Universidade Federal do Rio Grande do Norte

Centro de Ciências Exatas e da Terra

Departamento de Informática e Matemática Aplicada

Pós-Graduação em Sistemas e Computação

Aplicação de Ontologias para Métodos de Negociação de um Sistema Multiagente para o Reconhecimento de Padrões

Valéria Maria Siqueira Bezerra

Natal, Outubro de 2008

Universidade Federal do Rio Grande do Norte

Centro de Ciências Exatas e da Terra

Departamento de Informática e Matemática Aplicada

Pós-Graduação em Sistemas e Computação

APLICAÇÃO DE ONTOLOGIAS PARA MÉTODOS DE NEGOCIAÇÃO DE UM SISTEMA MULTIAGENTE PARA RECONHECIMENTO DE PADRÕES

Valéria Maria Siqueira Bezerra

Dissertação submetida ao Programa de Pós-Graduação em Sistemas e Computação do Departamento de Informática e Matemática Aplicada da Universidade Federal do Rio Grande do Norte, como parte dos requisitos necessários para obtenção do Grau de Mestre em Sistemas e Computação.

Orientadora: Profa. Dra. Anne Magály de Paula Canuto

Natal, Outubro de 2008

RESUMO

A utilização de agentes inteligentes em sistemas multi-classificadores surgiu devido à necessidade de tornar o processo de tomada de decisão de tais sistemas distribuído, autônomo e flexível. Baseado nisso, foi proposto o sistema NeurAge (*Neural Agents*) (Abreu et al 2004). Este sistema possui um desempenho superior a vários métodos de combinação centralizados (Abreu, Canuto, and Santana 2005). A negociação é importante para o desempenho de um sistema multiagente, porém a maioria das negociações são definidas de maneira informal. Um modo de formalizar as negociações é através do uso de ontologias. Dentro do contexto de classificação de padrões, o uso de ontologias fornece uma abordagem para formalizar os conceitos e regras que governam as relações entre esses conceitos.

O objetivo deste trabalho é utilizar ontologias para formalizar os métodos de negociação de um sistema multiagente para reconhecimento de padrões, mais especificamente o sistema NeurAge. Através de ontologias, pretende-se deixar o sistema NeurAge mais formal e aberto, permitindo que novos agentes possam fazer parte de tal sistema durante o processo de negociação.

Para a realização deste objetivo, o Sistema NeurAge será estudado com base em seu funcionamento e focalizando, principalmente, os métodos de negociação utilizados pelo mesmo. Na seqüência, algumas ontologias para negociação encontradas na literatura serão estudadas, e então aquelas que foram escolhidas para este trabalho serão adaptadas aos métodos de negociação utilizados no NeurAge.

ABSTRACT

The use of intelligent agents in multi-classifier systems appeared in order to making the centralized decision process of a multi-classifier system into a distributed, flexible and incremental one. Based on this, the NeurAge (Neural Agents) system (Abreu et al 2004) was proposed. This system has a superior performance to some combination-centered methods (Abreu, Canuto, and Santana 2005). The negotiation is important to the multi-agent system performance, but most of negotiations are defined informaly. A way to formalize the negotiation process is using an ontology. In the context of classification tasks, the ontology provides an approach to formalize the concepts and rules that manage the relations between these concepts.

This work aims at using ontologies to make a formal description of the negotiation methods of a multi-agent system for classification tasks, more specifically the NeurAge system. Through ontologies, we intend to make the NeurAge system more formal and open, allowing that new agents can be part of such system during the negotiation.

In this sense, the NeurAge System will be studied on the basis of its functioning and reaching, mainly, the negotiation methods used by the same ones. After that, some negotiation ontologies found in literature will be studied, and then those that were chosen for this work will be adapted to the negotiation methods used in the NeurAge.

"Talvez não tenhamos conseguido fazer o melhor, mas lutamos para que o melhor fosse feito... Não somos o que deveríamos ser, não somos o que iremos ser. Mas, graças a Deus, não somos o que éramos ...".

Martin Luther King

A minha mãe e a minha irmã.

Agradecimentos

Existem pessoas em nossas vidas que nos deixam felizes pelo simples fato de terem cruzado o nosso caminho. Algumas percorrem ao nosso lado, vendo muitas luas passarem, mas outras apenas vemos entre um passo e outro. A todas elas chamamos de amigo. Gostaria de deixar uma palavra de profunda gratidão a todos aqueles que, de uma forma direta ou indireta, contribuíram para a realização deste trabalho.

Primeiramente, agradeço a Deus que tem me acompanhado fielmente durante cada dia de minha vida e que conhece a fundo os meus pensamentos e nunca me abandona.

A Anne Magály, mais que uma professora, uma amiga com quem interagi quatro anos e com quem participei de lutas que me trouxeram cada vez mais experiência e amadurecimento.

A André Maurício e Patrícia Tedesco que gentilmente aceitaram participar e colaborar com este trabalho fazendo parte da banca.

A minha irmã Ana Raquel, pelo estímulo e apoio desde a primeira hora; pela paciência e grande amizade com que sempre me ouviu.

A minha amada mãe, pelo amor, carinho e educação. Sei que na dimensão que você se encontra, esta feliz em ver-me vencendo mais uma etapa. A você, minha eterna gratidão, meu carinho e minhas saudades.

Agradeço a Shirlly, minha irmã de coração, pela amizade, fidelidade, companheirismo e convívio (quase sempre) harmonioso, especialmente ao longo desses últimos 5 meses.

A Laura Santana pela ajuda incondicional e paciência nos momentos mais críticos no decorrer da dissertação.

Aos meus familiares que me alicerçam com o seu amor e compreendem sempre a minha ausência, por força das circunstâncias, em suas vidas.

Aos meus amigos do CIn/Motorola, que ao longo dos últimos 5 meses me ajudaram de alguma forma, seja com um simples sorriso, uma palavra amiga ou até mesmo “aturar” meus momentos de impaciência.

Aos meus amigos. Quero dizer-lhes que nesta batalha árdua e tão cheia de obstáculos, foi grande a importância de nossa amizade e que jamais chegaria ao topo sem a ajuda e a força que vocês me deram.

Reitero aqui minhas palavras, sendo este um trabalho individual e muitas vezes solitário, de que não se pode empreendê-lo sem a ajuda dos que nos cercam e querem bem. A todos, meu muito obrigada.

SUMÁRIO

1. Introdução 1

1.1. Introdução 2
1.2 Motivação e Justificativas 3
1.3 Objetivos 4
1.4. Estrutura do documento 5

2. Sistemas Multi-agente e Ontologias 6

2.1. Classificadores e Sistemas Multi-Classificadores 7
2.2. Agentes Inteligentes e Sistemas Multi-Agentes 8
2.3. Sistemas Multi-Agentes para Classificação 10
2.4.Negociação em Sistemas Multi-Agentes 10
2.4.1. Teoria Dos Jogos 12
2.4.2. Leilão 13
2.4.3. Argumentação 14
2.5. Plataforma de Desenvolvimento Multi-Agente 14
2.5.1. JADE (Java Agent DEvelopment framework) 15
2.6. Resumo do Capítulo 17

3. Sistema NeurAge 18

3.1. O Sistema NeurAge 19
3.2. Negociação Adaptada ao Reconhecimento de Padrões 21
3.2.1. Método Baseado na Teoria dos Jogos 22
3.2.1.1 Exemplo Ilustrativo do Método Baseado na Teoria dos Jogos 24
3.2.2. Leilão 245
3.2.2.1 Exemplo Ilustrativo do Método Baseado em Leilão 27
3.2.3. Argumentação 27
3.2.4. Negociação baseada em Sensibilidade 28
3.3. Resumo do Capítulo 30

4. Ontologias para Negociação 31

4.1. Ontologia 32
4.1.1 Ontologias para Negociação 33
4.2 Descrição das Ontologias para Negociação utilizadas 34
4.2.1 Ontologia para Negociação Automatizada 35
4.2.1.1 Papéis na Negociação 38
4.2.1.2 Taxonomia das Regras de Negociação 39
4.2.1.3 Protocolo Geral de Negociação 40
4.2.2 Negociação Baseada em Argumento 411
4.2.2.1. Modelo de Negociação 422
4.2.2.2 Ontologia para Negociação 433
4.2.3. Ontologia para Negociação entre Agentes de Informação 466
4.2.3.1. Objetos do Protocolo 47
4.2.3.2 Estados e Transições do Protocolo 48
4.3. Construção de Ontologias 49
4.3.1. Metodologias disponíveis para a construção de ontologias 49
4.3.2. Linguagens disponíveis para a construção de ontologias 50
4.3.3. Ferramentas disponíveis para a construção de ontologias 50
4.3.3.1 Protégé 51
4.3.4. Conclusões 51
4.4. Resumo do Capítulo 53

5. Adaptação das Ontologias aos Métodos de Negociação do Sistema NeurAge 54

5.1. Ontologias adaptadas aos métodos de negociação do NeurAge 55

5.1.1. Adaptação da Ontologia para o Método de Negociação Baseado na Teoria Dos Jogos *55*
5.1.1.1 Adaptação dos Conceitos da Ontologia 55
5.1.1.2 Adaptação dos papéis da Ontologia 56
5.1.1.3. Adaptação dos Relacionamentos entre os Conceitos da Ontologia 57
5.1.1.4. Adaptação das Regras de Negociação da Ontologia 57
5.1.1.5. Estrutura da Negociação 59
5.1.2 Adaptação da Ontologia Baseada em Argumento para o Método de Argumentação *61*
5.1.3 Adaptação da Ontologia entre Agentes de Informação ao Método de Negociação Leilão *64*
5.1.3.1 Objetos do Protocolo 65
5.1.3.2 Estados e Transições do Protocolo 66
5.1.4.Adaptação da Ontologia entre Agentes de Informação ao Método de Negociação Baseado em Sensibilidade *68*
5.1.4.1 Objetos do Protocolo 69
5.1.4.2 Estados e Transições do Protocolo 70
5.2 RESUMO DO CAPÍTULO 72

