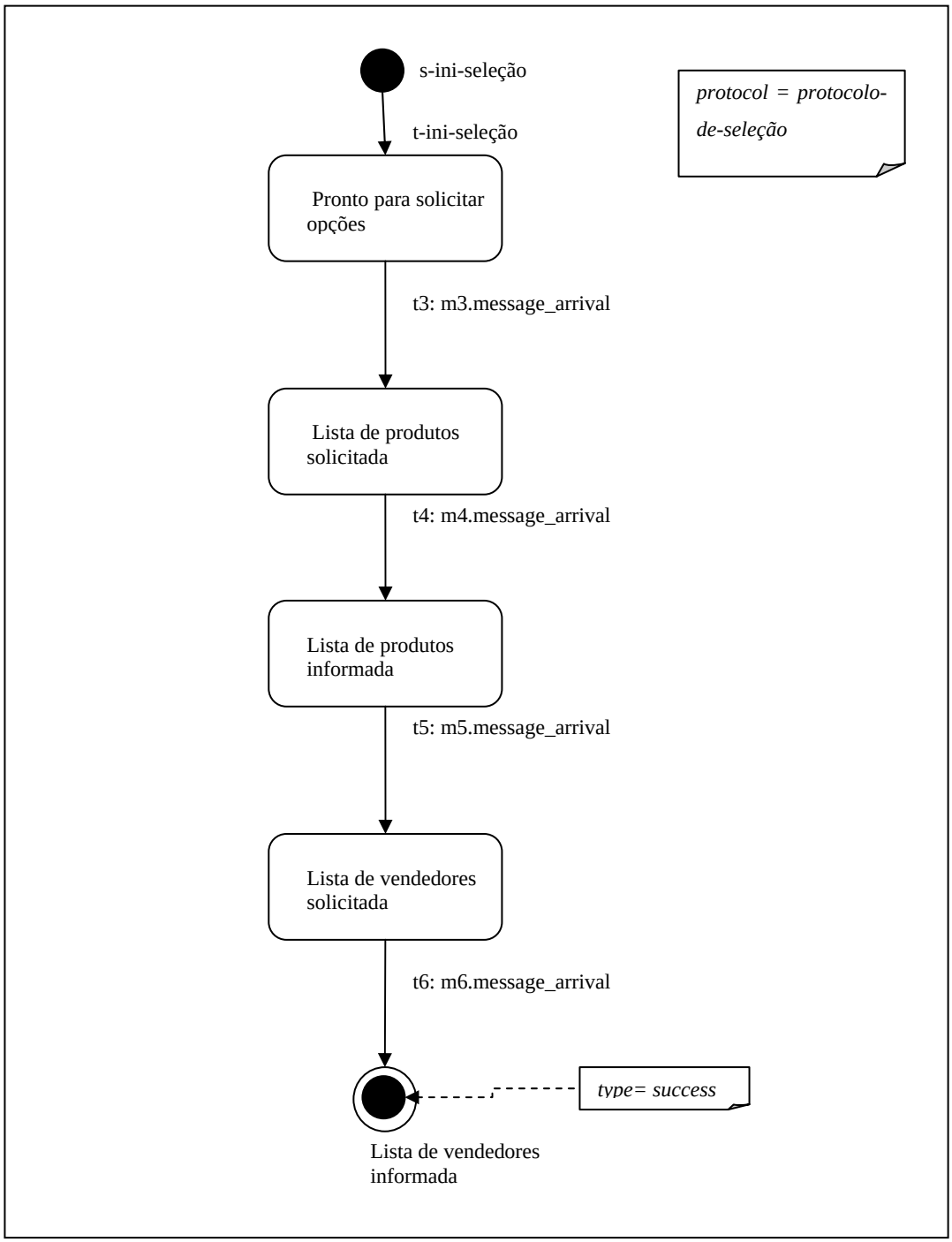


Figura 4-22 - Diagrama de Estados de Protocolo de *protocolo-de-seleção*

Um agente “usuário” é responsável por iniciar a conversação e, portanto, somente pode entrar na cena se o *Protocolo* estiver no *estado s3 - Pronto para solicitar opções.*

Um agente “anunciante” só pode entrar na cena em resposta a conversação iniciada por um agente “usuário”, ou seja, no *estado s4 - Lista de produtos solicitada*.

Após a especificação desse Protocolo, pode-se voltar ao Diagrama de Instâncias de Elementos de Lei da Cena Seleção e especificar o *Estado* em que cada *participante* pode entrar na *Cena*. O Diagrama de Instâncias de Elementos de Lei dessa cena será omitido neste texto, pois a modelagem é análoga à da *Cena* "*Chegada*".

4.4.2.4.3. O Protocolo da Cena Negociação

O *Estado Inicial* desse Protocolo recebe o rótulo *"s-ini-negociação"*. A partir desse *estado* deve sair uma *transição automática* conectada ao próximo *estado* do *Protocolo*.

O *estado de execução* que recebe uma *transição* (*t-ini-negociação*) do *Estado Inicial* (*s-ini-negociação*) será identificado por *s8, de rótulo "Pronto para iniciar negociações"*.

Quando o agente *usuário* enviar para o agente *vendedor* a mensagem *m7*, requisitando uma proposta para o produto selecionado, a *transição t7* será ativada, e o *Protocolo* passará para o *estado s9*, de *rótulo* "*Chamada para proposta solicitada*". Além disso, o *usuário* também fará algumas exigências sobre o preço máximo que ele pagará pelo produto e sobre a marca do produto desejada**.** O *vendedor* só pode enviar propostas com valores menores ou iguais ao preço máximo informado pelo usuário. Isso é implementado nas leis através de uma classe *EnforceValue*, que é chamada pela *restrição EnforceValue.* Essa *restrição* sempre permite a ativação da *transição t7*, mas ao ser executada, o valor do preço máximo é capturado e na próxima *transição* o valor informado será verificado.

A resposta do *vendedor* contém uma proposta de compra (mensagem *m8*). Estando sujeita à *restrição EnforceValue* (verifica se a proposta está dentro do preço máximo informado pelo *usuário*), a *transição t8* é ativada e o *Protocolo* passa para o *estado s10*, de *rótulo* "*Proposta enviada*".

Ao receber a proposta de compra, o agente *usuário* possui vinte segundos para decidir se vai aceitar ou não a proposta. Isso permite um prazo de validade para a proposta e garante ao vendedor a opção de cancelar a mesma. A *transição t8* tem como efeito a *ativação* do *relógio* de *id* "*tempo-para-decidir*". Caso os vinte segundos expirem, o *vendedor* ganha *permissão* para cancelar a proposta

feita. Essa regra deve ser especificada no Diagrama de Comunicação de Instâncias de Elemento de Lei da Cena *Negociação*.

Se em vez de oferecer uma proposta de compra, o *vendedor* decidir negar o pedido, enviando a *mensagem m11*, é ativada a *transição t11*, partindo do *estado s9*, e o *Protocolo* termina no *estado final de falha s13*, de *rótulo* "*Solicitação de proposta recusada*". Veja a Figura 4-23.

