O *Rationale* da Fidedignidade em Sistemas Multiagentes Abertos Governados por Leis

Maíra Athanázio Gatti[1], Gustavo Carvalho[1], Rodrigo Paes[1], Arndt von Staa[1], Carlos Lucena[1] and Jean-Pierre Briot[2]

[1] Departamento de Informática – PUC-Rio
Rua Marques de São Vicente 225, 4º andar RDC
Rio de Janeiro – RJ – Brasil

{mgatti,guga,rbp,arndt,lucena}@inf.puc-rio.br

[2]Laboratoire d'Informatique de Paris 6
Université Pierre et Marie Curie
Paris – França

Jean-Pierre.Briot@lip6.fr

Resumo. *Os trabalhos desenvolvidos até o momento no contexto da especificação de requisitos para sistemas multiagentes abertos não apresentam claramente o mapeamento e a documentação de requisitos de fidedignidade para infra-estruturas capazes de verificar em tempo de execução se os requisitos são obedecidos. Este artigo descreve uma abordagem chamada Casos de Leis, baseada em Casos de Uso de Segurança, Casos de Fidedignidade e Análise de Risco para especificar sistemas multiagentes abertos governados por leis. Desse modo, o objetivo do trabalho é apresentar a primeira etapa de uma metodologia para análise e projeto das leis que regem um sistema multiagente aberto, e esta será utilizada para integrar o ciclo de vida de desenvolvimento deste tipo de sistema.*

1. Introdução

A pesquisa em tecnologia de agentes de software vem evoluindo por ser considerada um paradigma promissor para a engenharia de sistemas distribuídos complexos. Neste tipo de sistema, nem sempre todos os subsistemas distribuídos que o compõem estão disponíveis a priori, isto é, podem ser associados novos componentes ao longo da execução do sistema. Analisando a abordagem de sistemas multiagentes, considerando que um agente pode ser caracterizado como um subsistema de um sistema distribuído, ele deve poder "entrar" e "sair" sem comprometer o sistema como um todo. Essa categoria de sistema é denominada sistema aberto, ou no caso, sistema multiagente aberto (SMA).

Sistemas multiagentes abertos também são considerados complexos [Jennings 2001], uma vez que o seu comportamento global emerge da interação entre os agentes, através da troca de mensagens, podendo tornar-se imprevisível. Para mitigar os riscos associados ao comportamento emergente que poderiam gerar diversos tipos de falhas, podem ser usadas leis de interação [Minsky 2000]. Estas são regras de interação de um sistema multiagente que especificam a integridade de um sistema aberto através de um conjunto de propriedades que devem ser obedecidas. Uma analogia com assertivas pode ser feita, de forma que é possível afirmar que leis procuram interceptar falhas dinamicamente. De

forma centralizada ou distribuída, o controle da integridade das leis ao executar o sistema é feito por mediadores que interceptam as mensagens ou informações de um ambiente e estes mediadores aplicam as leis ou penalidades previstas e especificadas.

Alguns trabalhos foram propostos como forma de assegurar a governabilidade de sistemas através de leis de interação, dentre eles [Esteva 2003, Dignum 2001, Minsky 2000, Castelfranchi et al 1999, Kollingbaum 2003, Paes et al 2005, Zambonelli 2002]. Este trabalho utiliza a abordagem proposta por [Paes et al 2005] através da linguagem de especificação chamada XMLaw, associada ao *framework* (M-Law) que implementa o mecanismo de regulação das interações dos agentes fazendo verificação em tempo de execução. Por questões de foco e espaço, as vantagens que o XMLaw apresenta em relação às demais abordagens não serão discutidas, e podem ser encontradas em [Paes 2005].

Apesar de existirem diversos trabalhos relevantes para assegurar a governabilidade de sistemas através de leis de interação, os trabalhos desenvolvidos até o momento, no contexto da especificação de requisitos para sistemas multiagentes abertos, não apresentam claramente o mapeamento e a documentação de requisitos de fidedignidade para infra-estruturas capazes de verificar em tempo de execução se os requisitos, ou melhor, se as regras são obedecidas. Fidedignidade é definida como o nível de confiança associada à entrega de um serviço por parte de um sistema. Isto implica que o sistema não apresenta erros, não é suscetível a falhas de segurança, se mantém íntegro durante a sua execução, satisfaz as especificações determinadas, satisfaz os usuários, dentre outras características [Avizienis et al, 2004].

Procura-se neste artigo prover uma maneira eficaz de registrar as decisões (o chamado rastro de *rationale*) para derivar requisitos de fidedignidade, que por sua vez, vão derivar as leis. Mais especificamente, a documentação iterativa de requisitos deve ser feita de forma a preservar o conjunto de decisões que foram tomadas e as razões que determinaram a escolha de requisitos ou de sua implementação. Sempre devemos considerar esta etapa como o primeiro passo de um processo para derivar uma solução com requisitos de fidedignidade plenamente satisfeitos por parte da implementação do sistema a priori especificado.

Desta forma, este trabalho visa a complementar os avanços obtidos na área de governança de sistemas multiagentes abertos com uma forma pragmática de documentação e modelagem baseada em um modelo conceitual adaptado da proposta original de Casos de Fidedignidade [Weinstock et al 2004] apresentados na seção 2. Acredita-se que a aplicação desta estrutura no contexto de sistemas multiagentes consiga representar, em diferentes níveis de detalhe, as decisões e os requisitos derivados a partir do entendimento do problema.

Na próxima seção são apresentados os trabalhos relacionados. Na seção 3 o modelo conceitual, no qual a especificação das leis é mapeada na linguagem XMLaw, é descrito brevemente. Após a apresentação do XMLaw, apresentamos na seção 4 a abordagem preliminar para solucionar esse problema com seu modelo conceitual, o processo de utilização e um exemplo de aplicação para a abordagem proposta. Feito isto, na seção 5 as principais discussões geradas são evidenciadas a partir do exemplo exposto. E por fim, na seção 6 apresenta-se a conclusão deste trabalho e os trabalhos futuros.

2. Trabalhos Relacionados

É possível observar uma baixa qualidade nos sistemas gerados pela falta de atenção aos detalhes nos vários estágios do desenvolvimento de uma solução – dados desconsiderados, suposições não documentadas, ausência de testes – que por sua vez resultam na falha em se identificar e resolver erros.

Durante muitos anos, casos de uso [Booch 1999] tornaram-se uma das mais populares abordagens para analisar e especificar requisitos funcionais. Entretanto, não conseguiram ser eficientes na engenharia de requisitos da qualidade, tais como disponibilidade, performance, confiabilidade, reuso, segurança, portabilidade e usabilidade.

Requisitos de fidedignidade devem ser baseados em serviços que funcionem corretamente independentemente das ameaças que sofram. Neste contexto, existem quatro abordagens relevantes que lidam com requisitos de fidedignidade que são apresentadas nas próximas sub-seções: casos de fidedignidade [Weinstock et al 2004], casos de mau uso (*misuse cases*) [Alexander 2003], casos de uso de segurança (*security use cases*) [Firesmith 2003] e análise de risco para derivar leis [Carvalho 2004].