6.Validação da Ontologia com o Método de Negociação Baseado em Sensibilidade 73

6.1. MODELAGEM DA ONTOLOGIA 74
6.1.1. Protégé *74*
6.1.2. Suporte JADE para Ontologia *74*
6.2. REPRESENTAÇÃO DA ONTOLOGIA UTILIZANDO PROTÉGÉ, *BEANGENERATOR* E JADE 74
6.3 IMPLEMENTAÇÃO DOS AGENTES DO NEURAGE 76
6.3.1. A Arquitetura dos Agentes *77*
6.4. CLASSES DA NEGOCIAÇÃO 78
6.4.1 Concepts *78*
6.4.1.1. AgentAction 79
6.4.2 Predicate *79*
6.4.3 Inicialização dos agentes e da ontologia *80*
6.5. ANÁLISE DO FUNCIONAMENTO DO SISTEMA 80
6.5.1 Trabalho Experimental *81*
6.5.2 Base de dados utilizada *81*
6.5.3 Cenário ilustrativo um do método de negociação *81*
6.5.4 Cenário ilustrativo dois do método de negociação *7884*
6.4.5 Cenário ilustrativo três do método de negociação *84*
6.4.6 Resultados e Análise de Desempenho *805*
6.6. RESUMO DO CAPÍTULO 80

7. Conclusões 87

7.1. CONCLUSÃO 88
7.2. TRABALHOS FUTUROS 89

8. Referências Bibliográficas 90

LISTA DE FIGURAS

Figura 2.1: Sistema multi-classificador (Santana 2005)..... 7

Figura 3.1. Sistema multi-classificador com comunicação entre as entidades..... 19
Figura 3.2. Arquitetura Interna de um Agente Classificador. 21

Figura 4.2: Diagrama de Atividades do Protocolo 41
Figura 4.3: Metodologias, ferramentas e linguagens de ontologia..... 53

Figura 5.1: Diagrama de Classes da Ontologia adaptada a Teoria dos Jogos. 56
Figura 5.2: Diagrama de Classes da Ontologia baseada em argumento..... 62
Figura 5.3: Exemplo do processo de negociação entre dois agentes *a* e *b* 63
Figura 5.4: Diagrama de Atividades do Protocolo 64
Figura 5.5: Diagrama de Classes da Ontologia adaptada ao Leilão 65
Figura 5.6: Diagrama de Atividades do Protocolo de Negociação..... 67
Figura 5.7: Diagrama de Classes da Ontologia adaptada ao método sensibilidade 69
Figura 5.8: Diagrama de Atividades do Protocolo de Negociação do NeurAge..... 71

Figura 6.1: Taxonomia dos *Concept*, *AgentAction* e *Predicate* 75
Figura 6.2 : Definição dos métodos set/get para slots 76

Figura A.1: Implementação do conceito `Oferta` 98
Figura A.2: Declaração do método `calculaDistancia()` 98
Figura A.3: Declaração do método `setSensibilidade()` 98
Figura A.4: Chamadas dos objetos `conceitoDistancia` e `conceitoSensibilidade`..... 98
Figura A.5: Cálculo do conceito `Punicao`..... 99
Figura A.6: Criação do objeto `conceitoPunir` 99
Figura A.7: Implementação do conceito *AgentAction* `EnviarPunicao`..... 99
Figura A.8: Implementação do *Predicate* `NegociaCom`..... 99
Figura A.9: Registro da Linguagem e Ontologia utilizados pelo agente..... 99
Figura A.10: Definições de *Concepts*, AID(s), *Predicates*, *slots* e tipos dos *slots*..... 100
Figura A.11: Envio do *AgenteAction* `EnviarPunicao` 100

LISTA DE TABELAS

Tabela 3.1: Matriz de *payoff* para dois Agentes. 22
Tabela 3.2: Confiabilidades dos dois Agentes. 24
Tabela 3.3: Matriz de *Payoff* para os Agentes. 25
Tabela 3.4. Negociação utilizando o Leilão. 25
Tabela 3.5: Matriz do Leilão para os Agentes. 27

Tabela 4.1: Objetos do Protocolo da Ontologia para Negociação 47
Tabela 4.2. Operações do Protocolo da Ontologia para Negociação 48
Tabela 4.3 Estados do Protocolo de Ontologia para Negociação 48

Tabela 5.1: Objetos do Protocolo da Ontologia para Negociação 65
Tabela 5.2. Operações do Protocolo da Ontologia para Negociação 66
Tabela 5.3 Estados do Protocolo de Ontologia para Negociação 66
Tabela 5.4: Objetos do protocolo de ontologia adaptada ao método de sensibilidade ... 69
Tabela 5.5. Operações do protocolo de ontologia adaptada ao método de sensibilidade 70
Tabela 5.6 Estados do protocolo de ontologia adaptada ao método de sensibilidade 70

Tabela 6.1: Saída dos dois agentes 81
Tabela 6.2: Análise da sensibilidade dos atributos para os dois agentes 83
Tabela 6.3: Média dos atributos em cada classes 84
Tabela 6.4: Distância absoluta dos atributos do padrão de teste para sua média 84
Tabela 6.5: Ordem decrescente dos atributos em relação às suas distâncias 84

Tabela 7.6 Porcentagem de acerto do SMA ComOntologia X SemOntologia 85

Capítulo 1

Introdução

Esse capítulo contém uma introdução sobre o trabalho, os principais tópicos abordados, os principais objetivos a serem alcançados, bem como a organização estrutural do mesmo.

1.1. Introdução

A necessidade de se ter um sistema computacional que trabalhe com o reconhecimento de padrões de forma eficiente, precisa e com alta precisão motivou o interesse pelo estudo na área de métodos classificadores (Cho 2004). Uma maneira de obter o melhor desempenho de um sistema de reconhecimento, seria utilizar vários tipos de classificadores, trabalhando conjuntamente. Esta idéia deu origem à área de Sistema Multi-Classificador, também conhecidos como comitês ou *ensembles*. No entanto, começou-se a perceber que esses sistemas eram inflexíveis em determinadas situações, por utilizarem um processo de tomada de decisão centralizado. Com isso, surgiu a idéia de se estudar Sistemas Multiagentes (SMAs) para o reconhecimento de padrões. Esta idéia visa tornar o processo de tomada de decisão dos Sistemas Multi-Classificadores mais dinâmico, autônomo e distribuído, melhorando, assim, o desempenho de tais sistemas.

Baseado na idéia supracitada, Abreu et al. (2004) idealizou o NeurAge (*Neural Agents*), um Sistema Multiagente para classificação de padrões, composto por agentes classificadores neurais. Nele, os agentes possuem a capacidade de decidir sobre a qual classe um padrão de entrada pertence e negociar uma saída comum. O funcionamento do sistema pode ser descrito da seguinte maneira: uma vez que é apresentado um determinado padrão ao sistema, cada agente fornece sua resposta sobre a qual classe o padrão pertence. Então, as respostas são "discutidas" até um determinado ponto que se chegue a um consenso entre eles (Abreu 2005).

Como já mencionado, no sistema NeurAge, as diferentes saídas dos agentes serão discutidas até que um consenso seja alcançado. Ou seja, os agentes entram em um processo de negociação. De uma maneira geral, um processo de negociação visa obter um acordo, implícito ou explícito, que corresponda a um consenso entre as entidades participantes. Embora o contrato seja o objetivo principal da negociação, o conteúdo é alvo de várias alterações durante o processo de negociação e, no final, esse pode não ser o melhor contrato possível para cada uma das entidades. Contudo, o contrato pode ser suficientemente bom e vantajoso para ambas as entidades. Também, é possível que pelo menos uma das partes não considere que o contrato a satisfaça e, conseqüentemente, pode rejeitar o acordo e desistir da negociação (Wooldridge 2002).

Como pode ser percebido, a negociação tem um papel fundamental para o desempenho do sistema NeurAge. Porém, os protocolos de negociação propostos para o NeurAge são definidos de maneira informal. Em Abreu (2005) não foi utilizado um formalismo para descrever os métodos de negociação. Tais métodos foram descritos usando linguagem natural, por isso diz-se que os métodos de negociação utilizados no sistema NeurAge são informais. Um modo de formalizar tais protocolos é através do uso de ontologias.

De maneira geral, as ontologias podem atuar sobre fontes de dados, proporcionar organização e recuperação mais efetiva. As ontologias possibilitam compreensão comum e compartilhada de um domínio, onde ocorre interação entre pessoas e sistemas. As ontologias podem ser empregadas em diversas áreas de atuação, tais como: segurança de sistemas (Brandão, Martimiano, and Moreira 2004), sistemas de informações geográficas (Sousa and Leite 2005), engenharia de software (Medeiros and Belchior 2004), reconhecimento de imagens (Freitas and Torres 2005), especificação formal (Qu, Zhang and Li 2004), padrões de projeto (Ribeiro, Sousa, and Girardi 2004), web semântica (Nadarajan and Chen-Burger 2006) e Lógica Fuzzy (Tho et al 2006). Em Sistemas Multiagentes, por exemplo, as ontologias podem ser aplicadas às interações entre os agentes, incluindo a negociação. Com a utilização de ontologias, a negociação entre os agentes envolvidos nos processos (computadores ou pessoas) torna-se mais eficaz, visto que serão reduzidas as diferenças conceituais ou terminológicas. Na literatura, existem várias ontologias, inclusive algumas delas são propostas para os métodos de negociação convencionais. Os métodos utilizados no NeurAge são adaptações desses métodos, então para evitar o desenvolvimento de uma ontologia do início ao fim, surgiu a idéia de utilizar ontologias estudadas na literatura e adapta-las aos métodos de negociação usados no NeurAge.

