Modelagem de um Sistema Multiagente para Compras de Suprimentos de TI Usando *Agent* UML

Autoria: Edmilson Lucena Néri, Norberto Hoppen

RESUMO

Sistemas multiagentes podem ser desenvolvidos para substituir o trabalho humano em várias atividades que exijam o poder de decisão, através da delegação. A fim de buscar evidências do afirmado e vivenciar uma experiência prática sobre o assunto, iniciou-se uma pesquisa que envolve sistemas multiagentes e processos decisórios e de negociação. Este artigo traz alguns dos primeiros resultados desta pesquisa, em particular, discussão sobre modelagem de sistemas multiagentes através da *Agent* UML - um conjunto de propostas para estender a UML para modelar agentes de software. Como cenário para especificação de um sistema e experimentação da *Agent* UML, escolheu-se o processo de decisão e negociação de compras de suprimentos de TI da Brasil Telecom S.A., onde se propôs uma solução alternativa, via agentes de software, à atual.

INTRODUÇÃO

Automatização e robotização são palavras já há algum tempo presentes nos meios acadêmicos e entre a população como um todo. Robôs que montam carros, sistemas eletrônicos de freqüência de funcionários, bancos de dados que recuperam quase que instantaneamente informações sobre clientes, para citar alguns, já fazem parte do cotidiano das empresas.

Embora já haja delegação de muitas tarefas, em especial das mecânicas, ainda há muito que se fazer, principalmente quando se pensa na automação daquelas ditas como intelectuais, como, por exemplo, o processo de decisão e negociação de compras corporativas. Se bem formalizados, os conhecimentos para a realização de tais tarefas são passíveis de automação. Um conceito que pode ajudar é o de agentes de software, que, em um sentido mais amplo possível, são software aos quais se delegam tarefas, executando-as autonomamente, a partir daí.

Neste artigo, há um relato de uma empresa real, a Brasil Telecom S.A., no tocante às compras de suprimentos de informática. Expõe-se o processo de como são feitas estas compras e sugere-se, como estudo de caso, uma arquitetura de um sistema multiagente que o substituiria.

Uma discussão sobre uma variante da *Unified Modelling Language* (UML), a *Agent* UML, é feita, à medida que se modela o sistema. Uma revisão bibliográfica mais extensa sobre o uso de agentes de software nas empresas, bem como detalhes sobre implementação do sistema proposto (em especial, a implementação dos mecanismos de decisão e negociação), estão fora do escopo deste artigo, mas estão presentes na pesquisa que envolve esta discussão e serão também submetidas à publicação.

AGENTES E SISTEMAS MULTIAGENTES

Não há consenso sobre o conceito de agentes. Um dos maiores problemas que os pesquisadores de agentes têm é justamente o conceito de agentes. Vê-se na literatura várias definições, cada uma tentando adequar-se aos propósitos daqueles que as definem.

Uma boa discussão sobre o conceito de agentes pode ser encontrada em Franklin e Graesser (1996). Lá se encontra uma coletânea de conceitos, com as devidas críticas, bem como, o conceito dado pelos próprios autores, que é: “um agente autônomo é um sistema

situado dentro - e uma parte - de um ambiente que sente este ambiente e age sobre ele, através do tempo, realizando sua própria agenda e assim afetando o que ele sentirá no futuro".

Entretanto, os próprios autores da última definição falam que ela é muito ampla. Para definir subtipos de agentes, definiram propriedades para diferenciá-los e sugeriram uma taxonomia inspirada na taxonomia dos seres vivos.

Mencionou-se o conceito "agente autônomo" sem fazer distinção ao conceito mais simples: agente. Para muitos, o termo "autônomo" é redundante, pois, definem autonomia como uma das propriedades fundamentais de uma entidade para ser considerada um agente.

Deve-se manter em mente que um agente não necessariamente é um software. Liang e Huang (2000), por exemplo, definem agente como sendo uma pessoa ou um negócio autorizado para agir no lugar de outra. Já em desenvolvimento de software, um agente é um programa de computador que pode operar autonomamente e efetua tarefas singulares sem a direta supervisão humana (HOFFMAN e NOVAK apud LIANG e HUANG, 2000).

Uma propriedade também importante de um agente é a sua comunicação com outros agentes. Há até quem defina, estreitamente, agentes tendo a comunicação como pré-requisito. Por exemplo, agentes de software são "componentes de software que se comunicam com seus colegas através de uma expressiva linguagem de comunicação de agentes" (GENESERETH, 1994).

Tal definição evidencia que a abordagem de agentes é muitas vezes pensada como mais de um agente comunicando-se e colaborando entre si.

Sistemas multiagentes, conforme Stone e Veloso apud Giese (1998), são o subcampo da Inteligência Artificial que proporciona os princípios para construção de sistemas complexos, envolvendo múltiplos agentes e mecanismos para coordenação do comportamento inteligente de um conjunto de agentes.

A última definição tem um viés claro da Inteligência Artificial, mas sistema multiagente pode ser entendido como um sistema que envolve mais de um agente autônomo de forma que estes cooperam entre si para que o sistema consiga atingir os seus objetivos. Para que possam cooperar entre si, necessita haver comunicação.

Para facilitar a interoperabilidade entre os agentes, durante suas implementações, recomenda-se a utilização de linguagens padrão de comunicação. Um exemplo de uma destas linguagens é a ACL (*Agent Communication Language)*, que é composta por três componentes: seu vocabulário, uma linguagem anterior chamada KIF (*Knowledge Interchange Format*) e uma outra linguagem chamada KQML (*Knowledge Query and Manipulation Language*). Uma mensagem ACL é uma expressão KQML em que os argumentos são termos ou sentenças em KIF formadas por palavras do vocabulário ACL (GENESERETH, 1994).

MODELANDO AGENTES

Para modelar sistemas baseados em agentes, há diversas técnicas propostas na literatura. Neste artigo, será apresentada a *Agent* UML. Para alguns autores, agentes são abstrações completamente diferentes às existentes. Assim, propuseram técnicas completamente novas (WOOLDRIDGE, JENNINGS E KINNY, 1999; WOOLDRIDGE, JENNINGS E KINNY, 2000; JO, 2001; BRESCIANI et al., 2001). Outros autores acreditam que agentes são muito parecidos com abstrações já existentes, em particular, com objetos. Assim, propuseram extensões de técnicas baseada em objetos (KARACAPILIDIS e MORAÏTIS, 2001; BAUER, MÜLLER e ODELL, 2001; ODELL, PARUNAK e BAUER, 2001).