2.1 Casos de Fidedignidade

Um caso de fidedignidade [Weinstock et al 2004] é uma documentação de evidências que provê um argumento válido e convincente quanto à fidedignidade de um artefato. O argumento baseia-se em uma estrutura que visa levantar evidências e provas de que um dado sistema possui todos os atributos de fidedignidade necessários para uma determinada aplicação. Ou seja, é uma cadeia de raciocínio documentada, baseada em evidências. Quanto melhor for a argumentação e as evidências durante o raciocínio, maior a chance de a hipótese ser verdadeira e, conseqüentemente, mais convincente é o caso de fidedignidade.

No processo de argumentação, é necessário pensar numa estratégia que provê a veracidade da hipótese. E, por sua vez, ao especificar uma estratégia talvez seja necessário especificar sub-hipóteses. Uma vez que todas as sub-hipóteses foram provadas, então se obtém a prova final da hipótese.

Considerando o processo acima, [Weinstock et al 2004] propuseram seis elementos que compõem um caso de fidedignidade: contexto, hipótese, estratégia, suposição, solução e evidência. Como será visto na seção 4, esta proposta será adaptada para um modelo conceitual de rastro do *rationale*. Porém antes, na seção 3, a abordagem de governança de leis utilizada será descrita para um melhor entendimento de quais leis exatamente estaremos especificando através dos Casos de Leis.

Durante o estudo da adaptação dos Casos de Fidedignidade para o mapeamento das leis, foi notada uma dificuldade no uso produtivo desta técnica, pois ela deriva um conjunto relativamente grande de informações sem ao menos definir um ponto de corte ou contexto para facilitar a organização da complexidade. Em conjunto com Casos de Fidediginidade, surgiram Casos de Mau Uso (*Misuse Case*) [Alexander 2003] e Casos de Uso de Segurança (*Security Use Cases*) [Firesmith 2003] como técnicas de elicitação de requisitos eficientes para complementar os Casos de Leis, que são Casos de Fidedignidade adaptados.

2.2 Casos de Mau Uso

A abordagem de casos de mau uso [Alexander 2003] é uma extensão da abordagem de casos de uso. O seu objetivo é modelar e especificar cenários negativos do sistema. Um caso de mau uso é simplesmente um caso de uso do ponto de vista de um ator hostil ao sistema em construção e que não deveriam poder ser realizados no sistema implementado. O ator pode ser tanto um ser humano, como um agente de software (inteligente ou não).

Casos de mau uso se adequam muito bem à análise de requisitos de ameaças de segurança causadas por intrusão (*security*) já que as ameaças do sistema correspondem às ações que um ser humano ou agente de software efetua de forma deliberada para afetar ou destruir o sistema. Novos casos de uso poderiam ser modelados com o objetivo de identificar essas ameaças e mapear em casos de mau uso a fim de atenuá-las.

Além dos requisitos de segurança causados por intrusão, casos de mau uso podem ser adaptados para quaisquer uns dos atributos de fidedignidade, tais como confiabilidade, disponibilidade ou segurança contra acidentes (*safety*). Estes atributos podem ser elicitados e analisados como ameaças causadas por agentes que não são necessariamente inteligentes. Tais agentes poderiam ser: erros humanos, tempestades, erros de projeto (*bugs*), problemas na rede, entre outros. Eles podem causar diversos tipos de falhas no software. Como casos de mau uso enfatizam ameaças e não requisitos de segurança propriamente ditos, casos de uso de segurança (*security use cases*) surgiram como uma forma de estender casos de mau uso com o objetivo de suprir esta deficiência.

2.3 Casos de Uso de Segurança

Casos de uso de segurança [Firesmith 2003] têm por objetivo analisar e especificar requisitos de segurança contra as quais a aplicação deve se proteger considerando um elenco de ameaças previamente identificado. Um caso de uso de segurança é construído a partir de um caso de mau uso. A diferença básica entre eles é que um caso de mau uso será bem sucedido se o seu objetivo for atingido (o que não se deseja para a aplicação). Por outro lado um caso de uso de segurança será bem sucedido se a aplicação evitar que um caso de mau uso seja bem sucedido.

Basicamente, a descrição de um caso de uso de segurança contém a identificação do caso de uso e do caminho que o levou a ser ativado, da ameaça de segurança, das pré-condições, das interações dos usuários, dos usuários de mau uso, do sistema, das ações do sistema e das pós-condições. Sendo que, as interações do sistema, ações do sistema, e as pós-condições devem ser especificadas como requisitos do sistema, e os outros não.

2.4 Análise de Risco para Requisitos de Fidedignidade

Métodos de análise de risco podem ser estruturados para auxiliar na especificação, desenvolvimento, monitoração e manutenção dos requisitos de sistema e com o objetivo de aumentar a fidedignidade. Como visto em [Carvalho 2004], requisitos de fidedignidade podem ser modelados como riscos, que por sua vez, guiam na especificação das leis. Em [Carvalho 2004], é proposto um formulário de cenários para especificar riscos, manter o rastro de causas e conseqüências, descrever quais atributos de fidedignidade as conseqüências são afetados, e especificar a decisão da solução.

Apesar da proposta ser bastante promissora, ela possui duas questões que não são tratadas: não é possível desenvolver e documentar explicitamente o *rationale*, isto é, mecanismo de raciocínio que permita ao analista, a partir do documento de requisitos, entender e registrar as suas decisões quanto a que lei deve ser derivada e, no que diz respeito a avaliar a criticalidade dos agentes de acordo com as leis (e, conseqüentemente, os riscos que derivam as leis), não é possível entender de que forma os riscos e probabilidades associados podem mesurar valores coerentes para cada evento do sistema.

3. Abordagem de Governança de Leis de Interação: o XMLaw

Como previamente explicado, as leis estabelecem uma regulamentação que especifica o comportamento desejado dos agentes. Em SMAs, os aspectos internos dos agentes são inacessíveis, e a única informação disponível sobre os agentes é o comportamento observável por meio das mensagens que eles emitem ao se comunicarem com os outros agentes. O modelo conceitual é baseado em elementos que compõe uma lei.

Dentre os elementos do modelo conceitual do XMLaw, o elemento *Message,* isoladamente, não permite especificar muitas restrições em uma interação. É preciso estabelecer a ordem em que elas ocorrem e quais as mensagens válidas em um determinado momento da interação. Protocolos de interação representam justamente padrões de interação entre os agentes, definindo quais são as interações válidas e inválidas. O elemento *Protocol* representa protocolos de interação entre os agentes e é representado por um autômato finito não determinístico [Menezes 1997], onde estados (elemento *State*) representam pontos na execução do protocolo e transições (elemento *Transition*) são as conexões entre os estados.