1.2. Motivação e Justificativas

A fim de que haja a discussão entre os agentes do sistema NeurAge, é preciso a comunicação entre os agentes. Analogamente, para que eles possam trabalhar em conjunto, é necessário que existam mecanismos de negociação eficientes, daí a importância dos métodos de negociação em sistema multiagentes. Um bom método de negociação garante um bom desempenho do sistema multiagente. No sistema multiagente proposto por Abreu et al. (2004), os métodos foram propostos de maneira

informal. Uma maneira de formalizar é através do uso de ontologia, tornando o sistema mais modularizado e mais fácil de entendê-lo.

Há uma tendência de tornar o sistema NeurAge mais aberto, em que um agente pode entrar e sair durante a execução do sistema. Pois, agentes inteligentes possuem características como autonomia e proatividade, ou seja, o agente deve tomar decisões e realizar ações importantes para a conclusão de uma tarefa ou objetivo sem a necessidade da interferência do ser humano ou qualquer outra entidade. Ao entrar no sistema durante o processo de negociação, o agente compreende o processo de negociação e é capaz de se encaixar no processamento do sistema, através da formalização dos métodos de negociação utilizando ontologias.

Atualmente, na literatura existe uma quantidade considerável de pesquisas que utilizam ontologias para negociação em SMAs, no entanto, na área de SMAs para reconhecimento de padrões não foi encontrado uma ontologia para negociação.

1.3. Objetivos

Como já foi mencionado, o principal objetivo desse trabalho é utilizar ontologias para formalizar os métodos de negociação de um sistema multiagente para reconhecimento de padrões, mais especificamente o sistema NeurAge. Para tanto, é necessária a realização dos seguintes sub-objetivos:

- compreender o funcionamento geral do sistema NeurAge, abrangendo, principalmente os quatro métodos de negociação utilizados pelo mesmo, que são baseados em teoria dos jogos, leilão, argumentação e em sensibilidade, proposto por Abreu (2005).
- adaptar ontologias para negociação já existentes aos quatro métodos de negociação utilizados nos NeurAge. Existe uma quantidade razoável de ontologias disponíveis na literatura, inclusive algumas delas são propostas para os métodos de negociação convencionais. Os métodos utilizados no NeurAge são adaptações desses métodos, então evitou-se realizar todo um processo de desenvolvimento de uma ontologia do início ao fim.
- formalizar as quatro ontologias através da ferramenta Protégé (Protégé 2005). Como são métodos de negociação diferentes, os mesmos utilizam como base

ontologias diferentes. Essa ferramenta proporciona uma modelagem mais formal da ontologia.

- validar a ontologia adaptada ao método de negociação baseado em sensibilidade, para tanto é necessário exportar a sua formalização para um sistema multiagente. Em seguida, foram criados cenários para verificar se a modelagem garante que a ontologia funciona de acordo com as regras de negociação do método para qual foi adaptada.

1.4. Estrutura do documento

Este trabalho está dividido em sete capítulos.

- O primeiro é uma breve introdução sobre os temas abordados no mesmo.
- O segundo faz uma explanação da teoria que é usada no trabalho, como por exemplo, classificadores, agentes inteligentes, sistemas multi-classificadores e sistemas multiagente para reconhecimento de padrões, bem como alguns trabalhos que já foram realizados nessa área.
- O terceiro capítulo apresenta o sistema NeurAge, e em seguida, descreve os métodos de negociações utilizados entre seus agentes, além de mostrar um novo método de negociação, o baseado em sensibilidade, proposto em Abreu (2005).
- Já no quarto capítulo, alguns conceitos de ontologias são introduzidos e, na seqüência, são apresentadas as ontologias para negociação encontradas na literatura que serão utilizadas neste trabalho, descrevendo seus conceitos, objetos e relacionamentos entre eles, que serão adaptados aos métodos de negociação utilizados no sistema.
- No quinto capítulo, são apresentadas as modificações nas ontologias para negociação estudadas. Tais alterações são necessárias a fim de que as ontologias possam ser utilizadas pelos métodos de negociação.
- O sexto capítulo mostra como foi feita a validação da ontologia juntamente com o método de negociação baseado em sensibilidade. Como também é ilustrada a base de dados utilizada, os resultados e uma análise dos mesmos.
- E, por fim, no sétimo capítulo, são mostrados as conclusões e os trabalhos futuros para esse trabalho.

Capítulo 2

Sistemas Multiagente

Para que os agentes possam “chegar a um acordo”, eles devem trocar mensagens entre si e ter métricas de decisão, ou seja, devem saber se os dados obtidos são os necessários e satisfatórios para selar o acordo. Um protocolo de negociação especifica os tipos de transações que os agentes podem fazer, bem como as seqüências de ofertas e contra-ofertas que são permitidas. Neste capítulo, serão apresentados classificadores, agentes inteligentes, sistemas multi-classificadores, sistemas multiagente para reconhecimento de padrões e a negociação em SMAs

2.1. Classificadores e Sistemas Multi-Classificadores

Um sistema classificador é um sistema que em sua entrada recebe uma característica ou uma seqüência de características de um padrão e toma a decisão referente à classe a qual o padrão pertence, a partir de um conjunto pré-definido de classes. A decisão do classificador é considerada correta se ela for a mesma classe conhecida como geradora do padrão (Saranli and Demirekler 2000).

Sistemas multi-classificadores são sistemas compostos de vários tipos de classificadores. Esses classificadores recebem um dado externo e emitem uma resposta, diretamente, para um classificador especial. Este classificador especial, também chamado de combinador, através de algoritmos de combinação toma a decisão sobre qual é a melhor resposta fornecida entre todos os reconhecedores (Cho 1999). A combinação de classificadores explora a idéia de que diferentes classificadores, especialistas ou módulos de reconhecimento podem oferecer informação complementar sobre padrões a serem classificados, melhorando a efetividade do processo de reconhecimento como um todo (Canuto 2001). A Figura 2.1 ilustra a organização de um Sistema Multi-Classificador (SMC) paralelo, também conhecido como *ensemble*. Na figura, o sistema possui três métodos de classificação e um módulo combinador (Santana 2005).

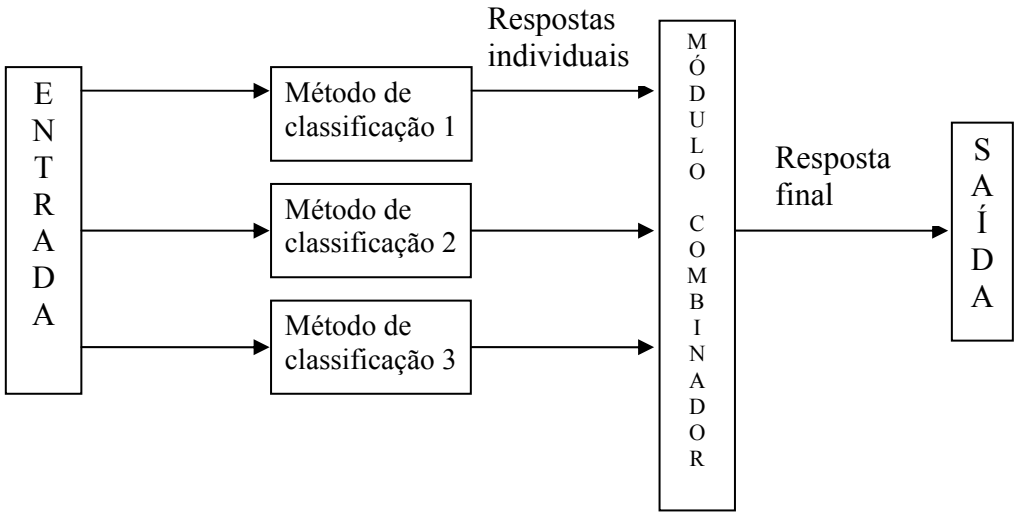

Figura 2.1: Sistema multi-classificador (Santana 2005)

Na modelagem desses sistemas, indentifica-se três passos básicos: a escolha dos componentes do sistema, a escolha da estrutura do sistema e a escolha do mecanismo de combinação que será utilizado. Uma etapa bastante importante também é a escolha do

método de reconhecimento de padrões que, geralmente, funciona utilizando redes neurais artificiais.

O método de combinação visa aproveitar as várias vantagens que os métodos individuais possuem. Porém, a existência do módulo de combinação, que necessita de todas as saídas dos classificadores para poder gerar a saída final, pode tornar o sistema um tanto inflexível, pois transforma a tomada de decisão um processo centralizado. Dessa forma, iniciaram-se estudos para tornar esse processo distribuído, evitando problemas com a adequabilidade do módulo combinador.

A combinação de classificadores tem como objetivo principal aumentar a capacidade de generalização e, consequentemente, o desempenho do sistema como um todo (Saranli and Demirekler 2001). Por esse motivo, nos últimos anos, tem sido objeto de intensas pesquisas. Contribuições têm sido dadas em uma ampla variedade de áreas, como por exemplo, reconhecimento de caracteres (Bhattacharya and Chaudhuri 2005; Bertolami and Bunke 2006), reconhecimento de voz (Schüller et al 2005; Kajarekar et al 2005), identificação de face (Czyz, Kittler, and Vandendorpe 2004; Chawla and Bowyer 2005), medicina (Li et al 2005; Li, Chan, and Krishnan 2005), indústria (Bonissone, Eklund, and Goebel 2005) e detecção de intrusos (Evangelista et al 2005).

2.2. Agentes Inteligentes e Sistemas Multiagentes

Com o objetivo de tornar os Sistemas Multi-Classificadores com tomada de decisão distribuída, melhor desempenho, mais dinâmico e por conseqüência mais eficiente, surgiu a idéia de utilizar Sistemas Multiagentes para o reconhecimento de padrões.