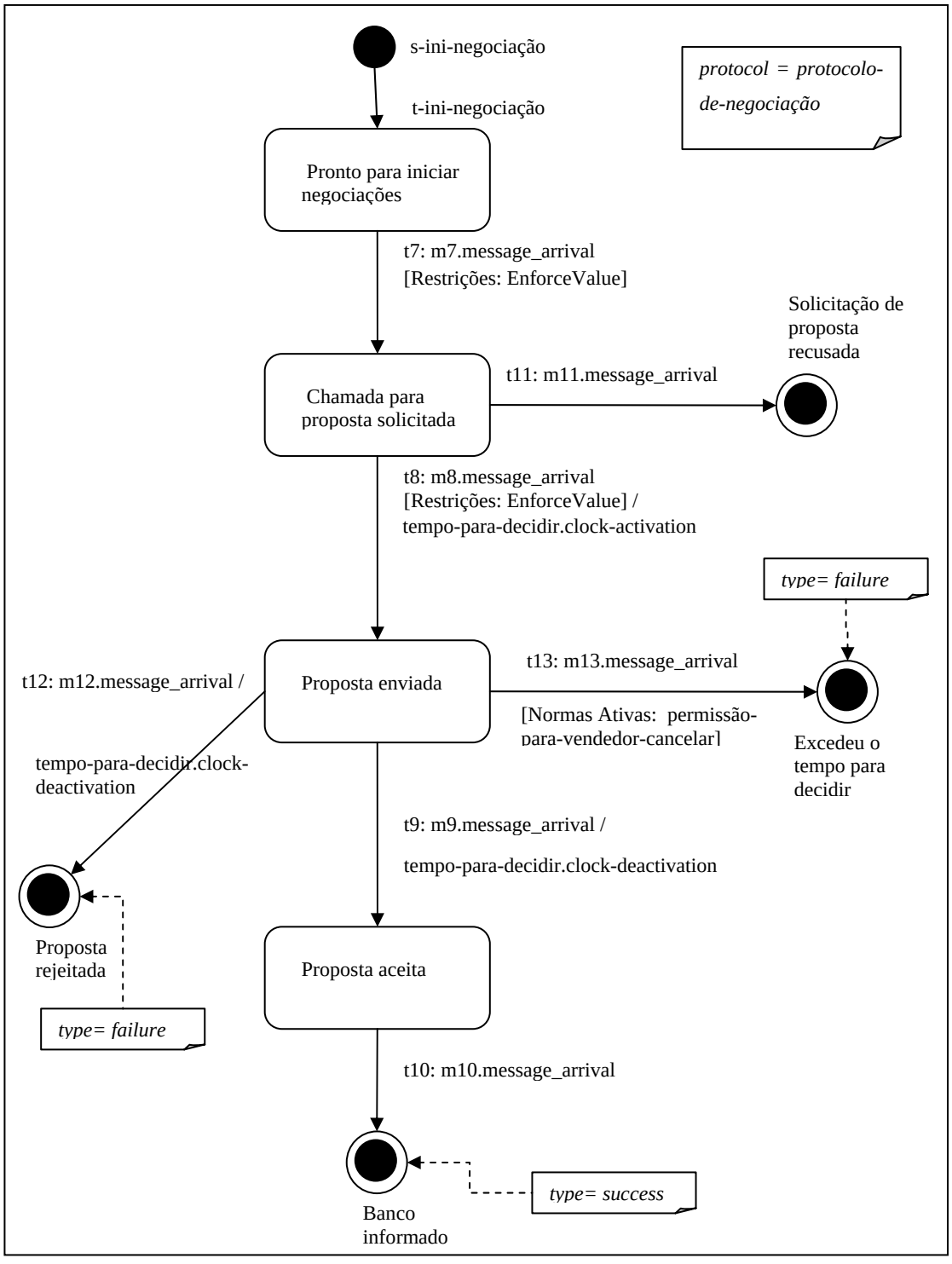


Figura 4-23 - Diagrama de Estados de Protocolo de *protocolo-de-negociação*

Se o *usuário* aceitar a proposta (mensagem *m9*), a *transição t9* será ativada e o *Protocolo* passará do estado *s10* para o *estado s11*, de *rótulo* "*Proposta aceita*". A ativação da *transição t9* tem como efeito a *desativação* do *relógio* de *id* "*tempo-para-decidir*".

Finalmente, o *vendedor* informa onde o pagamento deve ser feito (*mensagem* m10) e o *Protocolo* alcança o *estado final de sucesso* s12, de *rótulo "Banco informado"*, sendo, então, encerrado juntamente com a *Cena*.

Se o *Protocolo* estiver no *estado s10* e o *vendedor* cancelar a proposta (pela *mensagem m13*) após pelo menos 20 segundos de indecisão do *usuário* (a norma de *id* "*permissão-para-vendedor-cancelar*" deve estar ativa), então a *transição t13* será ativada e o *Protocolo* chegará ao *estado final de falha s15*, de *rótulo* "*Excedeu o tempo para decidir*".

Uma vez que se esteja no *estado s10,* caso o *usuário* rejeite proposta enviada pelo *vendedor* (*mensagem m12*), a *transição t12* será ativada, levando o *Protocolo* ao *estado final de falha s14,* de *rótulo* "*Proposta rejeitada*". A ativação da *transição t12* tem como efeito a *desativação* do *relógio* de *id* "*tempo-para-decidir*".

Um agente "*usuário*" é responsável por iniciar a conversação e, portanto, somente pode entrar na cena se o *Protocolo* estiver no *estado s8 "Pronto para iniciar negociações"*.

Um agente "*vendedor*" só pode entrar na cena em resposta à conversação iniciada por um agente "usuário", ou seja, no *estado s9* "*Chamada para proposta solicitada*".

Após a especificação desse Protocolo, pode-se voltar ao Diagrama de Instâncias de Elementos de Lei da Cena *negociação* para especificar o *estado* em que cada *participante* pode entrar na *cena*, e para especificar os novos elementos que foram modelados no Diagrama de Estados de Protocolo. São eles: o *relógio* de *id* "*tempo-para-decidir*" (e seus atributos) e a norma de *id* "*permissão-para-vendedor-cancelar*" (e seus atributos), ambos com um *vínculo* com a *Cena negociação*. Alem disso, o *vínculo* entre o *Relógio* e a *Norma*, de estereótipo

<<*ativação*>> com o papel[7] de *ativador* (na extremidade do *Relógio)* é *ativado* (na extremidade da *Norma).* Veja a Figura 4-24.

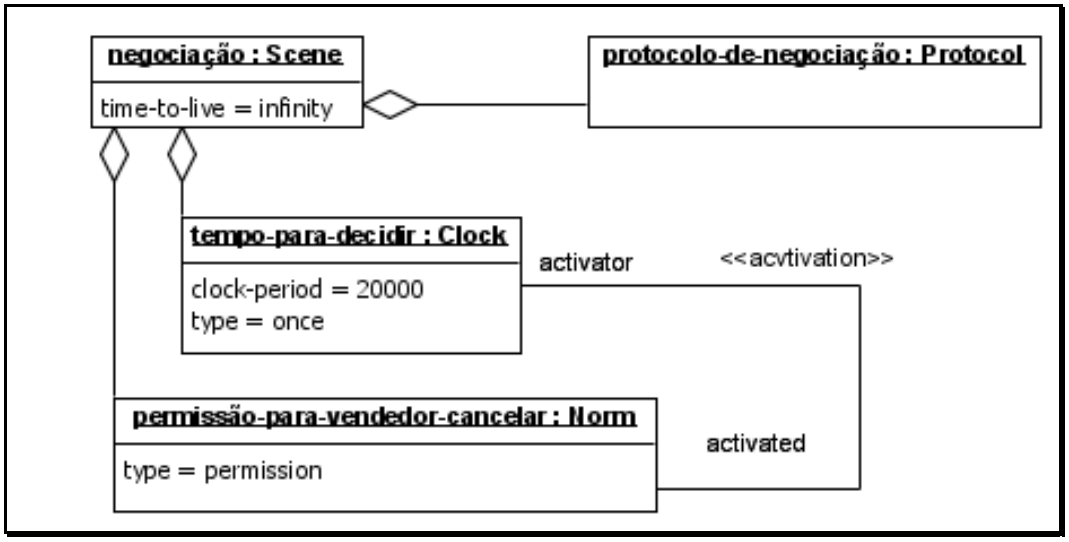


Figura 4-24 - Diagrama de Instâncias de Elementos de Lei da *Cena Negociação* – versão compacta

As instâncias de *Criador* e de *Participante* foram omitidas dessa visualização do Diagrama, pois a modelagem é análoga à da *Cena* "*Chegada*".