O modelo conceitual fornece um elemento denominado Relógio (*Clock*) que provê os meios para especificar leis sensíveis ao tempo. Um relógio é capaz de gerar eventos em intervalos de tempo específicos. Por exemplo, um relógio permite a ativação e desativação de normas após um determinado período de tempo ter se esgotado. Além disso, um relógio pode ser ativado por transições ou mesmo por normas.

Normas (*Norms*) descrevem quais comportamentos dos agentes são permitidos, quais são obrigados e quais são proibidos. As normas são geralmente adquiridas pelos agentes durante o decorrer das interações, e conseguem representar noções de compromissos adquiridos e cumpridos. Por exemplo, em um processo de negociação, um agente pode assumir o compromisso (obrigação) de pagar por uma mercadoria negociada e, enquanto essa obrigação não for cumprida, o agente fica impedido de participar de novas negociações.

O modelo conceitual utiliza a abstração de cenas para auxiliar na organização e modularização das interações, e são representadas pelo elemento do modelo conceitual *Scene*. Este elemento especifica quais agentes e quais os papéis de agentes podem interagir em uma cena, ou mesmo dar início a sua execução. Além disso, uma cena é composta por um protocolo de interação e por um conjunto de normas, ações e relógios. Estes elementos compartilham um contexto comum de interação definido pela cena. Isto significa que uma norma definida no contexto de uma cena é somente visível naquela

cena. Para que um agente inicie ou participe de um protocolo de interação de uma cena, é necessário que ele desempenhe algum papel (*role*) previamente definido na organização e especificado no contexto da cena.

Como mencionado anteriormente, além do modelo conceitual apresentado, existe um *framework*, chamado M-Law, que garante a regulação das interações dos agentes através dos elementos do modelo conceitual e do modelo de interação baseado em eventos. Este mecanismo intercepta as mensagens enviadas por agentes e verifica sua conformidade com as leis. Na próxima subseção, o processo de especificação das leis tal como é feito atualmente de forma *ad hoc* será apresentado para que fique mais clara a descrição da aplicação da nossa proposta nessa abordagem. Além disso, será brevemente descrita na subseção 3.2, a abordagem proposta em [Gatti 2006] e que auxilia na realização de uma análise de criticalidade dos agentes.

3.1. Especificação das Leis

Como visto, para especificar as Leis em XMLaw é necessário descrever elementos de leis como os papéis, normas, *clocks,* cenas, protocolos e mensagens. Cada cena especifica o protocolo de interação que os agentes devem seguir e, conseqüentemente, quais são as mensagens válidas. A partir do entendimento dos riscos e requisitos associados a um SMA, leis podem ser especificadas.

A seguir descrevemos os passos usuais para a especificação de leis:

1. Crie a organização (*LawOrganization*) e estabeleça os papéis (*Role*) de agentes e normas (*Norms*), ações (*Actions*) e *Clocks* que podem existir nela.
2. Crie as possíveis cenas (*Scenes*) nas quais os agentes desempenhando os papéis especificados poderiam interagir dentro da organização.
   - Defina o identificador de cada cena e, caso necessário, o seu tempo máximo de execução.
   - Defina os protocolos com as suas mensagens, transições e estados possíveis de acordo com uma máquina de estados. Caso seja necessário reutilizar um protocolo existente em outra cena (ou até mesmo em outro arquivo), utilize a tag *xi:include* para especificar onde se encontra o protocolo a ser incluído.
   - Defina as normas que regulam a interação e as normas que devem estar ativadas ou desativadas quando do início da execução da cena (opcional).
   - Defina os *clocks*, caso existam para a cena em questão.
3. Associe as normas e os *clocks* para cada mensagem ou transição previamente especificadas no protocolo.

3.2. Especificação da Criticalidade de Agentes

O uso de XMLaw para configurar a política de acompanhamento da evolução da criticalidade dos agentes é descrito nesta seção com o objetivo de exemplificar um cenário de aplicação desta abordagem.

Quando um agente entra em uma organização e começa a interagir com outros agentes, dependendo das interações as quais participa, sua criticalidade pode aumentar ou diminuir. Isto é, o agente pode se tornar mais importante ou menos importante e sua falha pode causar um maior ou menor impacto para o sistema.

Visando aumentar a disponibilidade de SMAs, o XMLaw pode ser usado para especificar a variação da criticalidade do agente de forma que seja possível monitorar os eventos que alteram sua criticalidade [Gatti 2006].

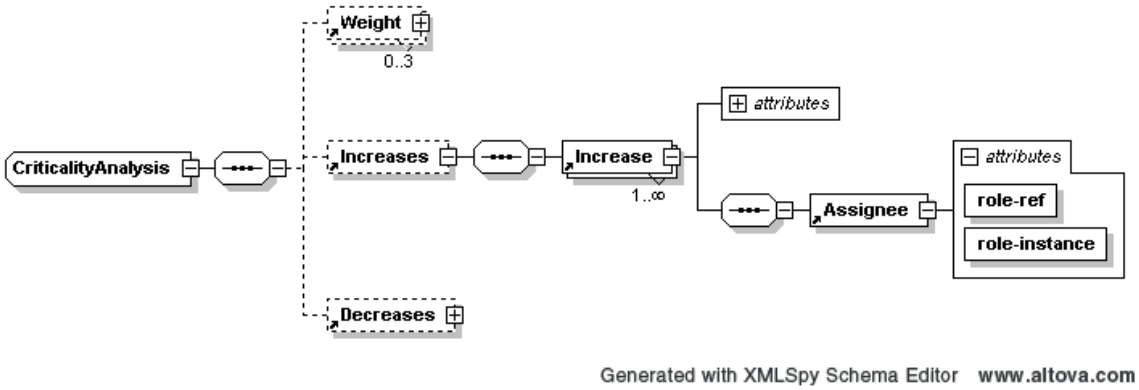

Figura 1. XMLaw: Componente Análise de Criticalidade

Na Figura 1 vemos o XML Schema para o componente de Análise de Criticalidade do XMLaw. O monitoramento descreve os eventos do protocolo que aumentam ou diminuam a criticalidade do agente que está desempenhado um determinado papel na cena. Para isto, é associado um peso para o evento em questão e o sistema recalcula a criticalidade do agente a partir deste peso.

4. Casos de Leis: Uma Abordagem para Apoiar a Derivação de Leis

Através da aplicação de casos de mau uso em diferentes contextos, e da abordagem dos casos de uso de segurança, foi possível mapear com uma certa facilidade, o uso das duas abordagens e da proposta de casos de uso como primeira etapa de um processo de derivação de leis de interação de agentes. Entretanto, notou-se a falta do *rationale* a respeito das interações dos usuários e sistemas, que levassem a garantia de que o caso de uso de segurança fosse adequado para lidar com o problema identificado. Este problema foi endereçado através da adaptação de casos de fidedignidade, já que estes são uma forma de estruturar a documentação de argumentos e evidências de que o sistema atende a requisitos de fidedignidade.

