2
Conceitos

Neste capítulo são apresentados alguns conceitos necessários para o entendimento desta dissertação. Visto que esta proposta está inserida no contexto de sistemas multiagentes abertos, serão apresentados conceitos de sistemas multiagentes governados por leis. Em especial, a abordagem XMLaw proposta para implementação de sistemas abertos é apresentada com algum grau de detalhe, uma vez que o framework apresentado nessa dissertação é destinado a agentes implementados para o middleware M-Law.

2.1
Sistema Multiagentes Abertos e Abordagens de Leis

Avanços na tecnologia de agentes baseiam-se no desenvolvimento de processos, modelos, mecanismos e ferramentas para a construção de sistemas de alta qualidade. Sistema multiagentes são compostos por vários agentes autônomos que interagem para alcançar seus objetivos [35]. Apesar de sua autonomia, agentes devem seguir algumas convenções sociais para que a ordem global do sistema seja mantida. Abordagens mais tradicionais implementam essas convenções sociais, ou leis, diretamente no código fonte do agente. No entanto, em sistemas multiagentes abertos, onde agentes podem entrar e sair de acordo com sua vontade e existe pouco controle sobre o comportamento dos agentes [2], essa abordagem torna-se impraticável. Abordagens de leis [22][7][11][8], separam as leis do sistema do código fonte, tornando-as explicitas para que o desenvolvimento e manutenção fiquem mais eficientes.

Podemos então considerar sistemas abertos como uma composição de diversos subsistemas onde regras devem ser obedecidas. Em sistemas multiagentes abertos consideramos que os subsistemas em questão são tratados como agentes de software. Isso significa que ele proverá diversas propriedades que descrevem agentes de software, tais como: autonomia, habilidade social e raciocínio [24].

Nesse tipo de sistema, consideramos que agentes devem se comunicar. Eles também podem cooperar ou competir enquanto interagem. Portanto, uma implementação de um ambiente para sistemas multiagentes abertos deve prover suporte para a interação dos agentes. No entanto, é difícil controlar a interação entre agentes

visto que não possuímos acesso a seu código fonte, por essa razão, abordagens de leis devem separar regras do sistema da implementação do ambiente.

2.2 XMLaw e M-Law

XMLaw é uma linguagem para aplicação de leis e possui um suporte de software chamado M-Law, que implementa um ambiente que descreve um sistema aberto para execução de agentes de software. A implementação do ambiente é representada por dois modelos: estrutural e dinâmico. O primeiro descreve como os elementos das leis estão relacionados enquanto que o segundo representa uma arquitetura baseada em eventos para a aplicação das leis.

A implementação do M-Law foi projetada como um framework e seus pontos de extensão permitem a implantação de infra-estruturas já existentes de agentes. O framework possui um agente mediador que funciona interceptando mensagens trocadas entre os agentes, verificando então a conformidade das mensagens com as leis, redirecionando-as para o destinatário real caso as leis permitam. Caso seja detectada uma mensagem que não esteja conforme às leis, o mediador a bloqueia. A arquitetura é apresentada na figura 2.1. Os agentes podem estar executando no mesmo domínio ou em domínios separados. O mediador também pode executar um qualquer domínio, resumindo, não existem restrições quanto ao local de execução dos agentes.

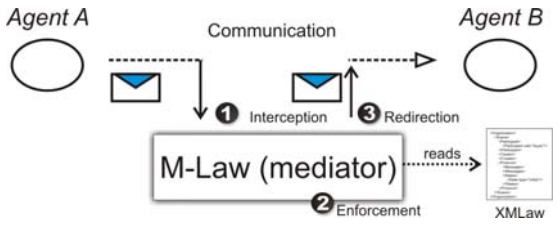

Figura 2.1: Arquitetura

Como dito anteriormente, o framework foi construído para prover suporte a uma linguagem de para especificação de leis, chamada de XMLaw [25]. A linguagem permite ao desenvolvedor do sistema especificar como as interações entre os agentes de todo o sistema ocorrerão. Essa linguagem foi escolhida por expressar idéias consideradas muito importantes para regular interações, tais como normas sociais e restrições temporais. Assim, antes da apresentação do framework, será apresentado brevemente as funcionalidades da linguagem XMLaw, para informações detalhadas verifique [25].