O efeito de *ativação* da norma de *id* "*permissão-para-vendedor-cancelar*" deve ser especificado no Diagrama de Comunicação de Instâncias de Elementos de Lei da Cena Negociação. Já o efeito da *ativação* e o efeito da *desativação* do *relógio* de *id* "*tempo-para-decidir*" foram especificados no Diagrama de Estados de Protocolo do *protocolo-de-negociação* e não podem ser especificados em outro diagrama, já que são causados pela ativação de *Transição*.

4.4.2.4.4. O Protocolo da Cena Pagamento

O *Estado Inicial* desse Protocolo recebe o rótulo "*s-ini-pagamento"*. A partir desse *estado* deve sair uma *transição automática* conectada ao próximo *estado* do *Protocolo*.

O *estado de execução* que recebe uma *transição* (*t-ini-pagamento*) do *Estado Inicial* (s-*ini-pagamento*) será identificado por *s16, de rótulo* "*Pronto para pagamento"*. Quando o agente *usuário* enviar para o agente *banco* a

[7] Não confundir com o elemento de lei *Papel*. Trata-se do papel que cada instância exerce no vínculo, como em UML.

mensagem m14, solicitando o serviço de pagamento em nome do *vendedor* do *Protocolo* anterior, o sistema verificará se a *norma* de *id "permissão-para-pagar"* está ativa. Caso esteja, a *transição t14* será ativada e o *Protocolo* passará para o *estado s17*, de *rótulo* "*Ordem de pagamento emitida*". Caso a *norma* não esteja ativa, a *transição* não será ativada e o evento de chegada de mensagem será perdido. Essa norma garante ao *usuário* a *permissão* para realizar o pagamento de sua compra e deve estar especificada no Diagrama de Instâncias de Elementos de Lei.

Como resposta, o *banco* informa ao *usuário* que recebeu o pagamento (*mensagem m15*), a *transição t16* é ativada, e o *Protocolo* passa para o *estado final de sucesso s18*, de *rótulo* "*Recibo enviado*". O *Protocolo* e a *Cena* são encerrados. Veja a Figura 4-25.

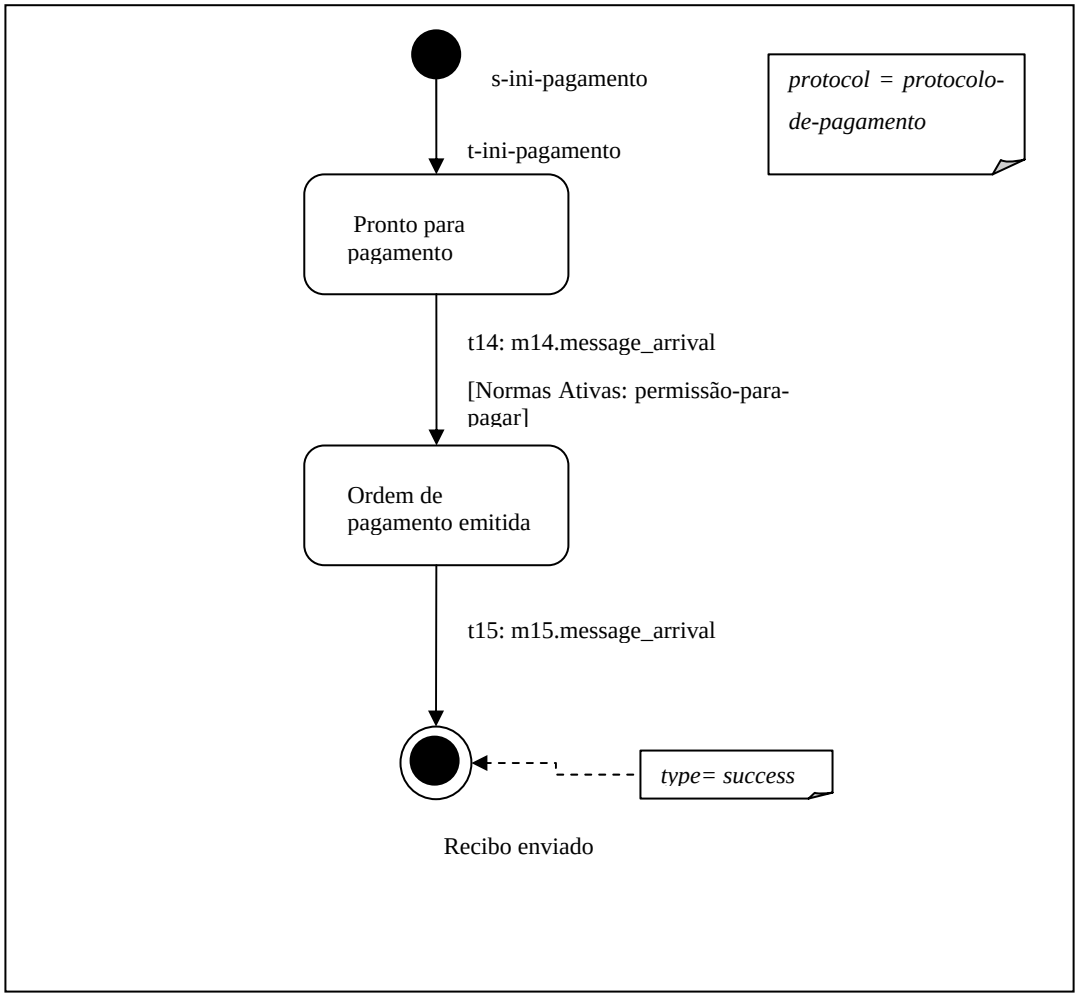


Figura 4-25 - Diagrama de Estados de Protocolo de *protocolo-de-pagamento*

A *norma* de *id "permissão-para-pagar"* deve ser especificada no Diagrama de Instâncias de Elementos de Lei. Esse é um exemplo de *norma* global, que é

usada na *Cena* para a validação da execução do *Protocolo.* Sua ativação e desativação devem ser especificadas no Diagrama de Comunicação de Instâncias de Elementos de Lei. Veja a Figura 4-26.