4.1. Caso de Leis

Em um sistema real, com várias necessidades e propriedades que precisam ser satisfeitas, é natural que a quantidade de informações derivadas de um processo de análise cresça com o desenvolvimento da aplicação, isto implicará em um número considerável de hipóteses e argumentos. Para facilitar o entendimento da documentação por parte de um leitor, é proposta, em conjunto com a proposta original de casos de fidedignidade, uma notação gráfica para a representação de elementos.

Pequenas adaptações para a notação proposta foram definidas, em prol de adequá-la ao português e também com o objetivo do uso de uma ferramenta de modelagem própria para a elaboração desta proposta.

O ponto principal da proposta de casos de fidedignidade está na estrutura dos argumentos e das evidências que apóiam e corroboram este argumento. Casos de fidedignidade formais ou informais devem ter como principal objetivo convencer um leitor da sua validade perante as necessidades identificadas. Desta forma, este documento é um elemento chave em nossa abordagem de governança de leis, sendo responsável por registrar todas as hipóteses, suposições, contextos e argumentos adotados no processo de derivação de leis de interação. Todos estes elementos estão estruturados em um modelo conceitual.

O modelo conceitual apresentado na Figura 2 é derivado da aplicação da proposta original de casos de fidedignidade a problemas de governança de leis.

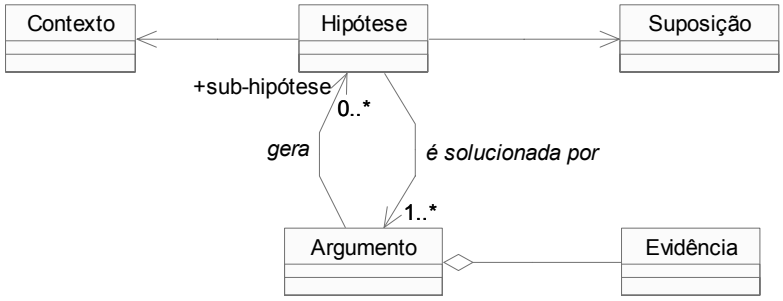

Figura 2. Modelo Conceitual

Analisando a Figura 2, o ponto principal de destaque é a possibilidade de descrever o conhecimento parcial sobre o sistema em desenvolvimento. Isto é, uma hipótese baseia-se em suposições e contextos e é solucionada por um conjunto de argumentos que são justificados por evidências. Suposições não precisam ser provadas, são assumidas como verdade e servem para caracterizarmos melhor o cenário abordado. O contexto nada mais é do que informações que especializam o problema e definem melhor o escopo de análise. Por sua vez, um argumento pode identificar novos contextos que derivam novas hipóteses que necessitam de mais argumentos para o completo entendimento da solução. Por outro lado, a partir de um argumento é possível comprovar a hipótese através de evidências por meio de informações e dados coletados experimentalmente ou ainda por terem passado por um processo de verificação de corretude.

Casos de fidedignidade são derivados a partir de hipóteses sobre um sistema e da demonstração de que evidências que comprovem que as hipóteses são verdadeiras. A Figura 3 descreve o processo de desenvolvimento do *rationale* de um Caso de Leis utilizando o modelo conceitual adaptado. Esta estrutura permite uma maior compreensão e um melhor detalhamento da solução em proposta.

Com o objetivo de exemplificar o *rationale* de um caso de leis, ilustramos na Figura 4 a especificação da sub-hipótese “o agente comprador não pode falhar”. Neste Caso de Leis, que será apresentado na seção 5 com mais detalhes, basicamente, deseja-se impedir que um comprador deixe de efetuar um pagamento em um cenário de negociação de bens.

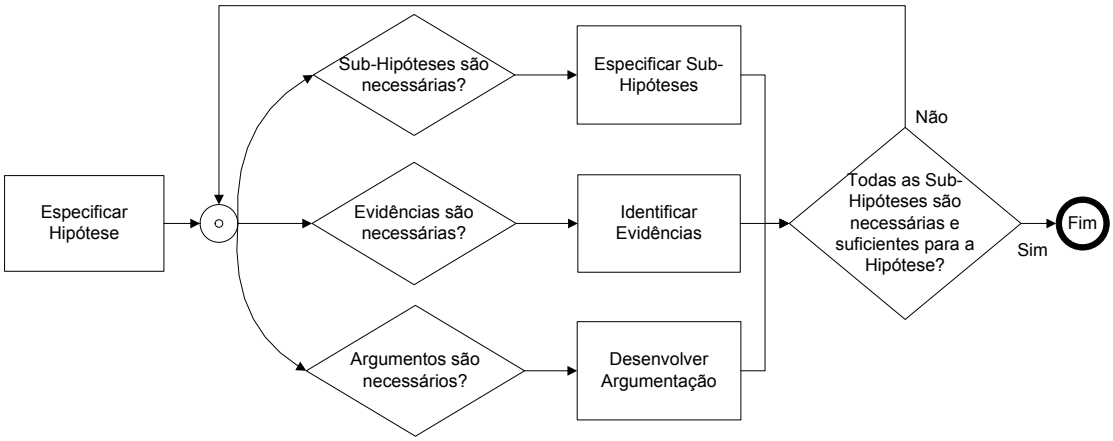

Figura 3. Desenvolvendo o *rationale* de um Caso de Leis

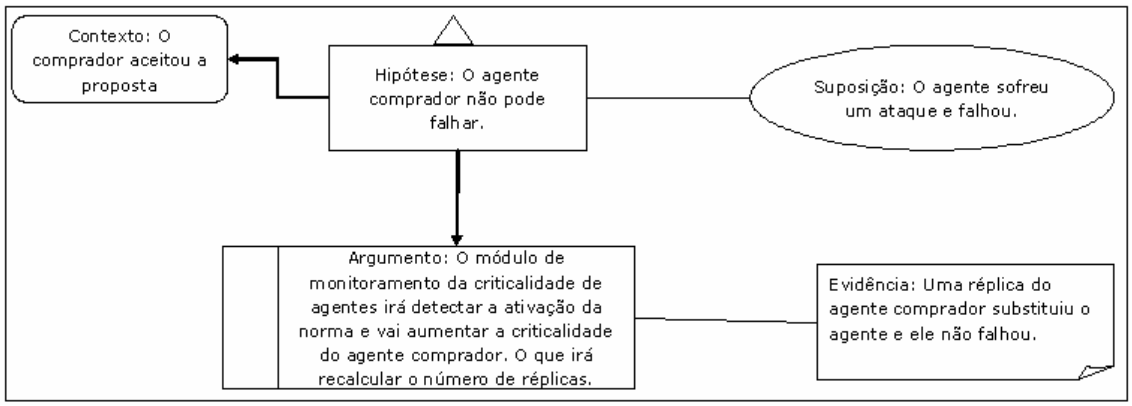

Figura 4. Exemplo da Especificação de uma Sub-Hipótese

4.2. Processo de Análise e Projeto de Leis

Para desenvolver Casos de Leis, é necessário que o Analista de Sistemas desenvolva os Casos de Uso e gere o documento de requisitos funcionais do sistema que, por sua vez, vão gerar os bens e serviços disponíveis pelo sistema. Feito isso, o desenvolvedor vai desenvolver Casos de Mau Uso e os Casos de Leis iterativamente, baseando-se sempre nos requisitos funcionais, não funcionais e ameaças do sistema.