Agent UML

Há uma organização (http://www.auml.org) que reúne pessoas que desejam estender a *Unified Modeling Language* (UML) para adequá-la aos conceitos e necessidades inerentes a agentes – A *Agent* UML (AUML). No *site* desta organização, encontram-se diversos artigos sobre o tema. Em geral, cada artigo trata de algum diagrama em especial da UML, sugerindo algumas extensões para um bom uso com agentes de software.

É importante lembrar que a UML é uma linguagem para especificação, documentação, visualização e desenvolvimento de sistemas orientados a objetos. Surgiu da união de alguns dos maiores autores e suas metodologias sobre análise e projeto de sistemas baseados em objetos. Hoje é, sem dúvida, a linguagem de modelagem mais difundida para desenvolvimento de software, tanto no meio acadêmico, como nas empresas. Muito de seu sucesso é devido às ferramentas que apóiam os desenvolvedores durante as fases de construção de um programa. Quase todas "falam" UML. Por causa de sua popularidade, muitos autores acreditam que estendê-la para agentes é melhor que criar uma linguagem completamente nova.

Para melhor modelar os diversos aspectos dos sistemas – sejam estáticos, dinâmicos, de implementação – a UML provê vários diagramas: Diagramas de Casos de Uso, Diagramas de Classes, Diagramas de Objetos, Diagramas de Interação (Seqüência e Colaboração), Diagrama de Gráficos de Estados, Diagramas de Atividades, Diagrama de Componentes e Diagramas de Implantação. Além disso, a UML provê alguns mecanismos de extensão, como estereótipos e restrições. Maiores detalhes sobre a UML podem ser vistos na especificação da linguagem (UML 1.4 Specification, 2002).

Bauer, Müller e Odell (2001) preocuparam-se com os diagramas de interação da UML: Seqüência e Colaboração. Eles acreditam que os mecanismos de interação são o coração da *Agent* UML. Dentre as extensões sugeridas, encontram-se a inserção do conceito de papéis, com a abordagem de agentes, estruturas para declarar *threads* de interação, protocolos intervalados e aninhados, e nova semântica para as setas que representam as interações. Os autores relatam a experiência de desenvolvimento de um sistema baseado em agentes, onde houve uma boa aceitação das extensões da UML. Um artigo mais detalhado, mas com o mesmo teor, é encontrado em Odell, Parunak e Bauer (2001).

De encontro aos esforços dos autores acima mencionados, Lind (2001) demonstra como representar, com a UML padrão, protocolos de interação entre agentes, utilizando apenas os próprios mecanismos de extensão oferecidos por ela. Há uma preocupação em não estender a UML, a fim de evitar dialetos da mesma. Contudo, Lind admite que a UML não é perfeita para a modelagem de agentes.

Bauer (2001) propôs extensões para os Diagramas de Classe da UML, de forma a adequá-la à realidade dos sistemas baseados em agentes. As mudanças são tantas que se criou um novo diagrama, chamado Diagrama de Classes de Agentes, onde são inseridos os conceitos de papéis, ações, capacidades, protocolos, etc.

Outras sugestões de adaptação do Diagrama de Classes da UML também foram feitas, como a criação do Digrama de Ontologia, que declara formalmente como será feita a comunicação entre os agentes (BERGENTI e POGGI, 2000).

Outras informações sobre a AUML podem ser vista diretamente no *site* da organização (www.auml.org), bem como na modelagem do sistema multiagente proposta neste trabalho.

CENÁRIO DO ESTUDO DE CASO

Desejava-se vivenciar a experiência de modelar um sistema multiagente com a *Agent* UML. Também, e principalmente, buscava-se mostrar, através de um exemplo, como agentes de software podem ser usados como soluções viáveis e, possivelmente melhores, para algumas tarefas que atualmente não são comumente vistas como passíveis de automação. As compras de suprimentos de informática feitas pela Brasil Telecom S.A., uma das maiores empresas de telecomunicações brasileiras, propiciaram um cenário para a proposição de um sistema multiagente, como solução alternativa à corrente.

Antes das apresentações das situações atual e proposta, é importante discorrer um pouco sobre as compras de suprimentos de informática feitas pela empresa em questão. Como toda empresa do seu porte, a Brasil Telecom S.A. tem vários sistemas de informação, conseqüentemente, necessita de um grande aparato de hardware: computadores, impressoras, equipamentos de rede, cabos, etc. As compras de suprimentos de informática que se refere o presente trabalho são aquelas necessárias para a manutenção da infra-estrutura de TI, usada pelos colaboradores da empresa para executar aplicações como pacotes *office*, correio eletrônico, navegadores *Web* e *front-ends,* de uma forma genérica. Exemplos podem ser: memórias, discos rígidos, teclados, baterias para placas mães, cabos coaxiais, etc. Não se está fazendo referência aos itens de TI relacionados com o processamento e armazenamento das grandes aplicações da empresa.

Há, na Brasil Telecom, equipes responsáveis pela manutenção por estes itens de hardware e software, os chamados Núcleos de TI. São eles que sinalizam a necessidade da reposição de alguma peça.

Solução Atual

Como existe uma certa regularidade na compra dos itens, a Diretoria de Materiais e Serviços (DMS), que é responsável pelas autorizações de compras da empresa, juntamente com a equipe Obtenção, da Gerência de Planejamento e Recursos da Diretoria de Tecnologia da Informação, elaboram um contrato aberto ou contrato guarda-chuvas.

Um contrato aberto funciona da seguinte forma: enumera-se os produtos e quantidades previstos que serão necessários comprar durante um certo período de tempo, por exemplo, seis meses. A planilha resultante é divulgada para os possíveis fornecedores dos produtos, formando-se uma concorrência. A empresa que tiver a menor cotação geral, ou seja, o somatório das quantidades de cada produto multiplicadas pelos valores unitários, assinará um contrato com a Brasil Telecom, responsabilizando-se a, durante a vigência do contrato, vender os produtos aos preços preestabelecidos, até a quantidade cotada. Exemplificando, se na planilha há uma linha que solicita a cotação de 50 pentes de memória de 128 MB e a empresa vencedora da concorrência estabeleceu que o preço unitário era de R$100,00, então, durante o período de vigência do contrato, a Brasil Telecom pode comprar até 50 desses produtos a R$100,00 cada, independentemente se comprará todos ou não e, que sejam comprados em lotes ou em unidades.

Quando um Núcleo de TI percebe a necessidade de reposição de um item, solicita a compra do mesmo à Obtenção, esta última verifica se há previsão no contrato aberto. Se houver, encaminha um pedido para a DMS que autoriza a compra, de forma rápida, uma vez que há uma "pré-autorização" (o contrato aberto). Se não houver previsão, a Obtenção pesquisa os possíveis fornecedores e encaminha um pedido à DMS. Como não há previsão no contrato aberto, a DMS leva mais tempo para autorizar, pois precisa analisar o pedido de compra.