Os principais elementos de XMLaw são: cenas, protocolos, normas, relógios, ações e restrições. Para melhor entender a lógica por trás desses elementos, considere um cenário habitual de compra em um shopping center. Primeiramente um

comprador chega ao shopping center, procura por coisas que pretende comprar, negocia o preço e, finalmente, paga pelo produto. Esse cenário pode ser modelado em quatro cenas distintas: chegada, procura, negociação e pagamento.

O principal propósito de um elemento cena é compor elementos do tipo protocolo, norma, relógio, ações e restrições de forma modular, definindo assim contextos de interação 2.2. Cada cena possui suas peculiaridades, como seu conjunto de normas e seu protocolo de interação. Cenas também restringem a participação dos agentes, especificando quais podem participar e em que momento.

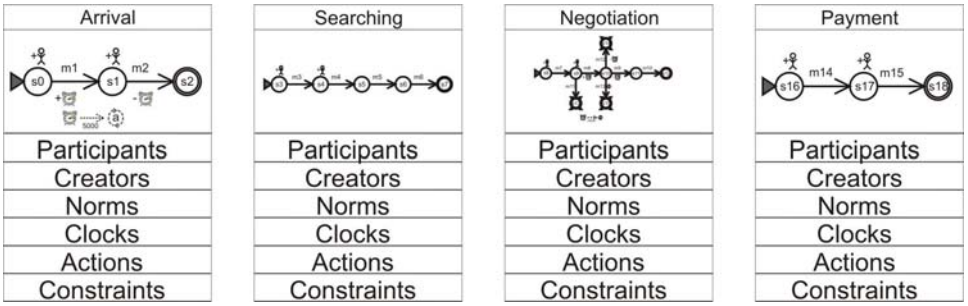

Figura 2.2: Cenas

Protocolos, ou protocolos de interação, são máquinas de estado não determinísticas onde a transição dos estados é disparada pela ocorrência eventos. Eventos formam a base de comunicação entre os elementos de XMLaw que podem gerar ou sentir eventos. Por exemplo, elementos do tipo transição podem sentir por eventos de chegada de mensagem e, uma vez tratado o evento, esse elemento pode gerar um evento de ativação de transição. Esse novo evento pode ser sentido por um elemento do tipo norma, que por sua vez também pode gerar novos eventos, e assim por diante.

Uma vez que XMLaw é apenas uma linguagem de especificação, o controle da geração e propagação dos eventos é feita de fato pelo mediador, no entanto, o desenvolvedor que escrever especificações em XMLaw deve estar atento à geração de eventos para que possa escrever as leis. Por exemplo, a listagem 2.1 em XMLaw mostra como um relógio é ativado quando a transição “t1” ocorrer, em outras palavras, quando o evento “transiction_activation” gerado pela transição “t1” for sentido pelo relógio “c1”.

```
1  ...
2  <Protocol id="contract-net">
3      ...
4      <Transitions>
5          <Transition id="t1" from="s0" to="s1" ref="m1" event-type="message_arrival
               " />
6      </Transitions>
7  </Protocol>
8  <Clock id="c1" type="periodic" tick-period="2000">
9      <Activations>
```

```
10          <Element ref="t1" event-type="transition_activation" />
11      </Activations>
12      ...
13  </Clock>
14  ...
```

Listagem 2.1: Ativação de relógio por transição

Relógios são elementos que permitem a especificação temporal de aspectos da lei. Existem dois tipos de relógio, um que gera apenas um evento após um tempo decorrido e outro que gera eventos continuamente num intervalo específico de tempo.

Normas são vistas como compromissos designados a priori ou assumidos a posteriori por agentes. Por exemplo, quando um comprador que participa de um leilão ganha um lance, ele adquire a obrigação de pagar pelo bem. Normas podem ser definidas como obrigações, permissões ou proibições.