Definir agentes com um conceito único ou padrão é difícil, pois existem várias abordagens e ponto de vista de diferentes autores. Além disso, devido às suas mais diversas aplicações, uma definição precisa se torna cada vez mais complicada e variada. No entanto, o que se tem visto é um consenso no fato de que agentes são entidades que se adaptam a um meio, reagem a ele e provocam mudanças nesse meio. Segundo Wooldridge (2000), um agente é como um sistema de computação situado em algum ambiente e capaz de ações autônomas sobre este ambiente de forma a atingir os objetivos desejados.

Existem algumas propriedades que são essenciais para uma melhor caracterização de agente. Uma delas é a autonomia que o agente inteligente deve ter, tomar decisões e realizar ações importantes para a conclusão de uma tarefa ou objetivo, sem a necessidade da interferência do ser humano ou qualquer outra entidade. Associada à autonomia está a pró-atividade, que nada mais é do que a capacidade que o agente deve ter de tomar iniciativas. A reatividade é a capacidade de reagir rapidamente a alterações no ambiente, ou seja, perceber o meio e responder de modo oportuno. Para alcançar seus objetivos, agentes devem ter uma habilidade de comunicação a qual nada mais é que uma capacidade de se comunicar com repositórios de informações que pode ser repositório de dados, outro agente ou o próprio ambiente (Wooldridge 2000).

Por fim, deve haver uma capacidade de cooperação: agentes inteligentes podem e devem trabalhar juntos para mútuo benefício na execução de uma tarefa complexa e um comportamento adaptativo, no qual possam examinar o meio externo e adaptar suas ações para aumentar a probabilidade de ser bem-sucedido em suas metas.

Sistemas multiagentes são sistemas constituídos de múltiplos agentes que interagem ou trabalham em conjunto de forma a realizar um determinado conjunto de tarefas ou objetivos. Esses objetivos podem ser comuns a todos os agentes ou não. Os agentes dentro de um sistema multiagente podem ser heterogêneos ou homogêneos, colaborativos ou competitivos, etc. Ou seja, a definição dos tipos de agentes depende do ambiente e da finalidade da aplicação na qual o sistema multiagente está inserido (Ferber 1999).

Pode-se definir um sistema como sendo multiagentes quando ele possui determinadas características: um ambiente, um conjunto de agentes, um conjunto de objetos, as relações que podem existir entre esses agentes e os objetos ou entre os próprios agentes e um conjunto de operações que podem ser realizadas (Ferber 1999).

Os sistemas multiagentes possuem um grande universo de aplicações possíveis, dentre as mais conhecidas podem-se citar web *mining* (Hamdi 2006), agentes móveis (Rech, Oliveira, and Montez 2005), sistema de controle de tráfego (Wang 2005), comércio eletrônico (Matsubara 2005), agentes para representar indivíduos em ambientes organizacionais (Canuto et al 2005), problemas de otimização (Xie, and Liu 2005), dentre outros.

2.3. Sistemas Multiagentes para Classificação

Os trabalhos que envolvem sistemas multiagente para classificação são, basicamente, de dois tipos: aqueles que subdividem as tarefas entre os agentes, de forma que cada agente é responsável por uma tarefa diferente; e aqueles em que todos os agentes do sistema possuem a mesma tarefa.

Dentre os trabalhos do primeiro grupo, estão Heutte, Nosary, and Paquet (2004) que analisa os sistemas que fazem a leitura automática de textos escritos à mão. A plataforma é composta por três níveis: texto, palavra e grafia, em que cada um tem um grupo de agentes associado. O trabalho mostra como a capacidade do sistema de leitura se adapta ao reconhecimento de cada escrita humana. Esta adaptação é garantida por interações sobre todo o texto entre os procedimentos de reconhecimento da palavra e da letra. Outro exemplo mostrado em Vuurpijl and Schomaker (1998) é descrito um *framework* que utiliza agentes inteligentes para reconhecimento de padrões. Com ele é implementado um reconhecedor de dígito para validação de senha. A modularidade do uso de agentes provê uma resposta mais rápida, um desenvolvimento mais estratificado e um paralelismo para o sistema proposto.

As pesquisas do segundo tipo podem ser encontradas, por exemplo, em Morency, Sidner, and Darrell (2005) e Abreu, Canuto, and Souto (2005). Morency, Sidner, and Darrell (2005) que descreveu um modelo de reconhecimento visual o qual integra sugestões de um diálogo entre um agente com a observação direta da reação do usuário. Pois, reações e gestos são usados essencialmente na interação entre pessoas. O trabalho foi investigar como a informação contextual pode gerar uma reação de ações durante interações com agentes. Já em Abreu, Canuto, and Souto (2005) é apresentada uma análise entre sistemas de classificação e métodos de negociação propostos no NeurAge (Abreu et al 2004), em que cada agente do sistema implementa um método de negociação diferente, e como métodos de combinação são utilizados Soma e o Voto.

2.4.Negociação em Sistemas Multiagentes

A negociação é um processo interativo onde uma decisão conjunta é tomada por dois ou mais agentes, cada um tentando alcançar um objetivo individual. Os agentes são chamados de persuasor e persuadido; o primeiro tenta convencer o segundo a dar um objeto físico, mudar seus desejos (ou as preferências de algum deles) e executar uma

ação em seu interesse. Um processo de negociação deve conter quatro componentes diferentes (Wooldridge 2002):

- Um conjunto de negociação, que representa o espaço de propostas possíveis do agente.
- Um protocolo, utilizado pelo agente que define as propostas legais que um agente deve fazer (em função do histórico das negociações anteriores), quando o acordo foi atingido, e o objeto acordado.
- Uma coleção de estratégias, uma para cada agente, que determina as propostas que serão feitas pelos agentes. A estratégia é essencialmente o programa do agente, que define como ele se comporta durante a negociação.
- Uma regra que determina o fim da negociação e o acordo feito entre os agentes.

A negociação procede, tipicamente, em uma série de rounds, com agentes movimentando-se alternadamente ou simultaneamente para fazer propostas. Estas são definidas pela sua estratégia, projetadas pelo conjunto de negociação, e devem ser legais, como descritas pelo protocolo. Alcançado o acordo, assim como define a regra, a negociação chega ao fim com a execução do acordo.

A negociação automatizada fundamenta-se na idéia de que agentes devem usar um protocolo compartilhado para resolver objetivos que surgem na negociação. Na maioria dos cenários de negociação de SMAs, o protocolo é fixado e adotado implicitamente: assume-se que um agente participante de uma negociação conhece o protocolo e concorda com ele a priori. Entretanto, para explorar completamente o potencial de ambientes abertos como a Internet – e a *Web* Semântica – os agentes não deveriam ser forçados a comprometer-se com um único protocolo de negociação, mas deveriam ser capazes de escolher o protocolo de negociação que é mais adequado ao tipo de interação na qual eles participam (Tamma, Wooldridge and Dickinson 2002).

Para participar de um processo de negociação, os agentes precisam ter um entendimento compartilhado das regras que descrevem as condições sob as quais a interação entre os agentes acontece, os acordos que podem ser feitos e as seqüências de propostas permitidas (Lomuscio, Wooldridge and Jennings 2000). Em abordagens atuais, os agentes atingem o acordo no protocolo de negociação a utilizar antes que eles possam realmente iniciá-la, ou então, o protocolo de negociação é imposto por uma

autoridade mais alta tal como o anfitrião da negociação (responsável pela criação e pela execução das regras que governam a participação) (Bartolini and Jennings 2002).

Alguns padrões prototípicos para negociação têm sido propostos. Por exemplo, a Linguagem de Comunicação de Agentes FIPA (ACL) oferece uma quantidade de performativas que pretendem, explicitamente, suportar a negociação (Fipa.org 2002). Um exemplo é a interpretação cfp (chamada de propostas), que pretende suportar negociação compartilhada via tarefas do estilo de redes-de-contratos (Wooldridge 2002).

Em sistemas multiagentes, a persuasão é uma maneira de um agente influenciar o comportamento de outro agente. Em alguns casos, o persuadido pode ser influenciado a agir da maneira desejada pelo persuasor, como também o persuadido pode se recusar a aceitar a proposta, inicialmente, e ser convencido a mudar suas crenças, objetivos ou preferências de modo que a proposta seja aceita. Em ambos os casos, a condição mínima para a negociação são as propostas feitas pelos agentes uns aos outros. Essas propostas podem ser aceitas ou rejeitadas como é o caso no protocolo da rede do contrato (Wooldridge 2002), por exemplo. Outra forma de negociação é fazer ofertas contrárias para alterar os aspectos da proposta que são insatisfatórias, conhecida como negociação baseada em argumentos proposta em Sierra et al (1997), onde os agentes podem enviar justificações ou argumentos junto com as propostas (contrárias), indicando porque devem ser aceitos.

Existem vários métodos de negociação, os principais são baseados em: teoria dos jogos, leilão e argumentação. A seguir será feita uma breve descrição desses métodos.

2.4.1. Teoria Dos Jogos

A Teoria dos Jogos tem sido usada em sistemas multiagentes, estudando modelos matemáticos de conflitos e cooperação entre pessoas. Os modelos da teoria dos jogos são representações abstratas de situações da vida real que envolve indivíduos com diferentes objetivos ou preferências (Osborne 2003).

Os modelos abstratos da teoria dos jogos podem ser usados como uma base para os protocolos de interação entre os agentes, que podem ser modelados como jogadores. Na teoria dos jogos os jogadores são motivados pelos seus próprios objetivos, assim os

agentes devem atuar em ambientes onde estas técnicas possam ser aplicadas. O uso de técnicas de teoria dos jogos requer cálculos importantes e recursos de comunicação.