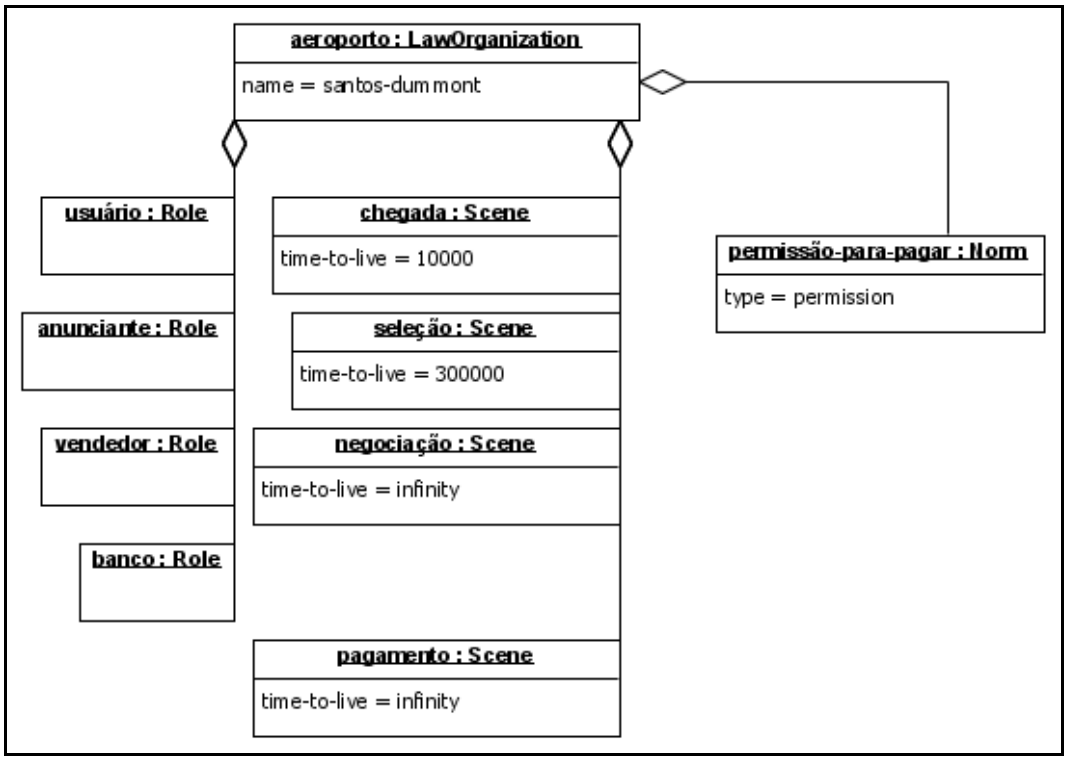


Figura 4-26 - Diagrama de Instâncias de Elementos de Lei

A criação de uma cena de pagamento só pode ser feita por um agente desempenhando o papel de *usuário*. Essa *permissão* é obtida pelos agentes que concluíram a *Cena negociação* com sucesso e é representada como uma *Norma* global, ou seja, uma *Norma* com um *vínculo* com a *Organização de Lei*.

Um agente "*usuário*" é responsável por iniciar a conversação e, portanto, somente pode entrar na cena se o *Protocolo* estiver no *estado s16 "Pronto para pagamento"*.

Um agente "*banco*" só pode entrar na cena em resposta a conversação iniciada por um agente "*usuário*", ou seja, no *estado s17 "Ordem de pagamento emitida"*.

Após a especificação desse Protocolo, pode-se voltar ao Diagrama de Instâncias de Elementos de Lei da Cena Seleção e especificar o *Estado* em que cada *Participante* pode entrar na *Cena*. Estas especificações serão omitidas, pois a modelagem é análoga à da *Cena* "*Chegada*".

4.4.3.
Diagrama de Comunicação de Instâncias de Elementos de Lei

O objetivo desse diagrama é a modelagem comportamental dos elementos dinâmicos que compõem a *Lei.* Esse diagrama permite modelar a ativação e a desativação de instâncias de elementos de lei através da chegada de eventos. Ele se baseia no Diagrama de Comunicação da UML[8], com restrições conforme a abordagem de leis.

Uma instância de elemento de lei é capaz de gerar eventos. Por exemplo, uma instância de *Relógio "Hora de prestar atenção"* gera o evento do tipo "*clock_tick"*. Se o efeito desse evento for a ativação/desativação de um elemento de lei (*Norma*, *Relógio* ou *Ação*), essa ativação/desativação deve ser modelada nesse diagrama (ou no Diagrama de Estados de Protocolo, no caso da ativação da *Transição*). No exemplo anterior, o *Relógio "Hora de prestar atenção"* gera o evento do tipo "*clock_tick"*, que tem como efeito a ativação da instância de *Norma "Permissão para perguntar*". Essa, ao ser ativada, gera o evento "*norm_activation*", que tem como efeito a ativação da instância de *Norma "Permissão para cochilar*". Veja a Figura 4-27.

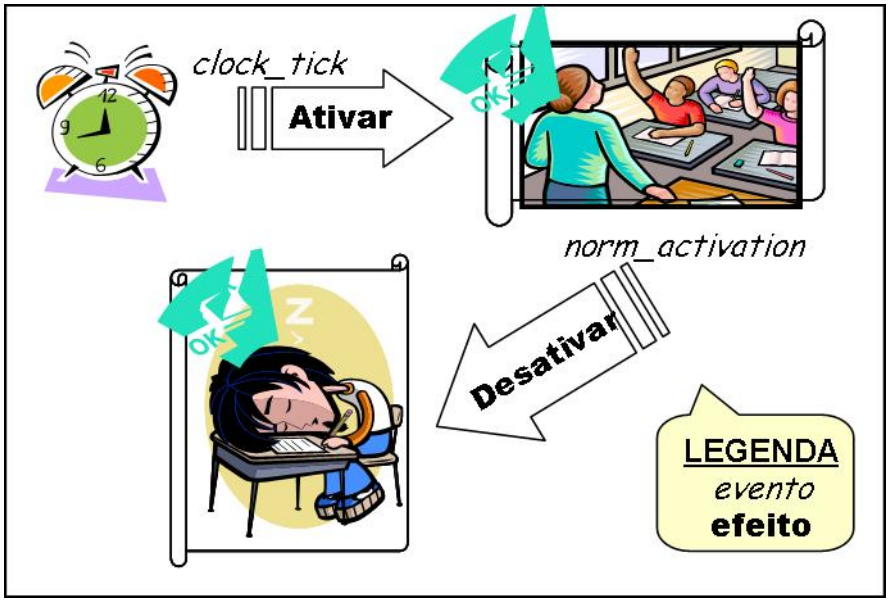


Figura 4-27- Exemplo de Comunicação entre Instâncias

Na Seção 4.4.1. foram apresentados os elementos que são *geradores de eventos*, os elementos que podem ser *ativados* e os que podem ser *desativados*.

[8] Na UML 1.x chama-se Diagrama de Colaboração; a partir da versão 2.0, Diagrama de Comunicação.

Quando ativados/desativados, tais elementos geram eventos de ativação/desativação, que podem ter como efeito a ativação/desativação de outro elemento de lei. Na Tabela 4-1 foram apresentados os tipos dos eventos que cada elemento pode gerar.

A *Transição* possui um tratamento especial. Ela pode ser ativada e seu evento de ativação também pode ter como efeito a ativação/desativação de um elemento de lei. Porém, a ativação da *Transição* e o efeito de ativação/desativação de elemento de lei como conseqüência de seu evento de ativação só podem ser especificados no Diagrama de Estados de Protocolo, para evitar a redundância de informação.

Seguindo essas regras, especificam-se, no Diagrama de Comunicação de Instâncias de Elementos de Lei, as instâncias de elementos de lei *ativadas/desativadas* como objetos, conectando-as com as instâncias *ativadoras*/*desativadoras* e especificando, para o *vínculo*, o estereótipo <<ativação>> ou <<desativação>>, conforme o tipo de ação executada. Para especificar qual *tipo de evento* é o causador da *ativação/desativação*, acrescenta-se uma mensagem[9] no vínculo, contendo o evento, destinada à instância ativada/desativada.