Finalmente, XMLaw provê dois mecanismos de extensão onde código java pode ser utilizado: restrições e ações. Restrições permitem o uso de código java para desempenhar validações complexas em mensagens trocadas pelos agentes. Ações permitem a ativação de código java através de eventos, fornecendo uma maneira efetiva de se estender as funcionalidades básicas de XMLaw.

M-Law oferece suporte a XMLaw e suas possíveis evoluções. Uma visão geral do framework pode ser vista na figura 2.3. Essa figura apresenta os quatro principais módulos do framework, representados pelas caixas cinzas. O módulo *Agent* contém classes para o desenvolvimento de agentes. Esse módulo provê uma série de facilidades para interação com o mediador e outros agentes através de métodos para o envio e recebimento de mensagens, para isso utiliza o módulo *Communication*, responsável pela comunicação. Na verdade, o módulo de comunicação é formado por um conjunto de classes abstratas e interfaces que precisam ser estendidas para que sua funcionalidade seja implementada. A idéia é poder usar frameworks multiagentes existentes que já possuam a comunicação inter agente implementada. A implementação oferecida juntamente com o framework utiliza o JADE Agent Framework [4]. Outras implementações foram feitas utilizando-se *sockets* (nesse caso não foi utilizado nenhum framework existente) e o ASF Framework [10].

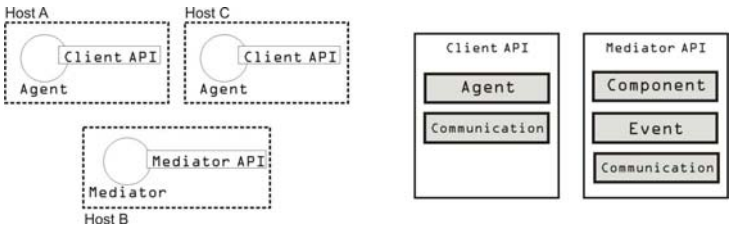

Figura 2.3: Visão Geral

A implementação do agente mediador, que se ocupa de monitorar e verificar a interação dos agentes, é composta pelo módulo *Event* e o módulo *Component*. Esse módulos não são disponibilizados para desenvolvedores de agentes mas foram utilizados para a construção do agente mediador e podem ser estendidos para criação de novas funcionalidades. Um exemplo de extensão do módulo componente para análise de criticalidade pode ser vista em [21].

O módulo *Event* implementa a propagação e notificação de eventos. Basicamente é uma representação do padrão observer [12], permitindo que os elementos recebam e notifiquem eventos.

Elementos como cenas, relógio e normas, são implementados dentro do módulo *Component*. A inserção de novos elementos é feita através da extensão de classes abstratas e interfaces do módulo. Portanto, componentes formam o conjunto de classes utilizadas pelo mediador que implementam o comportamento de elementos da linguagem XMLaw. Por exemplo, na figura 2.4 é apresentado o elemento cena de XMLaw e o conjunto de classes que implementam seu comportamento.