A Teoria dos Jogos é um método de negociação no qual existem algumas técnicas analíticas que auxiliam o entendimento de fenômenos que ocorrem quando acontece a tomada de decisão dos agentes. Especificando para cada agente um conjunto de possíveis ações e qual a preferência que cada ação tem no sistema. Dentre os principais conceitos utilizados, está o Equilíbrio de Nash (Wooldridge 2002). Para representar essas estratégias é usada uma tabela que mostra o custo de cada ação que um determinado agente pode tomar em relação a outro. Esta matriz é conhecida como matriz de ganho ou matriz de *payoff*.

2.4.2. Leilão

O Leilão é um protocolo de negociação, no qual uma seqüência de regras é seguida e onde se podem determinar recursos e valores a partir de uma proposta feita pelos participantes do mesmo. Os leiloes on-line são cenários simples de interação, o que facilita sua automatização e os torna populares.

De uma forma abstrata, um leilão ocorre entre um agente que é o leiloeiro e uma coleção de agentes que são os compradores. O objetivo do leilão é a venda de um bem a um dos compradores, em que o leiloeiro deseja maximizar o preço do bem, enquanto os compradores desejam minimizá-lo. Para tanto, o leiloeiro utiliza regras de encontro, explicitando quais propostas os agentes podem fazer, para alcançar o seu objetivo, enquanto os compradores utilizam uma estratégia que se adaptem às regras de encontro. Os leilões podem demorar muito tempo, como por exemplo, os de Internet que podem demorar dias para que se diga qual o agente vencedor.

A elevada aceitabilidade é explicada pela simplicidade e clareza e, se deve a: a compreensão das regras, pelo fato de serem pré-definidas e conhecidas por todos os participantes, a sua computação. Embora os leilões sejam uma forma de negociação comum em ambientes computacionais, não são imunes a problemas que podem influenciar no seu desempenho. Dentre esses, é possível citar anúncios fictícios, especulação do leiloeiro, dificuldades de alcançar um acordo, coligações entre licitantes e especulação para beneficio próprio (Fonseca 2000). Vale ressaltar a importância de explorar técnicas de segurança na negociação, com o intuito de garantir a confiabilidade e autenticidade das mensagens trocadas.

2.4.3. Argumentação

A Argumentação utiliza lógica de predicados que trabalha com diretivas que são afirmativas sobre alguma proposição. Essas diretivas são importantes para gerar a base de conhecimento que o agente possui para poder argumentar. Argumentos são expressões cujo objetivo é mudar as intenções (e conseqüentemente as ações) do ouvinte. Dentro do contexto de negociação de agentes *self-interested*, esta mudança de intenções pode tornar os agentes mais cooperativos. Há diferentes tipos de argumentos que poderiam ser usados por um agente para mudar as intenções do outro. Como por exemplo: dinheiro, promessa, apelo a promessas passadas, etc. Independente do argumento usado, o agente receptor deve avaliar o argumento e decidir se muda ou não suas intenções e ações.

Argumentação é essencial para efetuar acordos em situações não-cooperativas quando os agentes têm conhecimento incompleto sobre o outro agente ou o ambiente. Em tais situações, os agentes se comunicam através da troca de mensagens. A argumentação também é utilizada quando os agentes não têm habilidade ou tempo para fazer deduções.

A fim de negociar, efetivamente, um agente necessita de habilidade para (a) representar e manter o modelo de suas crenças, desejos, objetivos, e intenções, (b) entender crenças, desejos, objetivos, e intenções dos outros agentes, e (c) influenciar as crenças, intenções, e comportamento de outros agentes. Quando os agentes são não-colaborativos, o processo de argumentação é uma troca interativa de propostas para reduzir os conflitos e o alcance dos objetivos individuais dos agentes.

2.5. Plataforma de Desenvolvimento Multiagente

Nos últimos anos, algumas ferramentas para a criação de sistemas multiagentes têm sido desenvolvidas. Essas ferramentas vão, desde plataformas e *frameworks* para desenvolvimento de sistemas multiagentes até linguagens de programação para a elaboração de programas orientados ao paradigma de agentes.

Esse trabalho é baseado em Santana (2005), que escolheu a plataforma JADE (*Java Agent DEvelopment framework)* devido algumas características abaixo citadas, além de ser uma das plataformas mais usadas, o que facilitou o aprendizado da ferramenta.

- Plataforma distribuída de agentes;
- Ferramentas de Debugging;
- Suporte à execução de múltiplas, paralelas e concorrentes atividades de agentes;
- Ambiente de agentes que obedece aos padrões FIPA;
- Transporte de mensagens;
- Biblioteca de protocolos FIPA;
- Automação de registros;
- Serviços de nomes (*Naming Service*) em conformidade aos padrões FIPA;
- Integração - Mecanismo que permite aplicações externas carregarem agentes autônomos JADE.

Além das características acima citadas, que por si só já facilitam o desenvolvimento de sistemas multiagentes, JADE possui também algumas ferramentas úteis que simplificam a administração da plataforma de agentes e o desenvolvimento de aplicações. Essas ferramentas podem ser encontradas no pacote *jade.tools* (Santana 2005).

2.5.1. JADE (Java Agent DEvelopment framework)

JADE (JADE 2005) é um ambiente para desenvolvimento de aplicações, baseada em agentes conforme as especificações da FIPA para interoperabilidade entre sistemas multiagentes totalmente implementado em Java. Essa ferramenta foi desenvolvida e suportada pelo CSELT da Universidade de Parma na Itália. Segundo JADE (2005), o principal objetivo do JADE é simplificar e facilitar o desenvolvimento de sistemas multiagentes garantindo um padrão de interoperabilidade entre esses sistemas através de um abrangente conjunto de agentes de serviços de sistema, facilitando assim como possibilitando a comunicação entre agentes, de acordo com as especificações da FIPA: serviço de nomes (*naming service*) e páginas amarelas (*yellow-page service*), transporte de mensagens, serviços de codificação e decodificação de mensagens e uma biblioteca de protocolos de interação (padrão FIPA) pronta para ser usada.

A comunicação entre os agentes é feita via troca de mensagens. Além disso, lida com todos os aspectos que não fazem parte do agente em si e que são independentes das aplicações tais como transporte de mensagens, codificação e interpretação de mensagens e ciclo de vida dos agentes. JADE pode ser considerado como um

"middleware" de agentes que implementa um *framework* de desenvolvimento e uma plataforma de agentes. Em outras palavras, uma plataforma para desenvolvimento de agentes em conformidade com os padrões estabelecidos pela FIPA e um pacote, leia-se bibliotecas, para desenvolvimento de agentes em Java.

De acordo com Bellifemine et al (2003), JADE foi escrito em Java devido a características particulares da linguagem, particularmente, pela programação orientada a objeto em ambientes distribuídos heterogêneos. Foram desenvolvidos tanto pacotes Java com funcionalidades prontas pra uso, quanto interfaces abstratas para se adaptar de acordo com a funcionalidade da aplicação de agentes.

A Figura 2.2 mostra uma simulação feita com três agentes classificadores. Além dos agentes classificadores, podemos ver o agente *ams* (*Agent Management System*) que supervisiona o acesso e o uso da plataforma de agentes. Ele provê guia de endereços e controle de ciclo-de-vida, mantendo um diretório de identificadores de agentes (*Agent Identifier* – AID) e estados de agentes. Ele é responsável pela autenticação de agentes e pelo controle de registro. Cada agente tem que se registrar com o *ams* para obter um AID válido. Vemos também o agente *df* (*Directory Facilitator*) que provê o serviço de páginas amarelas dentro da plataforma. E o agente *RMA* (*Remote Management Agent*), que é o gerenciador remoto dos agentes, responsável pelo controle e gerenciamento da plataforma JADE (Santana 2005).

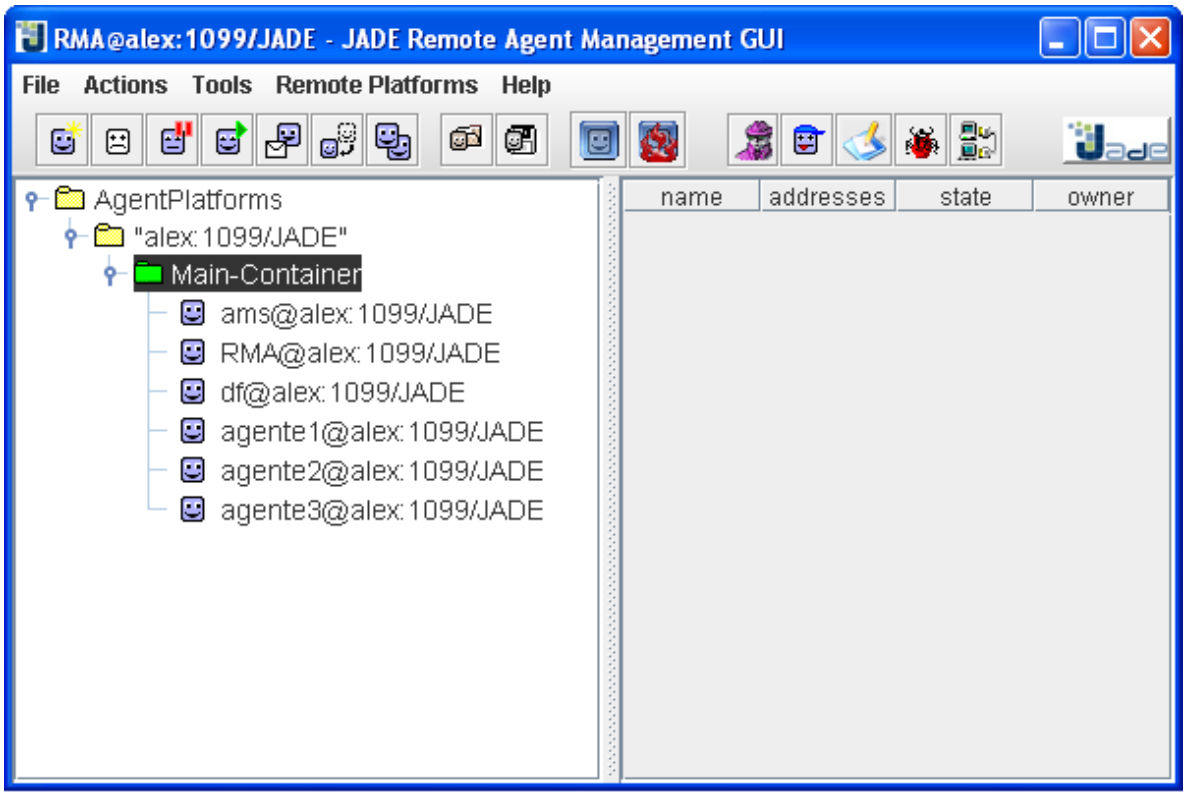

Figura 2.2: JADE Remote Agent Management GUI (Santana 2005)

O JADE suporta diversos plugins, inclusive alguns que ligam editores de ontologia para o JADE, tais como o editor Protégé (Protégé 2005), para mais detalhes ver Seção 6.1.2.