No exemplo anterior, especifica-se a instância de *Norma "Permissão para perguntar"* (ativada) conectada com a instância de *Relógio "Hora de prestar atenção"* (ativadora), através de um *vínculo de ativação*. Acrescenta-se, para o vínculo, a mensagem (evento do tipo "*clock_tick*") que foi gerada pela instância de *Relógio* ativadora. Na Figura 4-28 é possível notar que a *Norma "Permissão para perguntar*" ao ser ativada, desativa a *Norma "Permissão para cochilar"*. Note também que cada *vínculo* possui *um* estereótipo.

[9] Não confundir com o elemento de lei *Mensagem*. Trata-se do envio da mensagem de evento, como em UML.

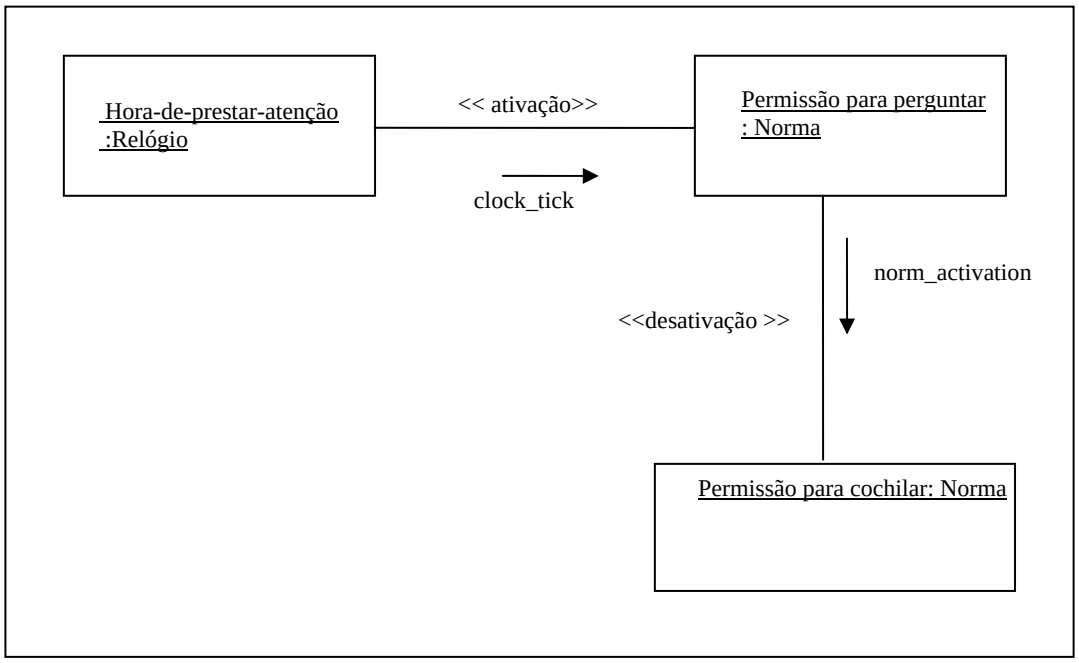


Figura 4-28 – Exemplo de DCIEL

A linguagem LawML possui algumas restrições que a diferenciam da UML. Cada vínculo nesse diagrama deve conter uma e somente uma mensagem[7], que necessariamente deve ser um *tipo de evento*. Se forem necessárias mais mensagens, devem ser criados novos vínculos. Vínculos não permitem restrições ou outros adornos, mas devem conter o estereótipo <<ativação>> ou <<desativação >>. Números de seqüência são opcionais, já que não são mapeados para código XMLaw.

Obrigatoriamente, todos os nós desse diagrama devem ser instâncias de elementos de lei identificadas pelo seu *id* e pelo seu tipo (qual elemento de lei está sendo instanciado). Não é permitido especificar os estados das instâncias, como em UML.

4.4.3.1. Sintaxe

Sintaticamente, esse diagrama se assemelha com o Diagrama de Comunicação da UML. Cada instância de *Elemento de lei* é representada por um retângulo, de maneira análoga ao Diagrama de Instâncias de Elementos de Lei. Contudo, não é necessário utilizar outros compartimentos, já que o foco desse diagrama não é de detalhamento das instâncias.

Cada vínculo é representado por uma linha contínua conectando os objetos (instâncias de elementos de lei). Caso ele seja de *ativação*, deve ser colocado um rótulo com o estereótipo <<ativação>>, e se for de *desativação*, o estereótipo <<desativação>>.

Um *evento* é representado como uma mensagem – uma seta com um rótulo indicando o *tipo de evento*. A instância fonte da mensagem representa o *elemento ativador* (que gera o evento) e a instância destino representa o *elemento ativado/desativado*. Veja a Figura 4-29.

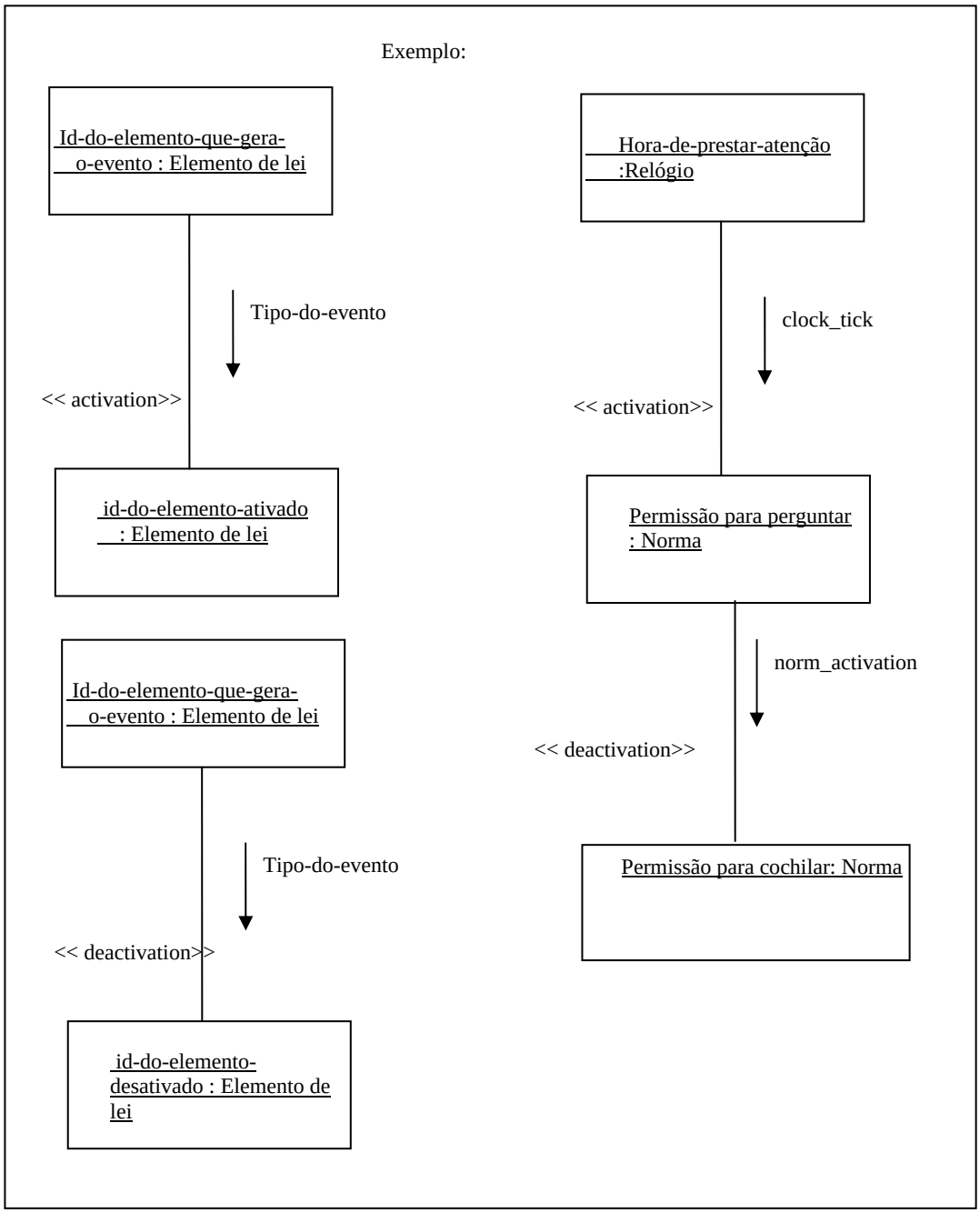


Figura 4-29 - Notação do Diagrama de Comunicação de Instâncias de Elementos de Lei