Após ter gerado os Casos de Leis, deve ser feito uma análise dos mesmos para gerar os requisitos de leis que, por sua vez, podem ser facilmente mapeados para a linguagem de especificação XMLaw.

Na Figura 5, vemos de forma abstrata o processo de análise e projeto de leis utilizando casos de leis e as responsabilidades envolvidas. E a seção 5 detalha um caso de lei gerado a partir do processo proposto como forma de exemplificar os elementos do modelo conceitual e o documento e *rationale* gerados.

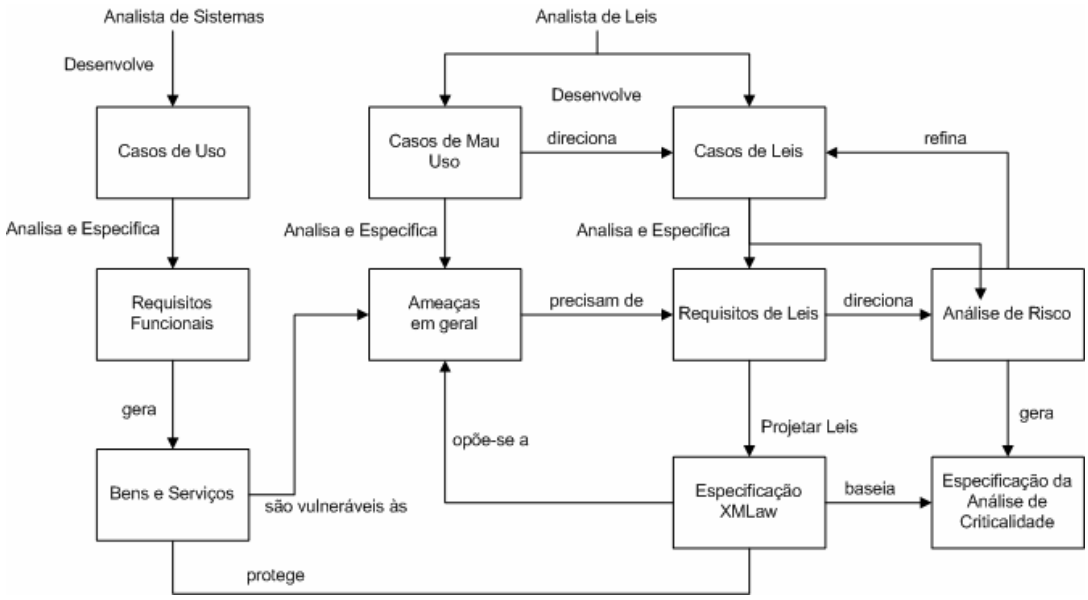

Figura 5. Processo de Análise e Projeto de Leis

Utilizando um processo de análise e projeto de leis de forma iterativa, pode-se refinar não somente as ameaças do sistema, como os requisitos de leis que atenuam tais ameaças. Para esclarecer melhor o que significa desenvolver casos de leis, foi realizado um estudo de caso a fim de avaliar o processo.

Supply Chain Management (SCM) é o planejamento e coordenação das atividades de uma cadeia de suprimentos (Chopra et al., 2004). Essas atividades podem ter vários participantes e organizações, e a coordenação desses participantes é um ponto chave para uma cadeia eficiente. O *Supply Chain* não está relacionado apenas com os produtores e fornecedores, mas também com transportadoras, centros de distribuição, e clientes. Essa cadeia é extremamente dinâmica e envolve um grande fluxo de informação, produtos e recursos financeiros entre diferentes estágios.

O TAC SCM (Collins et al., 2005) é um jogo que foi desenvolvido por pesquisadores do laboratório de *e-Supply Chain Management* da *Carnegie Mellon University* e *Swedish Institute of Computer Science* (SICS). Ele foi modelado para capturar a complexidade de uma cadeia de suprimentos dinâmica, porém com regras simples para que muitas equipes possam participar dessa competição. Basicamente, seis agentes "produtores de PC" participam em cada jogo do TAC SCM. Esses participantes competem por clientes com incerteza na demanda e por peças de um número limitado de fornecedores. Todo dia, os clientes enviam pedidos de cotações e selecionam as melhores ofertas baseadas no preço e na data de entrega. Os agentes são limitados pela capacidade da fábrica, e tem que gerenciar a compra de peças de oito fornecedores. Quatro componentes são precisos para montar um PC: CPU, Placa Mãe, Memória, e Disco Rígido. Cada componente está disponível em diferentes modelos. O jogo começa quando um ou mais agentes se conectam ao servidor do TAC SCM. O servidor simula o comportamento dos fornecedores e clientes, e oferece um ambiente para cada participante com um banco para sua conta corrente, uma fábrica, e estoques de peças e PCs. No final do jogo, o agente que possuir o maior valor de dinheiro na conta corrente é declarado o vencedor.

O jogo apresenta um bom nível de complexidade, pois cada agente deve competir em vários mercados por diferentes componentes do lado dos fornecedores, e em mais outros mercados por diferentes tipos de PCs no lado dos clientes. Além disso, todos esses mercados possuem interdependências e incertezas. Na Figura 6 é ilustrada uma parte do Diagrama de Casos de Leis para o TAC SCM. Note que, para cada caso de mau uso (elipse mais escura), existe pelo menos um caso de leis (elipse com um L dentro) que deriva leis de regulação para que o caso de mau uso não seja bem sucedido.