Voltando aos contratos abertos, algumas questões surgem deste tipo de prática. Parece evidente que o processo de compra de suprimentos será facilitado e agilizado, diminuindo o

tempo despendido pelo pessoal especializado em compras e dando a velocidade necessária para, neste exemplo, as equipes técnicas, que precisam reparar rapidamente os equipamentos. É importante ressaltar uma diretriz da empresa para que não haja estoque desses produtos.

Note-se também que a barganha neste procedimento é feita sobre o valor total de todos os itens, com as quantidades estimadas. Mas, como saber se as quantidades estimadas vão ser realmente as reais necessidades? Pode-se inferir que, ao final do período de vigência do contrato, as reais necessidades devem divergir das estimadas e a escolha do melhor fornecedor poderia ter sido outra.

Será que se a concorrência fosse feita produto por produto, o somatório de todos os itens seria menor? Até que ponto o fato de ser uma possível grande venda, o fornecedor diminuiria seus preços? O fato de o fornecedor comprometer-se a manter o preço durante toda a vigência do contrato não pode aumentar a cotação dos mesmos?

Diante de tantas questões, a mais evidente e suficiente para garantir a existência de contratos abertos é a agilidade. Há também o desejo de manter a compra de suprimentos de forma centralizada (como no caso da BrT). A centralização tem suas vantagens. Por exemplo, os núcleos não precisam entender de compras, tampouco perder tempo em decisão e negociação. Há também um maior controle da empresa quando se tem um processo centralizado.

Contextualizando para o caso das obtenções de suprimentos para os Núcleos de TI da BrT, tem-se: quando um técnico observa a necessidade de um item de reparo, faz a solicitação deste item a equipe de Obtenção. Não importa em que Núcleo este técnico está alocado (há Núcleos em todas as filiais e, às vezes, mais de um por filial), o pedido é feito para a Obtenção de TI, que fica na matriz. Caso o item esteja previsto no contrato aberto, tudo ocorrerá rapidamente, como necessário para o pessoal técnico, uma vez que a aprovação da compra foi tomada *a priori*.

Aprofundando a análise, a Obtenção de TI também não é a última palavra no processo de compras. Quem, de fato, negocia com o fornecedor e emite pedido é a Diretoria de Materiais e Serviços. Os contratos abertos também são criados e negociados pela DMS. A Obtenção participa definindo quais os itens necessários (especificações) e em que quantidade.

Uma solução alternativa para a compra de suprimentos de TI

Inicialmente, a equipe de Obtenção de TI estimaria as necessidades futuras de suprimentos de TI, elaborando uma tabela com as especificações dos produtos e as quantidades provavelmente necessárias. A DMS então, faria uma consulta, possivelmente através de um mercado eletrônico, dos preços, incluindo-os na tabela dos produtos. Até aqui, o processo está idêntico ao atual.

Diferentemente do que ocorre hoje, o próximo passo não seria uma concorrência para escolher um fornecedor para todos os itens – um contrato aberto. A sugestão é substituir tal tipo de contrato pela delegação da decisão de compra e negociação da mesma a agentes de software. Todas as vezes que houvesse necessidade de compra de um suprimento, o Núcleo criaria um agente de software que solicitaria cotações aos fornecedores, ordenaria as alternativas e negociaria a compra com o vencedor da concorrência.

Fundamentalmente, estaria sendo feito um processo de descentralização de compras. Esta descentralização, entretanto, não implicaria necessidade de treinamento do pessoal dos Núcleos para compras, tampouco um descontrole da DMS ou da Obtenção sobre as compras na Brasil Telecom. O conhecimento do processo estaria embutido no agente. Já o controle das compras estaria garantido pelas informações que os agentes reportariam. O treinamento do

pessoal dos Núcleos seria pequeno, apenas mostrando como criar e configurar os agentes. Para quem trabalha com suporte técnico de informática, não seria um grande desafio.

Nesta abordagem, haveria uma concorrência para cada produto, levando, possivelmente, a um custo total menor. Esta possibilidade, entretanto, não pode ser verificada, mas sim inferida, se o sistema multiagente fosse implantado na Brasil Telecom e comparado com os dados históricos, ou ainda melhor, com um possível contrato guarda-chuva em paralelo. Não se tem a pretensão de afirmar que a solução por agentes seria mais eficiente, mas a possibilidade existe, segundo a percepção da própria equipe de Obtenção.

ESPECIFICAÇÃO DO SISTEMA MULTIAGENTE

Os compradores delegam a compra de suprimentos de TI solicitados pelos Núcleos a agentes de software. Este processo inclui a busca por fornecedores, a solicitação de cotações, decisão de qual fornecedor comprar (caso haja decisão de comprar), negociação com o fornecedor escolhido, para melhorar o acordo, e relatórios sobre o processo.

A Brasil Telecom, como uma grande compradora, pode impor algumas condições aos seus fornecedores de suprimento de TI, como ser detentora do "mercado eletrônico", que também será um agente de software, e exigir que os fornecedores tenham também seus agentes de software para vender seus produtos. Estes agentes serão semelhantes (quase duais) aos agentes de compras, aos quais serão delegadas as condições de venda, bem como a negociação da mesma.

As variáveis de decisão para compras de suprimentos especificadas pela equipe de Obtenção da Brasil Telecom foram levantadas em entrevistas e, de certa forma, condizem com a literatura (BAILY et al., 2000; SLACK et al., 1997). São elas: preço, prazo de entrega, quantidade, qualidade, prazo de pagamento e marca.

Alguns comentários sobre as variáveis de compras mencionadas. Na Brasil Telecom o "prazo de pagamento" para tais tipos de compras é único. Geralmente, em torno de trinta dias, feitos em um único pagamento. É comum, entretanto, a mudança de tais prazos. Quem determina o prazo é a própria BrT, restando aos fornecedores aceitá-lo. Assim, a variável "prazo de pagamento" não entrará na negociação entre os agentes, ficando como uma variável restritiva.

Em relação à variável "marca", ela também é restritiva. Caso sejam enumeradas as marcas que serão aceitas, todas as demais, por exclusão, não podem ser compradas.

A "qualidade" é, sem dúvida, a variável mais difícil de ser mensurada. Um produto pode ter uma qualidade alta para determinado comprador, mas baixa para outro, variando de suas experiências anteriores, adequação para seus propósitos, etc. Assim, a qualidade tem que ser dita pelo comprador.

A única variável de compra que é tradicionalmente lembrada e não consta na relação é "fornecedor". Na verdade, ela está implícita. Assume-se que qualquer fornecedor participante do "mercado" já esteja habilitado para suprir os itens desejados. Eles são indiferentes em preferência.