4.4.3.2. Regras de Transformação

O mapeamento dos elementos desse diagrama para código no formato XMLaw é feito através das seguintes regras de transformação:

DCIEL01. Vínculo

- Cada vínculo será mapeado para um elemento de código *XMLaw Element*,que deve estar dentro de um elemento *XMLaw Activation* (ou *Deactivation*), de acordo com o tipo de vínculo; seja ele de *Ativação* (ou *Desativação*) de *Norma* ou *Relógio*. Esse elemento *XMLaw Activation* (ou *Deactivation*), por sua vez, deve estar dentro do elemento do modelo que estiver referenciado no *destino* da conexão;
- No caso da *ativação* de *Ação* o elemento de código *XMLaw Element* deve estar diretamente dentro do elemento de código *XMLaw Action* relacionado no *destino* da conexão;
- O atributo *XMLaw ref* de *Element* será dado pelo *Id* do elemento *gerador do evento* referenciado na fonte do vínculo;
- O atributo *XMLaw event-type* de *Element* será dado pelo valor do *tipo de evento* do vínculo, ou seja, o rótulo da mensagem[7].

DCIEL02. Instância de elemento de lei.

- Cada instância deve ser especificada no Diagrama de Instâncias de Elementos de Lei ou no Diagrama de Estados de Protocolo, no caso do *Estado*. Portanto, nesse diagrama nenhuma instância deve ser mapeada para um elemento de código XMLaw.

Na Figura 4-30 é apresentado o código no formato XMLaw gerado pela transformação do exemplo da Figura 4-28.

Código gerado pela transformação:

```
<Norm id="Permissão para perguntar"... >
    <Activation >
        <Element ref="Hora-de-prestar-atenção"
                 event-type="clock_tick" />
    </Activation >
        ...
</Norm>

<Norm id="Permissão para cochilar"... >
        ...
    <Deactivation >
        <Element ref="Permissão para perguntar"
                 event-type="norm_activation" />
    </Deactivation >
</Norm>
```

Figura 4-30 - Exemplo de código gerado pela transformação de DCIEL

4.4.3.3. Considerações

Todas as *instâncias* de elementos de lei e *vínculos* que forem especificadas nesse diagrama devem ser especificadas no Diagrama de Instâncias de Elementos de Lei. As instâncias do elemento *Estado* também devem ser especificadas no Diagrama de Estados de Protocolo.

Vale destacar que nesse diagrama não se recomenda detalhar os elementos de lei. Para isso, deve-se utilizar o Diagrama de Instâncias de Elementos de Lei ou Diagrama de Estados de Protocolo (se for *Estado*).

4.4.3.4. Problema do Aeroporto: Comportamento das Instâncias de Elementos de Lei

A especificação desse diagrama pode ser feita em paralelo à especificação do Diagrama de Instâncias de Elementos de Lei de Cena e do Diagrama de Estados de Protocolo, já que eles estão relacionados.

A *Norma* de *id "permissão-para-pagar"* é ativada quando a *Cena negociação* é finalizada com sucesso, ou seja, quando o *usuário* confirmar que deseja comprar o produto pelo preço acordado. A conclusão com sucesso da *Cena negociação* gera um *evento* de tipo "*sucessful_scene_completion*", que ativa essa *Norma* global. Veja a Figura 4-31. Quando a *Cena pagamento* é finalizada com sucesso, ela gera um *evento* de tipo "*sucessful_scene_completion*" que desativa essa norma global.

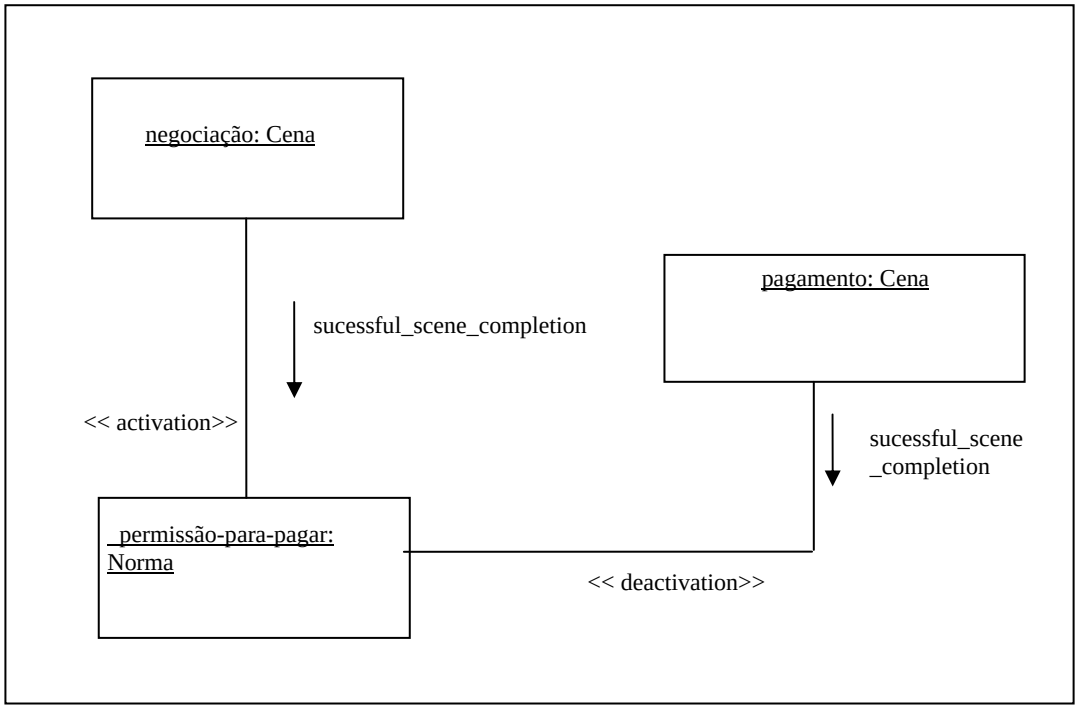


Figura 4-31 - Diagrama de Comunicação de Instâncias de Elementos de Lei do Aeroporto

Após a criação desse diagrama, deve-se voltar ao Diagrama de Instâncias de Elementos de Lei para criar o *vínculo* <<ativação>> entre a *Cena "negociação"* e a *Norma "Permissão-para-pagar"*, e o *vínculo* <<desativação>> entre a *Cena "pagamento"* e a *Norma "Permissão-para-pagar"*. Veja a Figura 4-32.