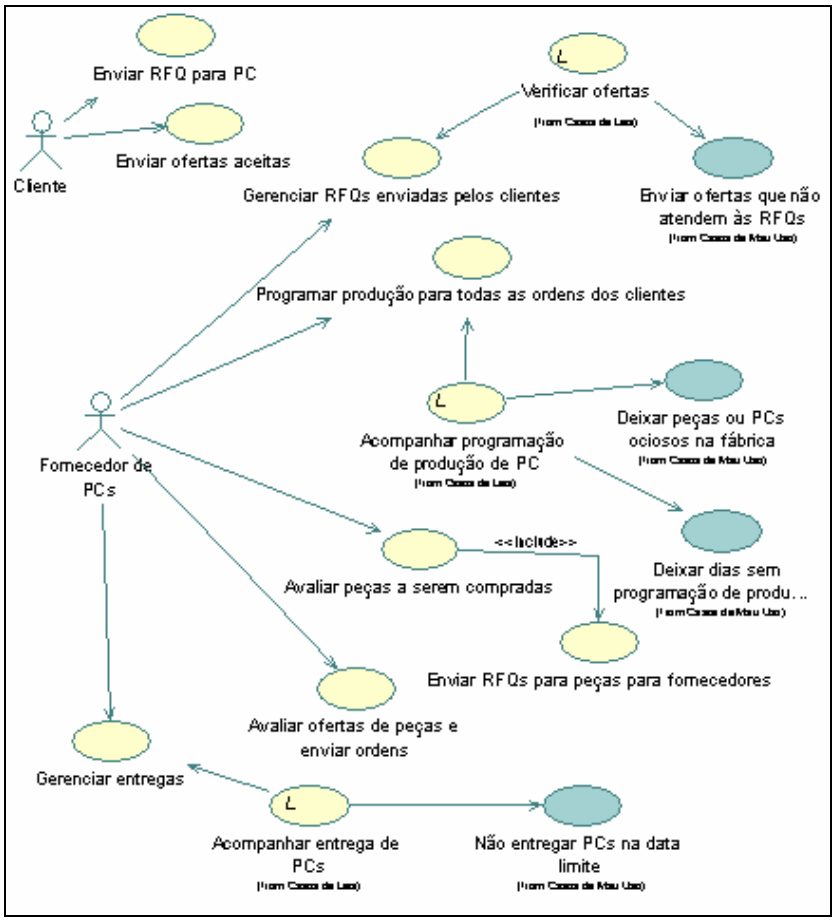

Figura 6. Diagrama de Casos de Leis

Já na Figura 7, é ilustrado o detalhamento de um caso de leis, "Acompanhar entrega de PCs" que está associado ao caso de uso "Gerenciar entregas" e que tem como objetivo impedir que o risco "Não entregar PCs na data limite" ocorra. Note que a partir do *rationale* é possível identificar os elementos que serão especificados no XMLaw, como *Clock* (relógio) e *Norm* (norma).

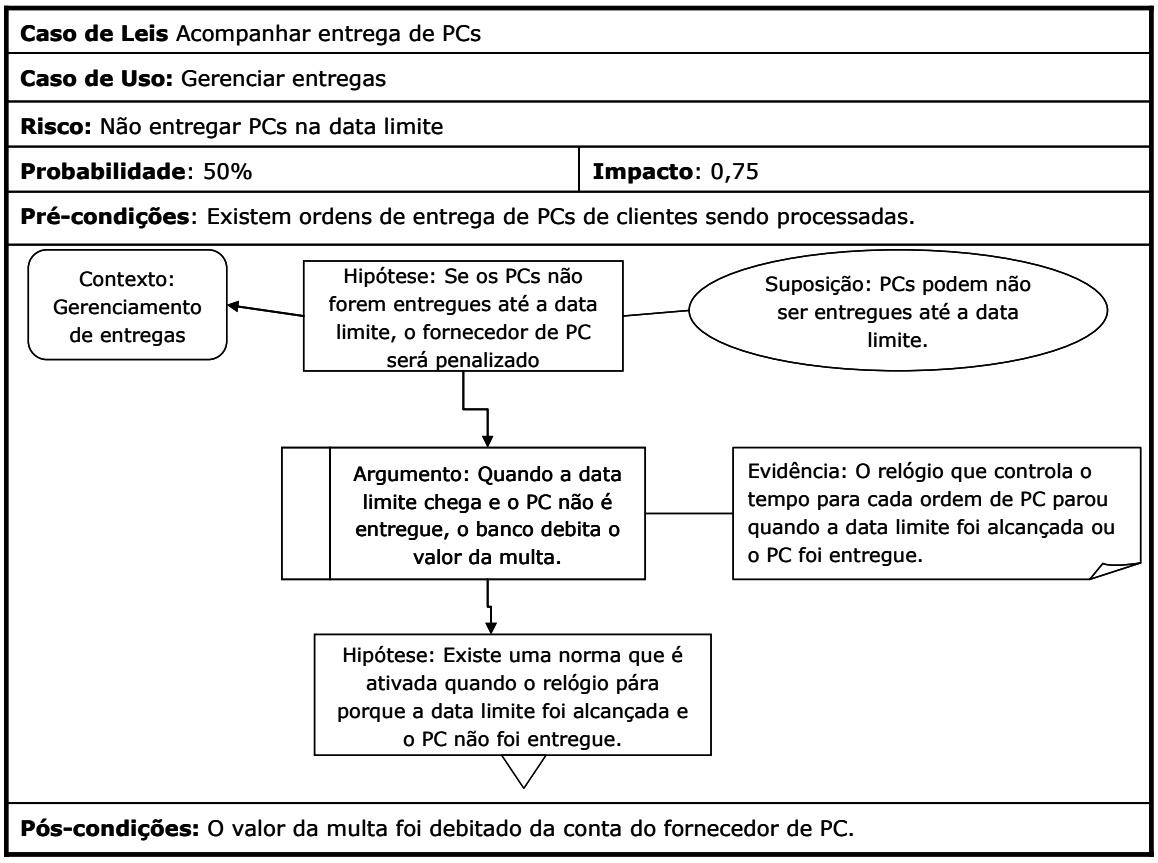

Figura 7. Casos de Leis: Detalhamento

5. Integrando Casos de Leis e Análise de Risco para Criticalidade

Nesta seção, o papel de casos de fidedignidade para o monitoramento da criticalidade de agentes de software será discutido como uma forma de descrever uma aplicação de governança de sistemas multiagentes abertos.

Como visto na subseção 2.4, métodos de análise de risco podem ser estruturados para auxiliarem na engenharia dos requisitos do sistema e aumentarem a fidedignidade do mesmo. Em [Carvalho 2004], requisitos de fidedignidade são modelados como riscos, que por sua vez, guiam a especificação das leis. Baseando-se nisso, o uso de análise de risco para apoiar a derivação de leis e mensurações de criticalidade de agentes está inserido na abordagem proposta.

Um Caso de Leis identifica o risco que está associado àquele determinado caso, analisa a probabilidade e impacto caso o risco efetivamente ocorra e identifica o evento da interação que detecte a ocorrência do risco.

A probabilidade é estimada seguindo a classificação definida pelo Departamento de Defesa dos Estados Unidos [MIL-STD-882C] e é uma classificação do risco com sua respectiva associação de um valor. Um risco pode ser classificado como improvável de ocorrer (0 a 20%), remoto (20% a 40%), ocasional (40% a 60%), provável (60% a 80%) ou freqüente (80% a 100%). A análise da classificação foge ao escopo deste artigo.

O impacto do risco é estimado também seguindo a classificação definida em [MIL-STD-882C]. São eles a saber: catastrófico]0,75;1], crítico]0,5;0,75], marginal]0,25;0,5] ou negligenciável]0;0,25]. Um risco catastrófico é capaz de causar morte, grandes perdas financeiras ou perda total do sistema. Um risco crítico é capaz de causar um ferimento grave, invalidez, alguma perda financeira ou grande perda do sistema. Um risco marginal é capaz de causar um ferimento leve ou pouca perda do sistema. E um risco negligenciável é incapaz de causar qualquer uma das perdas mencionadas.

Um evento da interação detalha o contexto da ocorrência e detalha como o risco pode ser detectado. Ele é importante no caso da especificação de leis porque vai determinar qual o evento da interação do protocolo no qual aquele risco está inserido e que vai determinar a variação da criticalidade do agente.