A escolha do melhor fornecedor será realizada através do uso de funções de valor. Estas funções também possibilitarão o uso de algoritmos para a negociação colaborativa. As variáveis de decisão de compra são imprescindíveis para a criação da função de valor.

Serão três tipos de agentes: o Comprador, o Fornecedor e o Mercado. Cada um deles terá, resumidamente, as seguintes características:

a) Agente Mercado

Será um único agente, responsável por manter o cadastro dos Agentes Compradores e Agentes Fornecedores que estão ativos no mercado. Toda vez que um Agente Comprador

solicitar os possíveis Agentes Fornecedores para o Agente Mercado, este último proverá uma lista com tais possíveis fornecedores.

O Agente Mercado terá o poder de decidir quem poderá entrar e continuar no mercado. Ele também será responsável pelo histórico e tentativas de compras, para controle do pessoal de Compras (no caso, Obtenção de TI ou DMS).

É o Agente Mercado que tem as especificações dos itens de compra, bem como a lista com os valores de qualidade de cada produto. Essas listas são determinadas pelo pessoal de Compras, através de interface com o Agente Mercado.

b) Agente Comprador

Cada vez que se necessitar comprar um item, será criado um Agente Comprador, que fará interface, inicialmente, com o Comprador, neste caso, o próprio pessoal dos Núcleos. Com esta interação, o agente terá que receber a especificação do item e as algumas das variáveis de decisão. Também terá que montar a função de valor, de acordo com as preferências do Comprador e com as especificações de qualidade que o Agente Mercado dará.

Depois de dotado de todas as informações necessárias para compras, o Agente Comprador pedirá ao Agente Mercado uma lista com os possíveis fornecedores. De posse desta lista, enviará pedido de cotação de venda. Depois de recebidas as cotações, o Agente Comprador elegerá a melhor – aquela que tiver o maior retorno da função de valor. Eleito qual o melhor fornecedor, o agente iniciará um processo de negociação colaborativa (ganha-ganha) com o Agente Fornecedor.

Fechada a compra, o Agente Comprador informará ao Agente Mercado o histórico de todo o processo, para fim de controle das equipes de Obtenção e da Diretoria de Materiais e Serviços.

c) Agente Fornecedor

O Agente Fornecedor assume várias características duais do Agente Comprador. A fundamental diferença é que este agente não escolhe o melhor comprador. Contudo, a interação com o Fornecedor, captando suas preferências e montando uma função de valor, é de fundamental importância para viabilizar a negociação, em especial, a colaborativa.

Limitações

O sistema multiagente proposto terá algumas limitações, como:

a) Restrição do universo de itens de compra;

b) Controle do Agente Mercado pela BrT;

c) Impossibilidade dos Agentes Fornecedores de escolher o melhor comprador. Como simplificação, o fornecedor nunca "recusará" um cliente, se tiver condições de entregar o produto dentro do prazo exigido.

MODELAGEM

A modelagem completa do sistema é bastante extensa, fugindo do propósito deste documento. Assim, serão apresentados apenas alguns elementos da modelagem, como ilustração, atentando-se para discussões pontuais sobre os diagramas, incluindo sugestões próprias de como melhor estender os diagramas de UML, uma vez que ainda não houve a definição do padrão *Agent* UM.

A modelagem computacional para os mecanismos de decisão e negociação de compras (através de funções de valor) foi realizada, porém está fora do escopo deste artigo.

Diagramas de Casos de Uso

Um "caso de uso" é uma unidade funcional de um sistema, caracterizado por seqüências de mensagens trocadas entre o sistema e os atores e por um conjunto de ações realizadas por este sistema. Um "ator" define um coerente conjunto de papéis que usuários de uma entidade podem desempenhar quando interagem com a mesma.

O Diagrama de Casos de Uso mostra atores e casos de uso juntos, com seus relacionamentos. Os casos de uso são representados por elipses e os atores, como "bonecos". Há alguns problemas no uso destes diagramas para modelar sistemas multiagentes. Talvez, o maior deles seja como representar um agente de software. Como casos de uso? Como atores?

Segundo a experiência de Cossentino, Chella e Lo Faso (2000), freqüentemente, é melhor modelar agentes de software como casos de uso e, todas as outras entidades externas, como atores. Eles também acreditam que outros agentes, que se relacionam com um agente qualquer, devem ser modelados como atores.

Todo o raciocínio desses autores está baseado nos conceitos acima expostos e no seguinte conceito de agente: "Um agente é um sistema computacional encapsulado que está situado em algum ambiente e é capaz de agir de forma flexível e autônoma neste ambiente, de forma a atingir seus objetivos" (ALBUS, McCAIM, LUMIA apud COSSENTINO, CHELLA e LO FASO, 2000).

Fazendo um paralelo entre este último conceito e o conceito de casos de uso anteriormente apresentado, os autores acreditam que um "sistema computacional encapsulado e que age sobre um ambiente" é uma "unidade funcional de um sistema" que se comunica e realiza ações. Já a interação com o ambiente, incluindo com outros agentes, está relacionada com as "seqüências de mensagens trocadas".

Por outro lado, como a implementação de cada agente, em um sistema multiagente, independe das implementações dos demais, sendo necessário apenas a padronização da comunicação, um agente pode enxergar um outro como uma entidade externa, com o qual só se preocupa em como se comunicar, para atingir seus objetivos. Assim, quando se está modelando um agente (através de casos de uso), os outros podem ser vistos como atores.

Outros autores, como FLAKE, GEIGER e KÜSTER (2001), não optaram da mesma forma. Eles representaram seus agentes de software através de ambos: atores e casos de uso. Fizeram uma proposta de extensão da UML, onde há atores convencionais, atores que são agentes (representados com a cabeça quadrada), casos de usos (*use cases*), casos de objetivos (*goal cases*), casos de reação (*reaction cases*) e nuvens, que representam o ambiente.

Apesar da extensão torna-se interessante para a representação de agentes, mostrando as relações entre si e entre os agentes e o ambiente externo, houve um excesso de mudanças, constituindo um diagrama completamente novo, sendo pouco considerá-lo apenas uma extensão do diagrama de casos de uso da UML.

A diferenciação gráfica entre atores convencionais e atores que são agentes de software, entretanto, parece bastante útil para um bom entendimento do sistema, tanto por parte dos desenvolvedores do sistema, como para os usuários do mesmo.