2.6. Resumo do Capítulo

O estudo realizado nesse capítulo, envolve a contextualização dos assuntos abordados no trabalho: classificadores, agentes inteligentes, sistemas multi-classificadores, sistemas multiagentes para reconhecimento de padrões e negociação nesses sistemas. Inicialmente, foram apresentadas algumas características de classificadores, sistemas multiagentes e uma breve introdução sobre negociação em sistemas multiagentes. Em seguida, foi descrita a plataforma multiagente JADE que foi utilizada por Santana (2005) para implementar o sistema NeurAge.

Capítulo 3

Sistema NeurAge

Nesse capitulo é mostrado o Sistema NeurAge, e também três dos principais métodos de negociação abordados, além do novo método proposto por Abreu(2005). Os agentes que compõem o sistema NeurAge utilizam os seguintes métodos de negociação: Teoria dos Jogos, Leilão, Argumentação e Negociação baseado em Sensibilidade.

3.1. O Sistema NeurAge

O sistema multi-agente desenvolvido é composto por um conjunto de agentes classificadores que deverão receber um padrão de entrada e gerar como saída uma resposta sobre a que classe o padrão pertence. Além disso, os agentes devem ser capazes de negociar a fim de chegar a um consenso sobre a resposta final do sistema. Cada entidade da Figura 3.1 pode ser vista e modelada como se fosse um agente com todas as suas características de coordenação de tarefas, cooperação e negociação (Abreu 2005).

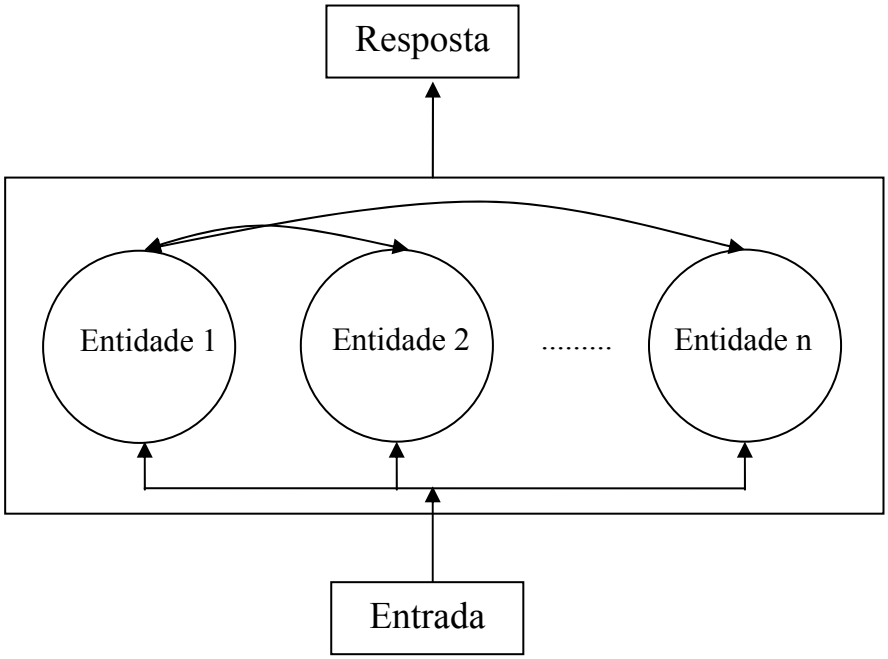

Figura 3.1. Sistema multi-classificador com comunicação entre as entidades.

A Figura 3.2, adaptada de Abreu et al (2004), ilustra a arquitetura interna do agente classificador. Como o objetivo principal de todos os agentes é o mesmo, a estrutura geral de todos os agentes é a mesma. Sua arquitetura é composta por quatro módulos principais: módulo controlador; módulo de tomada de decisão; módulo de negociação e módulo da rede neural, nesse caso, constituído de redes *Multi-Layer Perceptron* (MLP) (Haykin 1998). Esses módulos podem ser descritos como segue:

- Controlador: este módulo é responsável por receber as perguntas do usuário, assim como, para definir a ordem da ativação de seus processos internos.
- Tomada de Decisão: este módulo é responsável por raciocinar sobre seu conhecimento a fim de definir a melhor saída para um classificador da rede

neural. A idéia principal deste módulo é procurar por um resultado, eliminando aqueles que não cabem as circunstâncias existentes.

- Negociação: este módulo é responsável para a comunicação com outros agentes a fim de alcançar um resultado comum. Constrói-se um plano de ação para o protocolo da negociação ou usam-se os planos de uma ação feita previamente. Durante o processo da negociação, pode-se sugerir que um agente mude seu resultado. Entretanto, tem-se a autonomia para decidir mudar ou confirmar seu resultado atual.
- Rede Neural: este módulo é responsável por executar o método da rede neural do agente.

A idéia principal, atrás do funcionamento de um agente classificador, é que uma vez que a pergunta de usuário é fornecida, o controlador passa as informações necessárias ao módulo de tomada de decisão, que acessa a rede neural produzindo uma saída, usando a informação contida no domínio da base de conhecimento. Então, o controlador passa os resultados para o módulo de negociação onde ocorre a comunicação com os outros agentes a fim de alcançar um resultado comum. Durante o processo da negociação, pode ser necessária que o agente mude de opinião sobre o resultado corrente (Abreu et al 2004). O processo de mudar um resultado é possível porque o módulo da rede neural fornece uma lista com as possíveis saídas ordenadas segundo um critério de avaliação. Uma vez que um resultado comum é alcançado, fornece-se ao usuário.

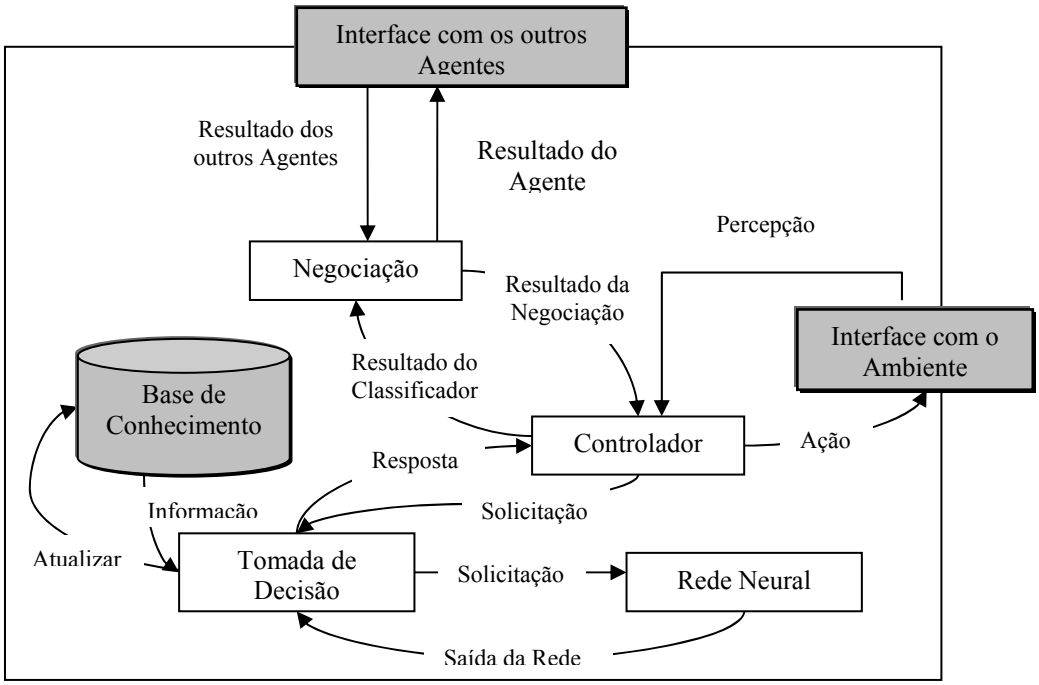

Figura 3.2. Arquitetura Interna de um Agente Classificador.

A próxima Seção descreve os métodos de negociação utilizados no NeurAge, para os quais serão adaptadas ontologias para negociação encontradas na literatura, para a tarefa de negociação no sistema de reconhecimento de padrões.

3.2. Negociação Adaptada ao Reconhecimento de Padrões

A negociação para agentes classificadores necessita de alguns ajustes em sua composição para que seja possível trabalhar com a informação que esses agentes fornecem (Abreu 2005).

Dentre os principais conceitos está a confiabilidade, que trata da certeza de que um agente tem sobre a classe que ele diz ser a correta. Para todas as saídas possíveis um agente possui uma certeza, porém a classe vencedora é aquela na qual a certeza é o maior valor. Normalmente usa-se o valor dessa certeza como sendo um número entre zero e um.