Identificado o evento da interação do protocolo, como visto na seção 3.2 durante a especificação da criticalidade no XMLaw, faz-se necessário detectar o peso, ou melhor, a importância do evento quanto à variação da criticalidade. O peso W será igual a P x I, onde P é a probabilidade do risco e I o impacto do risco associado àquele evento.

Referenciando a Figura 5, vemos que a análise de risco deve se dar após uma versão preliminar dos Casos de Leis a fim de refiná-los e gerar novos requisitos de leis, caso necessário. Uma vez finalizados a análise de risco, os refinamentos e os requisitos de leis e, por último, o projeto da especificação das leis em XMLaw, a especificação da variação da criticalidade dos agentes é feita, tendo como base a própria especificação XMLaw e os Casos de Leis previamente refinados com a análise de risco.

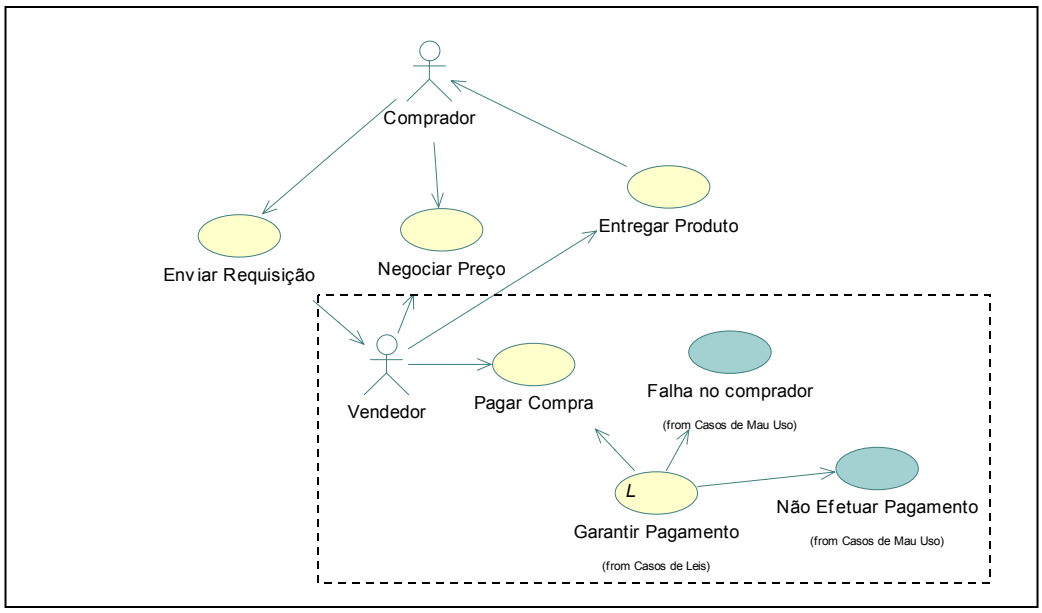

Figura 8. Caso de Uso para a Cena de Negociação

Nas Figura 9 podemos ver um exemplo de Caso de Leis com a hipótese principal gerando as sub-hipóteses, que serão geradas e documentadas a parte, e a especificação do risco, probabilidade e impacto. A Garantia do Pagamento implica em aplicar uma penalidade ao agente para evitar o calote, e em replicar o agente em casos críticos (Figura 4 exibida na sub-seção 4.1), que por sua vez requer o monitoramento de sua criticalidade.

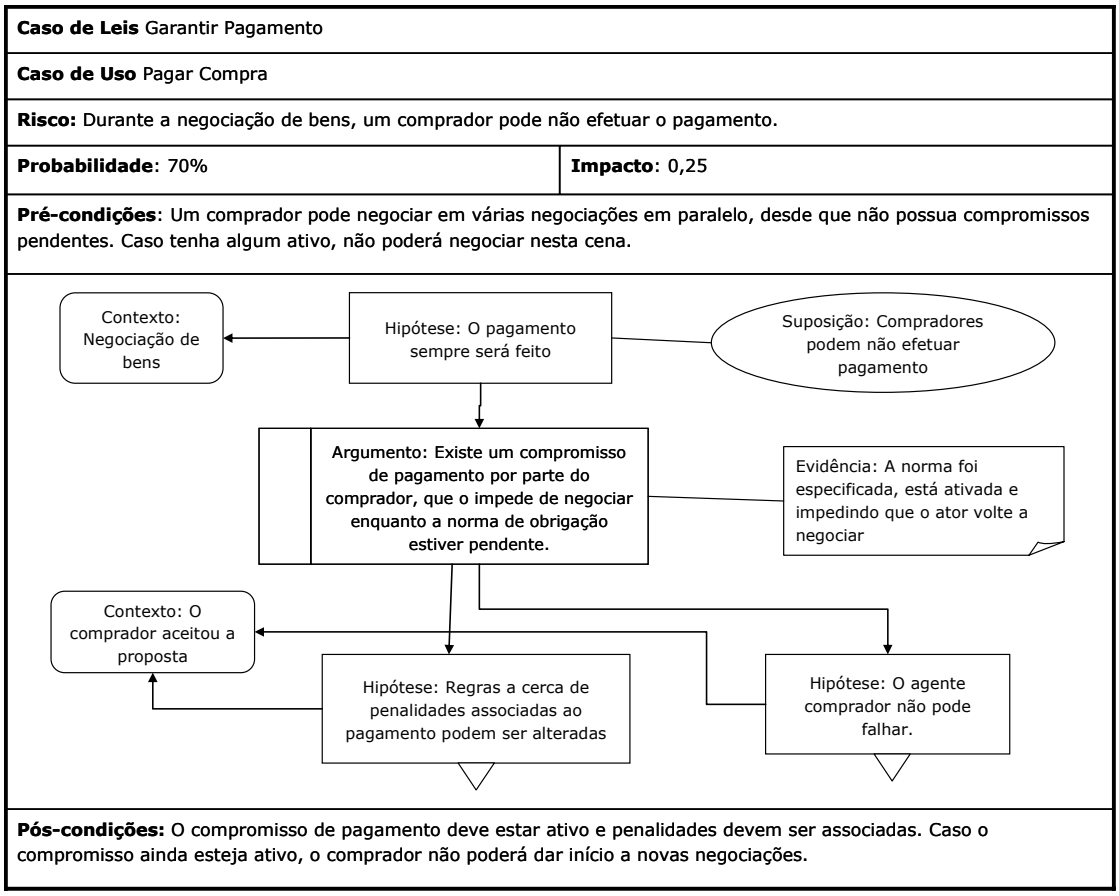

Figura 9. Exemplo de Caso de Leis: Garantir Pagamento

É importante observar que um caso de leis é composto pela documentação principal (o qual é exibido na Figura 9) e pelo detalhamento das sub-hipóteses que porventura surgirem no processo de argumentação da hipótese principal (o qual é exibido na Figura 4 para a sub-hipótese "O agente comprador não pode falhar"). Com este tipo de estratégia espera-se evitar a explosão de informações natural em problemas reais e complexos.