Uma sugestão para representar sistemas multiagentes, através de Diagramas de Casos de Uso, é ter vários diagramas, onde se mostra cada agente, um por um, como casos de uso. Os demais ficam representados como atores, seguindo, porém, a notação da cabeça quadrada. Poder-se-ia ter um primeiro diagrama que traria todos os agentes como atores (Figura 1) e que, um por um, seriam "explodidos" em casos de uso, em próximos diagramas.

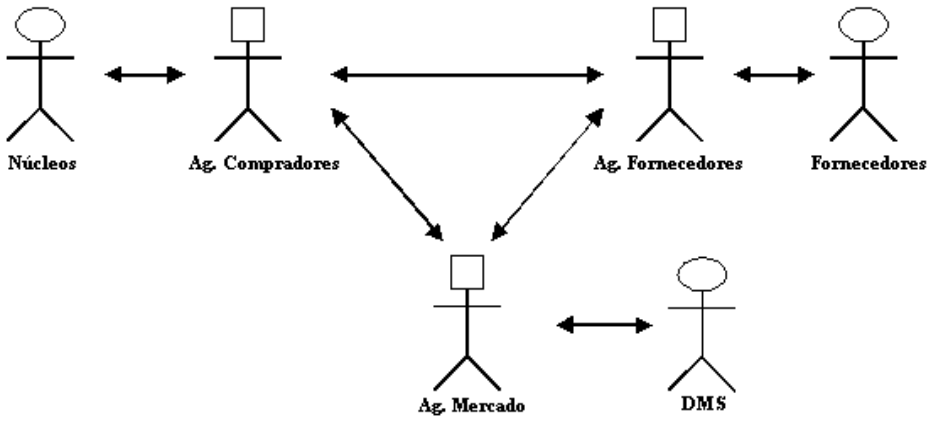

Figura 1 - Diagrama de Contexto Explodido em Atores

Os atores Agentes Compradores serão "explodidos" em um Diagrama de Casos de Uso. Note, no diagrama da Figura 2, que o retângulo reúne vários casos de uso, que, juntos, representam um Agente Comprador. Todavia, podem existir vários Agentes Compradores e Agentes Fornecedores. Assim, decidiu-se manter nos diagramas a nomenclatura no plural. Muitas vezes, contudo, pelo contexto, será percebido que se fala em uma instância destas "classes" de agentes, ou seja, estará se mostrando como um único agente se comporta, mas o discorrido servirá para quaisquer agentes dessas "classes". Outras vezes, fala-se sobre algumas ou todas as instâncias.

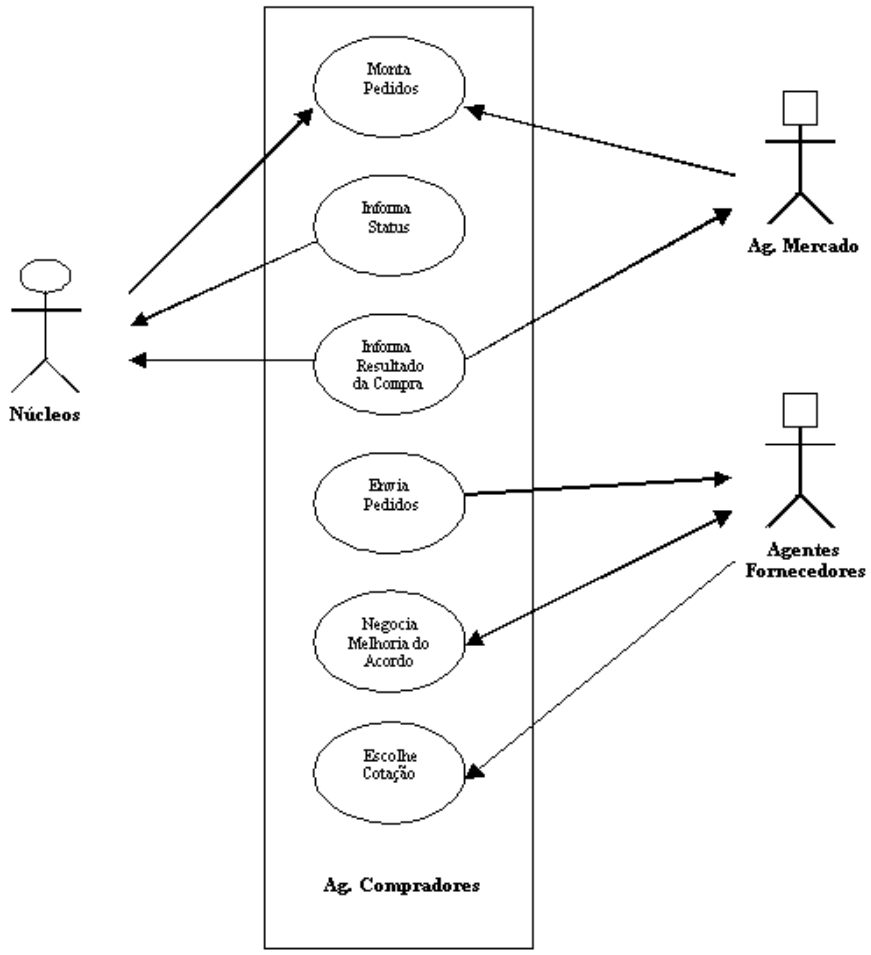

Figura 2 - Diagrama de Casos de Uso (Explosão dos Agentes Compradores)

Cada Caso de Uso do Diagrama deve ser apresentado também em forma de tabela ou quadro, como no Quadro 1. Contudo, por limitações do tamanho do artigo, será mostrado apenas o Caso de Uso Monta Pedidos. Os demais serão omitidos, mesmo tendo sido construídos durante a pesquisa.

Quadro 1 - Caso de Uso: Monta Pedido

Nome	Monta Pedidos
Atores envolvidos	Núcleos e Ag. Mercado.
Outros Casos de Uso referenciados	-
Precondições	O Agente Comprador já ter sido criado.
Iniciação	O Núcleo informa os requisitos de compra (produto, unidade, quantidade, quantidade máxima e prazo máximo de entrega, núcleo solicitante, técnico solicitante).
Descrição da execução	Preencher requisitos de compra, informados pelo Núcleo solicitante e pelo Agente Mercado (prazo de pagamento, preço máximo, "qualidade" de cada uma das marcas, possíveis fornecedores, marcas aceitáveis); Criar estrutura de preferências.
Conclusão	Formatar um pedido de cotação para ser enviado aos possíveis fornecedores (dependendo da implementação, um pedido pode ser uma estrutura única ou, simplesmente, uma verificação se todos os atributos necessários para fazer um pedido foram preenchidos).

Atributos dos Agentes

Alguns atributos são importantes para um bom entendimento de como funcionará o sistema multiagente, bem como para orientar o processo de implementação. Dependendo da implementação, estes atributos podem assumir diversas formas e nomes, mas sempre existirão. Houve na pesquisa uma definição dos principais atributos, com seus respectivos comentários.