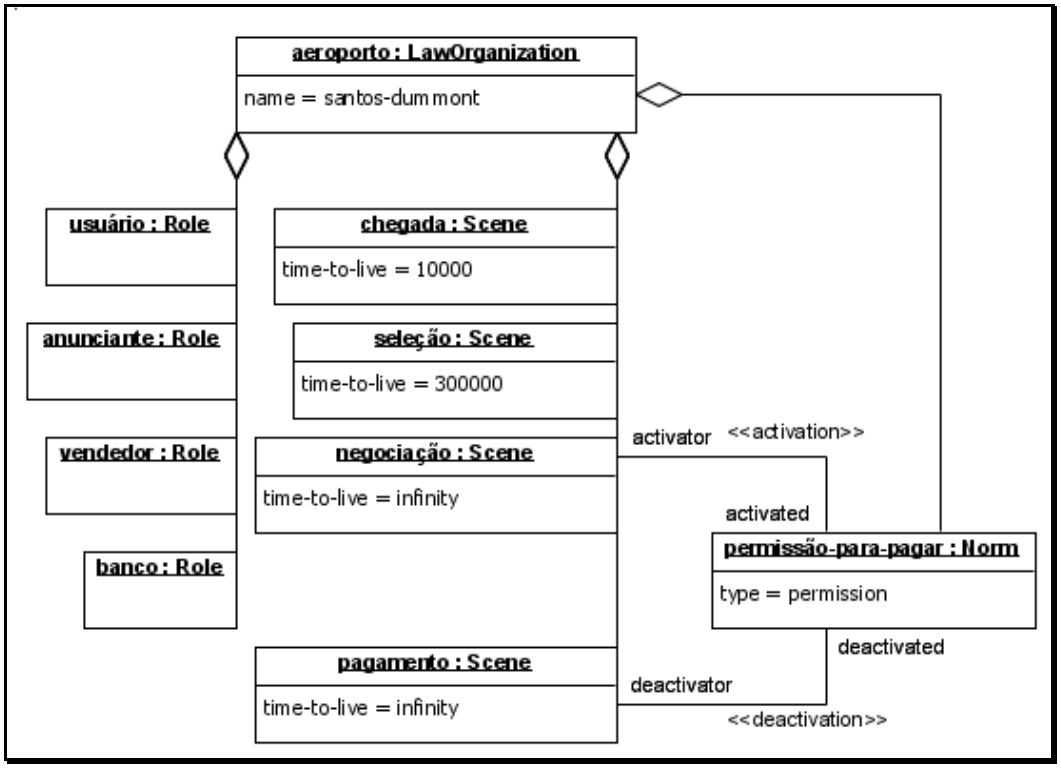


Figura 4-32 - Diagrama de Instâncias de Elementos de Lei do Aeroporto

4.4.3.4.1. A Cena Chegada

Após um agente no papel de *usuário* iniciar a conversação, o agente no papel de *anunciante* terá cinco segundos para enviar uma resposta. O não envio de uma resposta pode significar, por exemplo, que o agente *anunciante* não está funcionando corretamente, ou que existem problemas de comunicação. Nesse caso, uma *Ação* de recuperação deve ser executada para verificar o motivo da falha.

Caso o *relógio* de id="*tempo-para-responder-olá*" (que foi especificado no Diagrama de Instâncias de Elementos de Lei da *Cena* Chegada) dispare o seu alarme (ou seja, gere o evento "*clock_tick*"), uma *Ação* de recuperação automática do sistema, nomeada "*anunciante-fora-do-ar*", deve ser ativada. Veja a Figura 4-33.

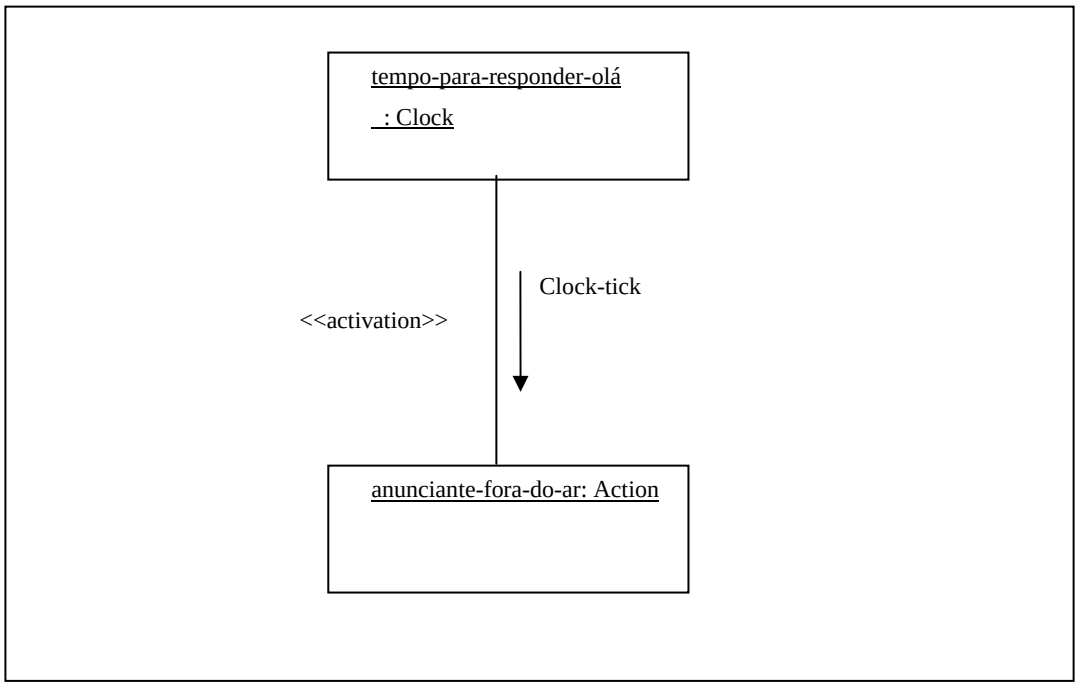


Figura 4-33 - Diagrama de Comunicação de Instâncias de Elementos de Lei da Cena Chegada

Observe que nesse diagrama não foram especificadas a ativação e a desativação do *relógio* de id="*tempo-para-responder-olá*", já que as mesmas foram especificadas no Diagrama de Estados de Protocolo do *Protocolo Chegada* por serem acionadas a partir da *ativação* de *Transições*.

4.4.3.4.2. A Cena Seleção

Essa *cena* não possui Diagrama de Comunicação de Instâncias de Elementos de Lei.

4.4.3.4.3. A Cena Negociação

Os agentes "*usuário*" possuem vinte segundos para decidir se aceitam ou não uma proposta enviada por um "*vendedor*". Depois de decorridos vinte segundos, será dada permissão ao "*vendedor*" para cancelar as negociações com o agente *"usuário"*, ou seja, cancelar a proposta feita. Essa regra é importante, pois protege o vendedor caso outro cliente esteja interessado no mesmo produto e o vendedor esteja impedido de vendê-lo por ter iniciado a negociação com um cliente.

A *ativação* e a *desativação* do *Relógio* (de *id* "*tempo-para-decidir*") já foram especificados no Diagrama de Estados de Protocolo do *Protocolo de negociação* por serem acionadas a partir da *ativação* de *Transições*. Portanto, não é permitido especificá-las nesse diagrama.

Contudo, a *ativação* da *Norma* "*permissão-para-vendedor-cancelar*" deve ser especificada nesse diagrama. Sua *ativação* ocorre quando o elemento *Relógio* de *id* "*tempo-para-decidir*" gerar um evento do tipo "*clock_tick*". Veja a Figura 4-34. Esses dois elementos já foram especificados no Diagrama de Instâncias de Elementos de Lei da Cena Negociação.