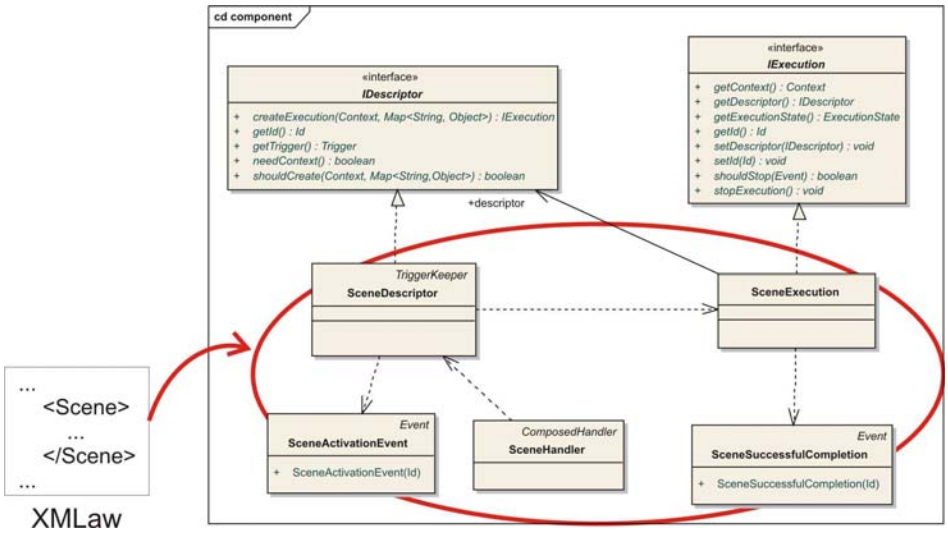

Figura 2.4: Visão Geral

Como exemplo de como o M-Law funciona num cenário, suponha que um agente, desempenhando papel empregado, peça por um aumento de seu próprio salário para o agente que desempenha o papel de contador. No entanto, existe uma norma especificada em XMLaw afirmando que empregados estão proibidos de pedir aumento salarial para eles mesmo. Apesar da simplicidade, o exemplo é bastante útil para ilustrar o fluxo de eventos dentro do M-Law, funcionando portanto da seguinte forma:

– Agente mediador lê as especificações e executa o módulo de componentes;

– Agente empregado invoca seu módulo de comunicação e envia uma mensagem requisitando um aumento salarial para ele mesmo;

– O módulo de comunicação redireciona a mensagem para o agente mediador;

– Mediador recebe a mensagem através de seu módulo de comunicação;

– Mediador dispara um evento de chegada de mensagem através do módulo de eventos;

– Módulo de eventos notifica o módulo de componentes;

– Elemento norma, que faz parte do módulo de componentes, recebe o evento, verifica que a mensagem não esta de acordo com a lei e dispara um evento indicando a não conformidade da mensagem;

– Mediador recebe o evento de não conformidade de mensagem e, como conseqüência, não redireciona a mensagem para o agente contador.

2.2.1 Usando o M-Law

Para que o M-Law seja utilizado são necessários pelo menos quatro tarefas. Primeiramente, é preciso que uma lei seja escrita segundo a linguagem XMLaw. Depois, o mediador deve ser iniciado executando-se os scripts fornecidos juntamente com a implementação do M-Law. Em seguida, o mediador deve ser informado da existência da nova lei para que ela seja publicada e, finalmente, os agentes podem ser executados.

A figura 2.5 detalha as duas principais classes necessárias para o desenvolvimento de um agente.

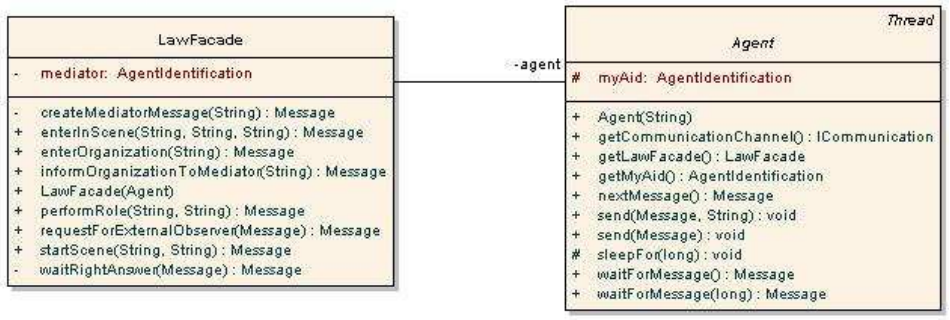

Figura 2.5: Classes do Módulo Cliente

Quanto ao desenvolvimento de agentes para aplicações, os desenvolvedores são recomendados a estender a classe disponibilizada pela API do cliente. Essa classe provê métodos para o envio e recebimento de mensagens, assim como métodos para comunicação direta com o mediador, uma vez que o mediador fornece informações importantes quanto ao estado corrente das interações, como as

cenas que estão executando e quantos agentes existem disponíveis no sistema. Na verdade, a classe *LawFacade* implementa métodos para a comunicação direta com o mediador enquanto que a classe *Agent* fornece acesso ao módulo de comunicação para o envio e recebimento de mensagens.