Conjunto de Mensagens

Na próxima seção serão mostradas as interações entre os atores do sistema. O primeiro passo, entretanto, é definir um conjunto de mensagens, para que seja possível a comunicação. Como as mensagens possuem nomes sugestivos e há limitações do tamanho do texto, não será apresentado aqui o conjunto de mensagens desenvolvidos na pesquisa, com suas respectivas descrições, origens, destinos e parâmetros. O leitor pode acompanhar através dos diagramas as mensagens.

Diagramas de Interação

Os diagramas de interação servem para explicitar o comportamento dinâmico do sistema modelado. A UML especifica dois tipos de diagramas de interação, o Diagrama de Seqüência e o Diagrama de Colaboração. O primeiro deles enfatiza a ordem em que os eventos ocorrem, já o seguinte, a estrutura dos objetos. Contudo, os dois são equivalentes na semântica, diferem apenas no formato, ficando a cargo do modelador qual ele acha mais interessante para representar o que deseja.

Escolheu-se, para a representação deste sistema, utilizar apenas os diagramas de seqüência, por terem sido considerados os mais adequados, além de terem sido encontrados mais facilmente na literatura revisada.

Na Figura 1, que mostra os atores convencionais e os atores agentes, não fica claro como se dará o relacionamento entre estes atores. Para tanto, serão usados os Diagramas de Seqüência, detalhando como são essas interações.

Serão apresentados agora alguns dos diagramas de seqüência construídos. O primeiro deles mostrará a interação entre os Núcleos e os Agentes Compradores.

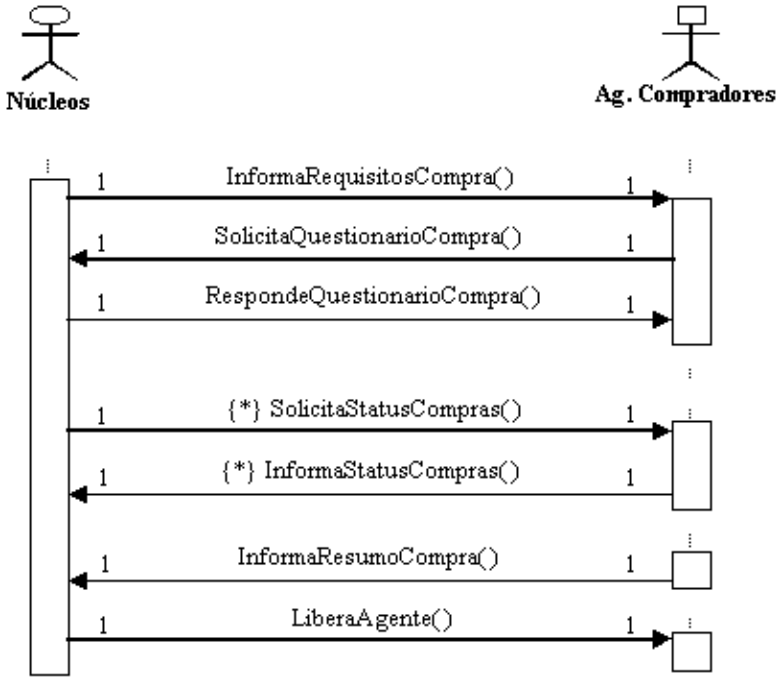

Figura 3 - Diagrama de Seqüência: Núcleos e Ag. Compradores

Como pode ser visto na Figura 3, a primeira interação entre o Núcleo e o Agente Comprador é informar quais são os requisitos da compra. O próximo passo é dado pelo Agente Comprador que envia um questionário para inferir as preferências do solicitante, que responde, sendo esta etapa imprescindível para a criação da função de valor.

Os numerais "1", vistos no diagrama, especificam a cardinalidade da relação. Como podem ser observadas, todas as relações são um-para-um. Isto é devido ao fato de, a cada compra que o Núcleo deseja realizar (por produto), é criado um agente de software para delegar esta função.

As interações "SolicitaStatusCompra( )" e "InformaStatusCompras( )" estão marcadas pelos símbolos "{*}" para informar que podem ser feitas várias vezes, a qualquer momento. As demais, só acontecem uma vez durante toda a "vida" do agente.

Ao terminar o processo de compra, o Agente Comprador informará um resumo do processo para o solicitante (Núcleo), que, em seguida, libera o agente.

Nos diagramas de seqüência, as linhas pontilhadas verticais indicam a linha de vida das entidades. O tempo aumenta de cima para baixo na linha. Note-se também que há retângulos finos acompanhando as linhas pontilhadas. A continuidade do retângulo significa que o ator em questão está processando, exclusivamente, a resposta para a mensagem recebida. Como o Núcleo é um ator externo – não faz parte do sistema – representam-no como um retângulo contínuo, do início ao fim da sua linha de vida. Já o Agente Comprador divide suas atenções entre as interações com o Núcleo e com os outros dois agentes. Assim, só em alguns momentos, seu processamento pára, aguardando uma resposta a alguma mensagem.

Há um importante diagrama que mostra as interações entre os Agentes Compradores e Fornecedores:

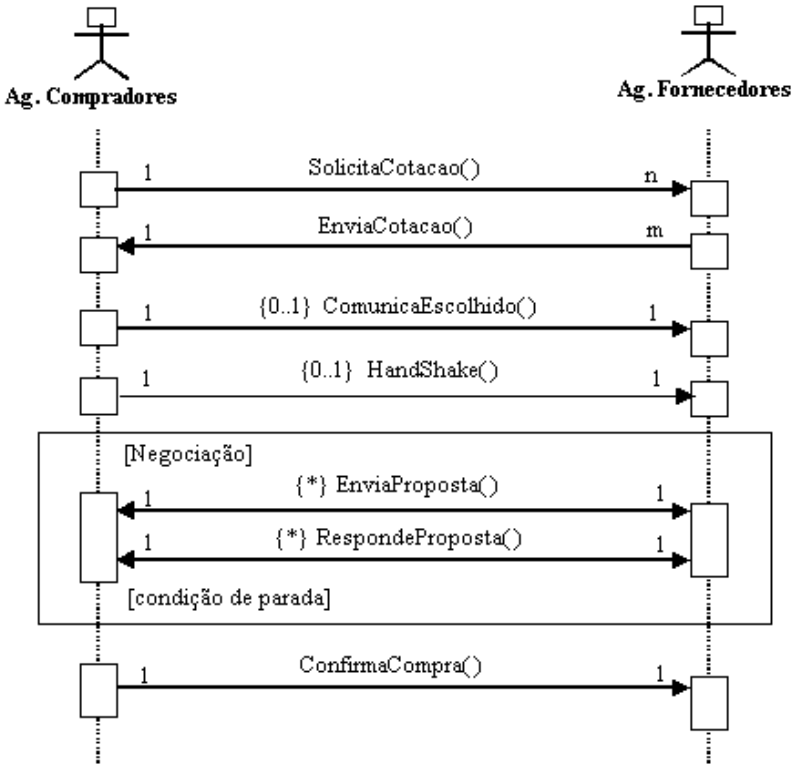

Figura 4 - Diagrama de Seqüência: Ag. Compradores e Ag. Fornecedores

Alguns comentários são necessários para um bom entendimento do diagrama da Figura 4. O primeiro deles é sobre cardinalidade. No primeiro diagrama, a cardinalidade das relações era da forma um-para-um. Agora, vêem-se também relações do tipo um-para-muitos (ou 1 para *n*). Por exemplo, quando o Agente Comprador solicita cotações, ele pode estar solicitando para n fornecedores. Dos *n* fornecedores, *m* podem enviar cotações para o Agente Comprador ($m \leq n$). Entretanto, observe que a mensagem que indica qual foi o Agente Fornecedor escolhido volta a ter cardinalidade um-para-um, ou seja, o Agente Comprador envia a mensagem "ComunicaEscolhido( )" apenas para aquele que ganhou a concorrência.

Um outro ponto interessante são os símbolos {0..1} na mensagem "ComunicaEscolhido( )". Isto quer dizer que nenhum ou, no máximo, um Agente Fornecedor vai recebê-la.

A mensagem "*HandShake()*" é importante para o bom funcionamento dos algoritmos de decisão e negociação, podendo ser ignorada para a presente discussão.

O retângulo que envolve as mensagens "EnviaProposta()" e "RespondeProposta()", juntamente com os rótulos "[Negociação]" e "[condição de parada]", indicam que há uma estrutura de repetição entre os agentes. O que acontece aqui é que, os negociadores (Comprador e Fornecedor) alternam-se, em sucessivos envios de propostas, por parte de um deles, até o outro aceitar algum dos acordos possíveis contido em uma proposta. Uma vez aceito algum acordo de uma proposta, este passa a ser o acordo atual e os papéis são invertidos: quem propunha, escolhe e vice-versa. Esse processo alternado de envio de proposta e escolha de novo acordo só termina quando a condição de parada do algoritmo implementado for satisfeita.

Diagramas de Atividades

Há outros diagramas na UML que mostram o comportamento dinâmico dos sistemas modelados. Dois deles são o Diagrama de Estados e o Diagrama de Atividades. Os dois têm muito em comum, em especial, têm estados e transições de estados. A diferença é que no primeiro, o foco está nas transições dos estados, evidenciando os eventos que causam tais transições. Uma outra diferença é que, no diagrama de atividades, os estados são geralmente do tipo "estado de ação". Um estado é como se fosse uma fotografia de um sistema em um determinado momento. Já um "estado de ação" demonstra qual operação está sendo realizada naquele momento do sistema. Para acompanhar tal tipo de diagrama, observam-se os estados de ação e vê-se que a transição de um estado de ação para outro é feita pela finalização do primeiro estado, sem necessidade de eventos externos para haver mudanças de estado. Não obstante, pode-se utilizar estados "normais" no Diagrama de Atividades, bem como estados de ação no Diagrama de Estados.

Os Diagramas de Atividades mostram o fluxo de operações necessário para se atingir um determinado fim. A circunferência "cheia" indica o início das atividades. A "cheia" dentro de uma "vazia", indica o término. Como forma ilustrativa, escolheu-se o fluxo principal de operações do Agente Fornecedor. Os dois outros diagramas, referentes aos dois outros agentes, também foram desenvolvidos na pesquisa.

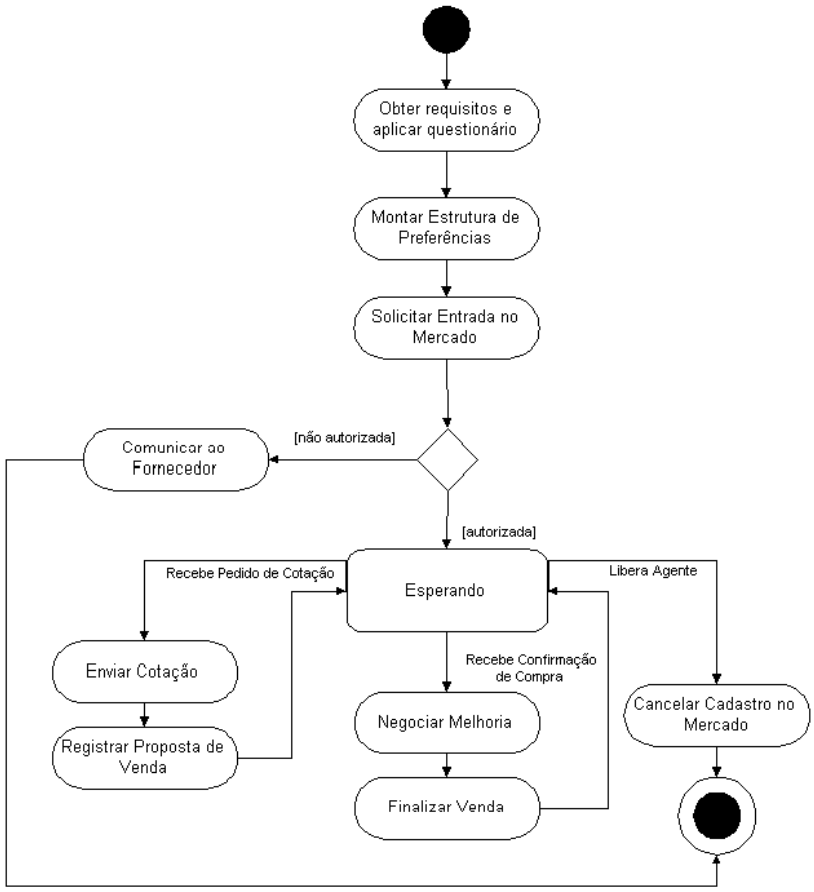

Figura 5 - Diagrama de Atividades de um Ag. Fornecedor

CONSIDERAÇÕES FINAIS

♦ Embora o sistema não tenha sido desenvolvido por completo e aplicado na empresa, este artigo traça direções para uma automatização da função Compras. O próprio exercício de modelagem força a sistematização do conhecimento sobre as tarefas que o sistema modelado realizará. Assim como realizado para o cenário "Compras", agentes de software podem viabilizar novas abordagens para funções organizacionais. A delegação de decisões de gestão a softwares abre um grande leque de oportunidades para investigação acadêmica, bem como para as empresas dispostas a sistematizar, formalizar e implementar, através destes agentes, seus processos decisórios;