É importante ressaltar que só irá acontecer negociação, se os agentes discordarem entre si de acordo com as classes, se todos afirmarem que a classe vencedora é a mesma, simplesmente passa-se para o próximo padrão.

Outro conceito significante é o de classe original, que é o valor da confiabilidade correspondente à classe que o agente o qual está sendo avaliado, considera como sendo

a sua vencedora, e classe mudada, que é o valor da confiabilidade correspondente à classe que o outro agente que está sendo analisado, considera como sendo a sua vencedora. Por exemplo, se o agente 1 acha que a classe A é a vencedora e o agente 2 acha que é a classe D, então, para o agente 1, a classe original é a confiabilidade correspondente a classe A e a classe mudada é a confiabilidade correspondente à classe D, da mesma forma, para o agente 2, a classe original é a confiabilidade correspondente a classe D e a classe mudada é a confiabilidade correspondente a classe A.

3.2.1. Método Baseado na Teoria dos Jogos

Na teoria dos jogos, um agente escolhe seu plano de ação e o do seu adversário apenas uma vez. A fim de ajudar os agentes a tomar suas decisões, é usada uma matriz de *payoff*, que mostra o custo de cada ação que um determinado agente pode tomar em relação a outro,onde cada célula representa os valores payoff que os jogadores terão no caso dessas ações serem escolhidas.

		Agente 1	
		Muda	Não Muda
Agente 2	Muda	Muda_ação1, Muda_ação2	NãoMuda_ação1, Muda_ação 2
	Não Muda	Muda_ação1, NãoMuda_ação2	NãoMuda_ação 1, NãoMuda_ação 2

Tabela 3.1: Matriz de *payoff* para dois Agentes.

A Tabela 3.1 mostra as quatro possibilidades que o sistema possui, os dois agentes mudarem de opinião ou não mudarem e um mudar e o outro não mudar. A escolha será a melhor para ambos, ou seja, a que ambos tiverem a menor perda. Analisando a célula $C_{2,1}$ da Tabela 3.1 - (Muda_ação1, NãoMuda_ação2) - significa dizer que o agente 1 deve mudar sua classe vencedora para a classe escolhida pelo agente 2, enquanto que o agente 2 não deve mudar sua classe vencedora (NãoMuda_ação2).

Para que a negociação baseada na Teoria dos Jogos seja adaptada ao Reconhecimento de Padrões é preciso definir medidas que serão utilizadas para definir os novos valores da matriz de *payoff* mostrada na Tabela 3.1 (Abreu 2005).

Baseado nisto, a estratégia de teoria dos jogos tem sido ajustada para ser usada no NeurAge. Neste sentido, todos os agentes do NeurAge sempre terão duas possíveis ações, que são: manter a classe escolhida(manter) ou mudar para a classe escolhida pelo

outro agente (mudar). Isto também é importante para definir a medida de *payoff* para preencher a matriz, que é baseada na confiança da classe escolhida (vencedor) de um agente e da classe a ser mudada (a classe escolhida do outro agente). Existem várias maneiras de se calcular a nova confiabilidade, as formas utilizadas foram 1 e 2. Quando um agente escolhe mudar sua classe, sua confiabilidade pode ser calculada como

$$Confiabilidade\_Caso\_Mude = \left(\frac{ClasseOriginal + ClasseMudada}{2} \right) \qquad \textbf{(1)}$$

Por outro lado, quando um agente resolve manter a classe escolhida, o novo valor da sua confiabilidade é dado por:

$$Confiabilidade\_Caso\_Nao\_Mude = ClasseOriginal - ClasseMudada \qquad \textbf{(2)}$$

A negociação só acontece entre dois agentes por vez. Os agentes recebem um padrão de teste igual, cada um diz qual a classe vencedora e qual a confiabilidade para essa classe. Esses valores são armazenados juntamente com todas as saídas de todas as classes para cada agente. Geralmente, os agentes são classificados em uma ordem decrescente de confiança e os dois primeiros são escolhidos para iniciar a negociação. Esses agentes entram em negociação, calculando suas confiabilidades para mudar e para não mudar. É escolhida a célula da matriz de custo na qual a solução seja melhor (maior valor) para ambos serem o seu novo valor de confiabilidade.

Isso é definido como um round e o vencedor é, então, comparado ao terceiro agente num segundo round. Esse processo continua até existir apenas um agente. Se ambos os agentes escolherem diferentes ações, o agente que escolheu mudar a ação é descartado. O agente que manteve a ação é considerado o vencedor e continua no processo de negociação. Porém, sua confiabilidade é substituída pelo valor da matriz de *payoff*.

Se ambos os agentes escolhem a mesma ação, ambos continuam no processo de negociação, mas seus valores de confiabilidades são trocados pelos valores de *payoff*. Em todos os casos, a mudança no valor de confiabilidade poderá modificar o valor da saída da classe escolhida pelo agente. Neste sentido, o agente decide manter a classe escolhida como vencedor ou mudar a classe vencedora.

Na arquitetura interna dos agentes que usam esse método de negociação, a base de conhecimento compartilhada contém as informações sobre mudar e manter os valores dos agentes para os rounds atuais e anteriores. Por outro lado, a base de conhecimento do domínio tem informação sobre a confiabilidade do agente, a nova confiabilidade e

um *threshold*, que é individual para cada agente. Com um *threshold*, um agente pode decidir, desistir do processo de negociação no caso de sua confiabilidade ser menor que seu *threshold*. Um agente pode decidir mudar, por ele mesmo, a classe vencedora. Como já foi mencionado, essa decisão pode ser baseada em regras internas e é individual para cada agente, dependendo da experiência passada. Por exemplo, informação do vencedor passado ou negociações mais baixas de uma classe particular pode ser usada para ajustar o *threshold* da classe, tornar o agente mais flexível, ou não mudar a classe vencedora (Abreu 2005).

3.2.1.1 Exemplo Ilustrativo do Método Baseado na Teoria dos Jogos

Como exemplo para ilustrar a operação desse método, é considerada uma tarefa de reconhecimento de três classes (A, B e C) no qual os padrões contêm cinco atributos (at1, at2, at3, at4 e at5) . O sistema é composto por dois Agentes (Ag1, Ag2). Após o treinamento da rede neural, o seguinte padrão de teste é apresentado aos agentes: 0,7; 0,4; 0,34; 0,9; 0,22. O módulo de rede neural de cada agente produz suas respectivas saídas. A Tabela 3.2 mostra a saída de cada um dos agentes.

Agent 1 (Ag1)		**Agent 2 (Ag2)**	
Classe	Confiabilidade	Classe	Confiabilidade
A	0,90	B	0,87
B	0,30	A	0,56
C	0,25	C	0,34

Tabela 3.2: Confiabilidades dos dois Agentes.

Nesse caso, a classe escolhida pelo Ag1 é A, por ter obtido a maior confiabilidade, seguido por B e C. A classe escolhida pelo Ag2 é B, seguida por A e C. Como já mencionado, a matriz de *payoff* é calculada para cada ação dos agentes. Logo, o valor de *payoff* de manter para o Ag1 = 0,9 - 0,3 = 0,6 e para o Ag2 = 0,87 - 0,56 = 0,31. Também, o *payoff* de mudar é Ag1 = ((0,9 + 0,3)/2) = 0,6 e para Ag2 = ((0,87 + 0,56)/2) = 0,71. Assim, a Tabela 3.3 define a matriz de *payoff*. Em cada célula, são apresentados dois valores que representam as novas confiabilidades para ambos os agentes. Como pode ser visto na Tabela 3.3, o Ag1 pode escolher ambas as ações, mas para o Ag2 a melhor ação é mudar de classe. Sendo assim, a melhor escolha para ambos é o Ag1 manter a classe e o Ag2 mudar de classe. Como ambos os agentes escolheram ações diferentes, Ag2 é descartado e Ag1 continua na negociação.

		Agente1 (Ag1)	
		mudar	manter
Agente 2 (Ag2)	mudar	0,6; 0,71	0,6; 0,71
	manter	0,6; 0,31	0,6; 0,31

Tabela 3.3: Matriz de *Payoff* para os Agentes.

3.2.2. Leilão

Diferentemente do Leilão, mostrado na sessão 2.4.2, para o reconhecimento de padrões, todos os agentes são de todos os tipos: vendedores e compradores. Neste caso, foram utilizadas algumas funções para calcular os valores das novas confiabilidades, mostradas na Tabela 3.4 retirada de Abreu (2005).

	Ag1	Ag2	Ag3	Ag4
Ag1	$conf_j = \left(\frac{\sum p_i, i = 2,3,4}{cons\ \tan te} \right)$	$p_j = conf_i - said_i(cl_j)$	$p_j = conf_i - said_i(cl_j)$	$p_j = conf_i - said_i(cl_j)$
Ag2	$p_j = conf_i - said_i(cl_j)$	$conf_j = \left(\frac{\sum p_i, i = 1,3,4}{cons\ \tan te} \right)$	$p_j = conf_i - said_i(cl_j)$	$p_j = conf_i - said_i(cl_j)$
Ag3	$p_j = conf_i - said_i(cl_j)$	$p_j = conf_i - said_i(cl_j)$	$conf_j = \left(\frac{\sum p_i, i = 1,2,4}{cons\ \tan te} \right)$	$p_j = conf_i - said_i(cl_j)$
Ag4	$p_j = conf_i - said_i(cl_j)$	$p_j = conf_i - said_i(cl_j)$	$p_j = conf_i - said_i(cl_j)$	$conf_j = \left(\frac{\sum p_i, i = 1,2,3}{cons\ \tan te} \right)$

Tabela 3.4. Negociação utilizando o Leilão.