Considerando o risco de falha do comprador, o que resultaria na ausência de pagamento do produto, para se estimar o peso da criticalidade associado a este evento, bastaria multiplicar a probabilidade pelo impacto do risco se concretizar, isto é 70%*0,25 = 0,175. E, no componente de criticalidade do XMLaw, existiria o seguinte código:

```
...
<Increase event-id="obligation_of_payment" event-type="norm_activation" value="0.175" >
        <Assignee role-ref="customer" role-instance="$customer.instance" />
</Increase>
...
```

6. Conclusão

Casos de leis conjugam a aplicação de casos de uso e casos de fidedignidade. A abordagem de casos de uso é útil para mantermos uma visão geral sobre os requisitos que

são identificados e casos de fidedignidade são úteis para registrar o detalhamento da análise.

É importante entendermos a razão pela qual casos de fidedignidade são utilizados no contexto de governança de sistemas abertos. A primeira razão está na documentação das decisões que são tomadas e o *rationale* em torno desta decisão. A decisão em nosso contexto está diretamente relacionada à criação de leis de interação que serão verificadas e aplicadas em tempo de execução. A manutenção da integridade prevista para um sistema está na derivação de leis de interação que serão identificadas a partir da documentação da necessidade, do entendimento de todas as hipóteses, contextos e suposições traçadas para a aplicação e da derivação de leis de interação que verificarão em tempo de execução a integridade da aplicação.

Em nossos experimentos, o uso de análise de risco, ou uma postura em identificar claramente potenciais problemas que serão abordados futuramente, tem se mostrado bastante promissor no intuito de promover a identificação e o detalhamento de estratégias para mitigar potenciais problemas. Além disto no contexto de criticalidade, esta abordagem pode ser utilizada para determinar valores referentes ao nível de variação no monitoramento da importância de agentes.

Referências

Alexander, I.; (2003) "Misuse cases: use cases with hostile intent"; Software, IEEE, 20(1), Jan.-Feb. Page(s):58 - 66.

Avizienis, A.; Laprie, J-C.; Randell, B.; Landwehr, C.; (2004) "Basic Concepts and Taxonomy of Dependable and Secure Computing"; IEEE Transactions on Dependable and Secure Computing 1(1); Los Alamitos, CA: IEEE Computer Society; pags 11-33.

Booch, J., Rumbaugh, I. J., (1999) The Unified Modeling Language User Guide; Addison Wesley.

Carvalho, G.; Paes, R.; Choren, R.; Lucena, C.; (2004) "Towards a Risk Driven Method for Developing Law Enforcement Middleware"; Third International Workshop on Agent-Oriented Methodologies, OOPSLA 2004, Vancouver, British Columbia, Canadá, 24-28 Outubro.

Castelfranchi C., Dignum F., Jonker C.M., Treur J. (1999) "Deliberative Normative Agents: Principles and Architecture". Proceedings of the Sixth International Workshop on Agent Theories, Architectures, and Languages (ATAL-99), Orlando, FL, July 15-17.

Chopra, S.; Meindl, P.. (2004) "Supply Chain Management – Strategy, Planning, and Operations". Pearson Prentice Hall, Second Edition.

Collins, J., Arunachalam,R., Sadeh,N., Eriksson,J., Finne,N., Janson, S. (2005) "The Supply Chain Management Game for the 2006 Trading Agent Competition". Carnegie-Mellon University, Pittsburgh, USA.

Dignum, F. (2001), "Agents, markets, institutions, and protocols". In: Agent Mediated Eletronic Commerce, The European AgentLink Perspective., p.98-114. Springer-Verlag, 2001.

Esteva, M. (2003) "Eletronic Institutions: from specification to development" PhD thesis, Institut d'Investigació en Intel.ligència Artificial, Catalonia – Spain, October.

Firesmith, D.G. (2003) “Security Use Cases”, Journal Of. Object Technology 2(3), May-June.

Gatti, M.A. de C., Lucena, C.J.P. de, Briot, J.-P.; (2006); “On Fault Tolerance in Law-Governed Multi-Agent Systems”. In: 5th International Workshop on Software Engineering for Large-scale Multi-Agent Systems (SELMAS) at ICSE’ 2006.

Hewitt, C. and Jong, P de; (1982) “Open systems on conceptual modeling”. Technical report, Massachusetts Institute of Technology.

Jennings, N. R., (2001), “An Agent-Based Approach for Building Complex Software Systems”, Communications of the ACM, 44(4), pages 35-41, April.

Kollingbaum, M. J. and Norman, T. J. (2003). NoA - A normative agent architecture. In Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, pages 1465-1466.

Luck, M., McBurney, P., and Preist, C., (2003) "Agent Technology: Enabling Next Generation Computing. A Roadmap for Agent-Based Computing", Version 1.0.

Lussier, B. et al. (2004) 3rd IARP-IEEE/RAS-EURON Joint Workshop on Technical Challenges for Dependable Robots in Human Environments, Manchester (GB), 7-9 September, 7p.

Menezes, P. Blauth (1997) “Linguagens Formais e Autômatos”, Editora Sagra-. Luzzato.

MIL-STD-882C: “Standard Practice for System Safety Program Requirements”, US Department of Defense, 1996.

Minsky, N.H., Ungureanu, V., (2000) “Law-governed interaction: a coordination and control mechanism for heterogeneous distributed systems”. ACM Transactions on Software Engineering Methodology, 9(3):273-305.

Paes, R., Carvalho, G. R., Lucena, C.J.P., Alencar, P. S. C., Almeida, H.O.; and Silva, V. T.. (2005), “Specifying Laws in Open Multi-Agent Systems”. In: Agents, Norms and Institutions for Regulated Multi-agent Systems (ANIREM), AAMAS’2005.

Paes, R., (2005) “Regulando a Interação de Agentes em Sistemas Abertos - uma Abordagem de Leis”. Dissertação do Mestrado, Departamento de Informática, PUC-Rio, Março.

Pierce, R. H. and Baret, H. (2005) “Structuring a Safety Case for and Air Traffic Control Operations Room”, Proc. Thirteenth Safety Critical System Symposium, Redmill and Anderson (Eds.). London: Springer Verlag, February.

Sommerville, I. (2004) “Software Engineering”, Addison-Wesley, 7th ed.

Vazquez-Salceda, J., Dignum, V., Dignum, F., (2005), Organizing Multiagent Systems, Autonomous Agents and Multi-Agent Systems, 11, Springer; pages 307-360.

Weinstock, C.B; Goodenough, J.B.; Hudak, J.J., (2004) "Dependability Cases“, CMU/SEI-2004-TN-016, Proc. of The International Conference on Dependable Systems and Networks.

Zambonelli, F., (2002) ‘‘Abstractions and infrastructures for the design and development of mobile agent organizations,’’ in M. Wooldridge, G. Weiss, and P. Ciancarini (eds.), Agent-Oriented Software Engineering II, LNCS 2222, Springer-Verlag, pp. 245–262.