♦ A *Agent* UML ainda é um conjunto de propostas para extensão da UML. Assim, ainda há diversas formas diferentes para expressar conceitos idênticos. Evidentemente, isto é o caminho normal para o estabelecimento de um padrão, que é o que se busca. Como linguagens de modelagem são usadas por diversos profissionais para representar suas abstrações do mundo real, os administradores também devem se engajar na definições dos padrões destas linguagens. Este trabalho corrobora esta tese, apresentando, através de um exemplo, algumas dificuldades e soluções para melhor representar abstrações do "mundo" empresarial;

♦ Embora existam diversos esforços, a *Agent* UML não parece ser uma solução ideal, ou próxima disso, para modelar agentes, pois ainda está bem enraizada nos conceitos de objetos, não de agentes. Em casos que não há mecanismos de aprendizado nos agentes, como o proposto neste trabalho, ela ainda parece oferecer bons resultados. Entretanto, mesmo os agentes sem tais mecanismos têm algumas características diferentes de objetos. Assim, uma extensão de uma linguagem para modelar objetos pode não ser razoável para modelar agentes;

♦ Como não se determinou a forma em que o sistema multiagente deveria ser implementado, não se chegou, na modelagem feita, a níveis mais próximos da implementação. Nem os Diagramas de Classes, por exemplo, foram utilizados, pois já induzem, em demasia, uma implementação orientada a objetos. A forma como o sistema pode vir a ser implementado não faz parte do escopo deste trabalho.

♦ Algumas passagens do texto, ou até mesmo alguns componentes dos diagramas, são referentes aos mecanismos de decisão e negociação, que não foram aqui mostrados, pois fugiam ao propósito desse artigo, muito embora, foram implementados e testados, de forma isolada, ou seja, fora do contexto do uso de agentes. Os resultados foram animadores. Mais que isso, provocadores e serão também encaminhados para publicação à parte do restante da modelagem.

REFERÊNCIAS BIBLIOGRÁFICAS

ALBUS, J., McCAIN, H., LUMIA, R. **NASA/NBS Standard Reference Model for Telerobot Control System Architecture (NASREM)**, NBS Technical Note 1235, Robot System Division, NIST, 1987 apud.

BAILY, P., FARMER, D., JESSOP, D., JONES, D., **Compras – Princípios e Administração***, São Paulo*: Atlas, 2000.

BAUER, B., MÜLLER, J.P., ODELL, J. **Agent UML: A Formalism for Specifying Multiagent Interaction**, em **Agent-Oriented Software Engineering**, CIANCARINI, P., WOOLDRIDGE, M. eds., Berlin, Alemanha, p.91-103, 2001.

BAUER, B.**UML Class Diagrams Revisited in the Context of Agent-Based Systems**. Proceedings of Agent-Oriented Software Engineering (AOSE 2001), Montreal, Estados Unidos, 2001.

BERGENTI, F., POGGI, A. **Exploiting UML in the Design of Multi-Agent Systems**. Proceedings of ESAW Workshop at ECAI, Paris, França, p.106-113, 2000.

BRESCIANI, P., PERINI, A., GIORGINI, P. et al. **A Knowledge Level Software Engineering Methodology for Agent Oriented Programming**. Proceedings of the Fifth International Conference on Autonomous Agents (ACM AGENTS'01), Montreal, Quebec, Canadá, p.648-655, 2001.

COSSENTINO, M., CHELLA, A., LO FASO, U. **Design agent-based systems with UML**. International Symposium on Robotics e Automation, México, 2000.

FLAKE, S., GEIGER, C., KÜSTER, J.M. **Towards UML-based Analysis and Design of Multi-Agent Systems**. International Symposium on Information Science Innovations in Engineering of Natural and Artificial Systems (ENAIS 2001), Dubai, 2001.

FRANKLIN, S.; GRAESSER. A. **Is it an Agent, or just a Program? A taxonomy for Autonomous Agents**. Proceedings of the Third International Workshop on Agents Theories, Architectures, and Languages, Spring, Verlag, 1996.

GENESERETH, M.R. **Software Agents**. Communications of the ACM, Vol.37, No.7, July, 1994.

GIESE, L.F. **Estrutura de agentes para os processos de compra e venda utilizando tomada de decisão difusa**. Dissertação de Mestrado, Curso de Pós-Graduação em Ciência da Computação, UFSC, 1998.

HOFFMAN, D.L., NOVAK, T.P. **Marketing in hyper- media computer-mediated environments: conceptual foundations.** Journal of Marketing 60 (2), p.50-68, 1996 apud.

KARACAPILIDIS, N., MORAÏTIS, P. **Intelligent Agents for an Artificial Market System**. Proceedings of the Fifth International Conference on Autonomous Agents (ACM AGENTS'01), Montreal, Quebec, Canadá, pp. 592-599, 2001.

LIANG, T.P.; HUANG, J.S. **A framework for applying intelligent agents to support electronic trading**. Decision Support Systems, Vol.28, p.305-317, 2000.

LIND, J.. **Specifying Agent Interaction Protocols with Standard UML**. Proceedings of the Second International Workshop on Agent-Oriented Software Engineering (AOSE-2001). Heidelberg, Alemanha, 2002.

ODELL, J., PARUNAK, H., BAUER, B**., Representing Agent Interaction Protocols in UML**, em **Agent-Oriented Software Engineering**, CIANCARINI, P., WOOLDRIDGE, M. eds., Berlin, Alemanha, p.121-140, 2001.

SLACK, N. et al. **Administração da Produção**. Editora Atlas S.A., São Paulo, 1997.

STONE, P., VELOSO, M. Multiagent Systems**: A Survey from a Machine Learning Perspective.** *CMU CS technical report number CMU-CS-97-193, USA*, 1997 apud.

UML 1.4 Specification, disponível em http://www.omg.org/technology/documents/formal/uml.htm, visitado em março/2003.

WOOLDRIDGE, M., JENNINGS, N.R., KINNY, D. **A Methodology for Agent-Oriented Analysis and Design**. Proceedings of The Third International Conference on Autonomous Agents (ACM AGENTS'99). Seattle, Estados Unidos, P.69-76, 1999.

WOOLDRIDGE, M., JENNINGS, N.R., KINNY. **The Gaia Methodology for Agent-Oriented Analysis and Design**. Journal of Autonomous Agents and Multi-Agent Systems. 3(3), p.285-312, 2000.