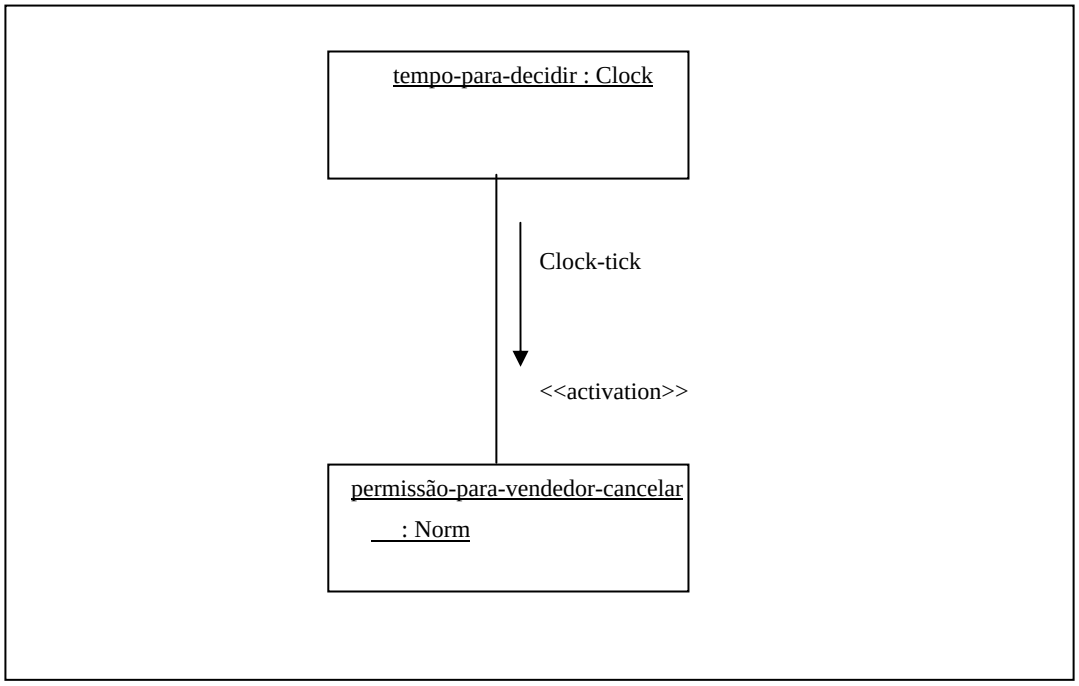


Figura 4-34 - Diagrama de Comunicação de Instâncias de Elementos de Lei da Cena *negociação*

4.4.3.4.4. A Cena Pagamento

Essa *cena* não possui Diagrama de Comunicação de Instâncias de Elementos de Leis, já que ela não possui *Ação, Norma* ou *Relógio*.

4.5. Laudo de Consistência entre Diagramas

Artefatos que podem ser rastreados são essenciais para lidar com mudanças: levam a menos erros durante o desenvolvimento, desempenham um papel importante em situações contratuais e melhoram a aceitação do software (Davis, 1997; Jarke et all, 1999).

Nesta seção será apresentado um breve laudo de consistência entre os diagramas, que serve como base inicial para a geração e validação do rastreamento entre os elementos dos modelos gerados pelo desenvolvedor de leis. Não faz parte deste trabalho a descrição de um método completo de rastreamento entre elementos do modelo.

Para validar os modelos, o desenvolvedor de leis deve seguir às seguintes regras:

1. Toda instância *ativável* (*Ação, Relógio* ou *Norma*) que aparecer no Diagrama de Instâncias de Elementos de Lei deve aparecer sendo ativado ou em um dos Diagramas de Estados de Protocolo ou em um dos Diagramas de Comunicação de Instâncias de Elementos de Lei.

2. Todo *Papel* do Diagrama de Instâncias de Elementos de Lei (que deve possuir um vínculo com a *Organização de Lei)* deve ser especificado como *Participante* em pelo menos uma *Cena*, no Diagrama de Instâncias de Elementos de Lei.

3. Todo *Estado* que aparecer no Diagrama de Instâncias de Elementos de Lei deve aparecer em um (e somente um) dos Diagramas de Estados de Protocolo (como *Estado* de *Protocolo*).

4. Todo *Estado* que aparecer na atribuição de valor ao atributo *States-ref* da instância de *Participante* do Diagrama de Instâncias de Elementos de Lei deve ser definido em um dos Diagramas de Estados de Protocolo (como *Estado* de *Protocolo*).

5. Todo *Estado* que aparecer no Diagrama de Comunicação de Instâncias de Elementos de Lei deve ser definido em um dos Diagramas de Estados de Protocolo (como *Estado* de *Protocolo*).

6. Toda *Cena, Mensagem, Estado, Ação, Relógio* ou *Norma* que aparecer no Diagrama de Comunicação de Instâncias de Elementos de Lei deve aparecer no Diagrama de Instâncias de Elementos de Lei.

7. Toda *Cena, Mensagem, Ação, Relógio* ou *Norma* que aparecer no Diagrama de Estados de Protocolo deve aparecer no Diagrama de Instâncias de Elementos de Lei.

8. Todo *Estado* que aparecer como *gerador de evento de ativação* (de *Transição*) no Diagrama de Estados de Protocolo deve ser definido em um dos Diagramas de Estados de Protocolo (como *Estado* de *Protocolo*).

9. Todo *vínculo* entre *instâncias* que aparecer no Diagrama de Comunicação de Instâncias de Elementos de Lei deve aparecer em um dos Diagramas de Instâncias de Elementos de Lei.

4.6. Considerações sobre LawML

Este capítulo mostrou a LawML, uma linguagem gráfica para modelagem de leis de regulação de interações em Sistemas Multi-Agentes abertos. Para mostrar os impactos da notação proposta, encontram-se, a seguir, algumas das limitações encontradas nos trabalhos relacionados apresentados no capítulo anterior e que foram abordadas pela LawML:

1. *Linguagem Gráfica*: a LawML permite a especificação gráfica das leis de regulação das interações entre os agentes. Com isso, são evitadas as desvantagens da especificação textual das leis, que, conforme o tamanho do sistema, tende a ser muito grande, gerando um trabalho cansativo e custoso.

2. *Linguagem de modelagem baseada na UML*: a LawML se baseia na notação padrão empregada para modelagem de sistemas orientados a objetos, aproveitando a familiaridade da comunidade de desenvolvimento de software com essa notação, o que poderia evitar o estudo de uma nova notação.

3. *Modelagem das leis em visões*: a LawML oferece uma divisão lógica da modelagem das leis, na forma de visões. Essas visões estão relacionadas e têm o objetivo de facilitar a modelagem da estrutura estática e dinâmica da lei.

4. *Linguagem relacionada à abordagem de leis:* a LawML está baseada na abordagem de leis, que se diferencia das outras abordagens por prover tanto um modelo abstrato, baseado em eventos, que capta os principais elementos da lei, quanto um mecanismo de software para a aplicação das leis especificadas.

(a) *Elementos mapeados para código no formato XMLaw*: os elementos da LawML estão mapeados para código, o que permite um desenvolvimento focado nos modelos gráficos.

(b) *Laudo de Consistência*: fornece um passo inicial para um trabalho de verificação de consistência dos modelos gerados.

Vale lembrar que esta linguagem não garante o desenvolvimento do Sistema Multi-Agente completo, já que ela é voltada para as leis que servem para regular o comportamento desejado desse sistema. Cabe ressaltar também que a linguagem LawML não é uma extensão formal da UML.