Na Tabela 3.4 está representado um leilão com quatro agentes, porém isso pode ser aplicado a N agentes. A diagonal principal representa as novas confiabilidades do agente da linha correspondente, que é calculado por :

$$conf_j = \left(\frac{\sum p_i, i = 2,3,4}{cons\ \tan te} \right) \quad \textbf{(3)}$$

Onde: i = Ag da linha correspondente.

j = Ag da coluna correspondente

$conf_j$ = nova confiabilidade do agente correspondente a linha-coluna

constante = valor determinado através de características do problema

Por exemplo, o cálculo da célula $C_{1,1}$, é através de:

$$Conf\_Ag1 = \left(\frac{p\_Ag2 + p\_Ag3 + p\_Ag4}{cons\,\tan te} \right)$$

As demais linhas representam o cálculo da nova confiabilidade do agente da linha correspondente:

$$p_j = conf_i - said_i(cl_j) \quad \textbf{(4)}$$

Onde: i = Ag da linha correspondente.

j = Ag da coluna correspondente

$conf_i$ = confiabilidade do agente correspondente a linha

$said_i$ = valor de Ag_i em relação a classe que Ag_j escolheu como correta.

cl_j = classe correspondente ao Ag coluna.

A célula $C_{1,3}$ é calculada pela diferença entre a classe escolhida para o Ag_1 e a classe escolhida do Ag_3, por exemplo:

$$p\_Ag_3 = conf\_Ag_1 - said\_Ag_1(idven\_Ag_3)$$

Como no método anterior, este método também usa a medida de confiabilidade como base para seu funcionamento. Inicialmente, o padrão de teste é enviado a todos os agentes classificadores, em seguida cada um indica a classe vencedora e sua confiabilidade e, então, obtem-se todas as saídas dos agentes classificadores. Uma vez tendo iniciado o leilão, é calculado o custo de cada agente. Esse custo é baseado na soma de diferenças entre a confiabilidade do vencedor e as confiabilidades relacionadas a classes escolhidas dos outros agentes. Esta soma é dividida por uma constante. O agente considerado como perdedor é aquele que obtiver o custo mais alto. A confiabilidade da classe escolhida dos agentes é substituída pela diferença entre a confiabilidade e seu custo. Ao perder duas vezes seguidas, o agente é eliminado da negociação. O agente que permanecer até o fim é considerado o vencedor e a classe escolhida desse agente é vista como a classe geral escolhida.

A arquitetura interna dos agentes que usam este método de negociação é similar à anterior, no qual a base de conhecimento compartilhado contém as informações sobre os valores do custo dos agentes para os rounds atuais e anteriores. Por outro lado, a base de conhecimento do domínio tem informação sobre a confiabilidade do agente, a nova confiabilidade e um *threshold*, que é individual para cada agente. Como no método

anterior, um agente pode decidir mudar ou manter a classe vencedora. Como também pode decidir encerrar o processo de negociação após cada round, baseado em regras internas.

3.2.2.1. Exemplo Ilustrativo do Método Baseado em Leilão

Usando o mesmo exemplo da Seção 3.2.1.1 e Tabela 3.2, um terceiro agente pode ser acrescentado ao sistema. Baseado na Tabela 3.4 e na confiabilidade do terceiro agente (C = 0,85; B = 0,4; A = 0,37), os valores de custo para os três agentes foi calculado e pode ser visto na diagonal principal da Tabela 3.5.

	Agente 1	**Agente 2**	**Agente 3**
Ag1	0,125	0,9 – 0,3 = 0,6	0,9 – 0,25 = 0,65
Ag2	0,87 – 0,56 = 0,31	0,084	0,87 – 0,34 = 0,53
Ag3	0,85 – 0,37 = 0,48	0,85 -0,4 = 0,45	0,093

Tabela 3.5: Matriz do Leilão para os Agentes.

O valor de custo do Ag1, por exemplo, foi calculado somando as células $C_{1,2}$ (0,6) e $C_{1,3}$ (0,65) divido por 10 (constante) (1,25/10 = 0,125). As células $C_{1,2}$ e $C_{1,3}$ são caluladas pela diferença entre a classe escolhida para o Ag1 e a classe escolhida do agente correspondente da coluna. Nesse exemplo, Ag1 é considerado o perdedor e a nova confiabilidade para a classe escolhida será: Ag1 = 0,775; Ag2 = 0,786 e Ag3 = 0,757.

Como no método anterior, uma vez que a confiabilidade dos agentes é mudada, eles devem escolher uma classe diferente ou manter a mesma. Nesse exemplo, mesmo com a nova confiabilidade para classe A do Ag1, a mesma continua como a vencedora do sistema. Esse processo continua até que só exista um agente no processo de negociação.

3.2.3. Argumentação

Na argumentação, os agentes possuem a capacidade de calcular o custo que ele terá se for mudar de classe da mesma maneira que já foi explicado anteriormente. Esse custo é dado por Abreu (2005):

$$Custo = \left(\frac{ClasseOriginal + ClasseMudada}{2} \right) \qquad \textbf{(5)}$$

Através dessa função custo, o agente terá fundamentos para construir a sua argumentação de forma segura e forte. Essa construção é baseada na lógica de predicados e é formada por sentenças que representam o desejo, a crença e a intenção do agente – arquitetura BDI.

Inicialmente, os agentes recebem o padrão de teste e calculam as suas saídas para cada uma das possíveis classes de resposta. Após isso, é feita uma ordenação dos seus resultados de forma decrescente. Argumentação sempre ocorre entre os agentes de menor confiabilidade e, no caso de existirem agentes com a mesma classe, negocia aquele com a maior confiabilidade. O agente que possuir a menor confiabilidade, inicia a argumentação. Quando a afirmação de um agente1 derrubar a argumentação de um agente2, o processo é encerrado e agente1 é o vencedor.

3.2.4. Negociação baseada em Sensibilidade

Este protocolo de negociação proposto em Abreu (2005) é baseado na Teoria Dos Jogos, porém, leva em consideração mais detalhes no momento da negociação. Este método também é composto por vários rounds e o agente com a confiabilidade mais alta no último round é escolhido o mais adequado para classificar o padrão de entrada. Diferente dos métodos anteriores, o decremento da confiabilidade dos agentes é baseado na análise de sensibilidade.

A sensibilidade é uma medida que indica a forma como um agente reage a pequenas mudanças nos atributos de entrada. O cálculo é feito a partir de variações nos atributos de um padrão não utilizado no treinamento, e verificando a reação do agente a essas variações. Se um agente tiver uma sensibilidade alta para um determinado atributo, significa que ele é muito susceptível a pequenas variações nesse atributo, caso contrário, ele é pouco susceptível a essas variações (Abreu 2005). Essa análise pode ser feita excluindo e/ou variando os valores de um atributo de entrada e analisando a variação do desempenho do método de rede neural. Esta análise é realizada para os atributos de todas as classes do problema de classificação.

O principal objetivo desta análise é investigar a sensibilidade de uma rede neural para um certo atributo e usá-lo no processo de negociação. Junto com a análise de sensibilidade, informações como a distância de um atributo do padrão de entrada e a média de treinamento deste atributo são também usados para definir o decremento a ser sugerido para o agente.

Todos os agentes devem permitir que suas redes neurais sejam treinadas e negociar um resultado comum para o padrão de teste. Isso pode ser feito como segue.

1. Apresenta-se um padrão de teste, produzindo como saída: a confiabilidade para cada classe e a classe vencedora (aquela com maior confiabilidade).
2. Calcula-se:
   a. a análise de sensibilidade para todos os atributos do padrão testado, variando-se o valor de cada atributo, um de cada vez.
   b. a média dos valores de todos os atributos em cada classe.
   c. a diferença dos atributos do padrão de entrada em relação a sua média para a classe vencedora, anteriormente calculada.
   d. a ordenação dos atributos em ordem decrescente da diferença calculada anteriormente (do atributo menos similar ao mais similar) por classe.
3. Os agentes verificam se seus resultados são divergentes. Se todos os agentes fornecem a mesma resposta, não há negociação e o processo é encerrado.
4. Caso contrário, começa o processo de negociação tentando mostrar para os outros agentes que seus resultados não são tão bons. A melhor forma para convencer os outros agentes é diminuindo sua confiabilidade (grau de certeza). O que pode ser feito da seguinte forma:
   a. do primeiro ao último atributo (De acordo com a ordenação feita em 2d):
      i. escolhe um agente para começar a negociar;
      ii. envia uma mensagem ao agente sugerindo uma diminuição (punição) na sua confiabilidade.

      É importante enfatizar que uma vez que um agente recebe uma mensagem de punição, ele também fará o mesmo com outro agente. Quando todos os agentes enviam mensagens de punição, diz-se que houve um round. Esse processo se repete até que todos os atributos sejam analisados ou que um usuário especifique uma quantidade de rounds.
5. Depois do processo de negociação, o agente classificador com o mais alto grau de certeza em sua resposta é dito ser o mais capacitado para classificar o padrão e a sua saída é considerada a saída correta.

A idéia principal do processo é quanto mais distante o atributo da sua média de treinamento e quanto mais sensível o atributo for a pequenas variações, maior é a

probabilidade da rede neural classificar erroneamente o padrão de entrada. Neste sentido, esse fato é usado para sugerir um decremento no grau de confiabilidade da classe vencedora de um agente. O valor de punição é calculado como segue:

$$Punicao = \frac{D * S}{R * C} \quad \textbf{(6)}$$

Onde:

D: A diferença entre o atributo corrente do padrão de teste e sua média;

S: A sensibilidade do classificador para os atributos correspondentes da classe vencedora;

R: A ordem dos atributos em relação à diferença D;

C: Constante que define a intensidade da punição, que no trabalho usou-se com o valor 20.

No caso do agente mudar a classe vencedora, quando a confiabilidade da classe vencedora se torna menor que a confiabilidade de outras classes, o agente pode decidir, baseado em experiências passadas e regras internas, escolher a classe subseqüente ou manter a classe vencedora. Um agente também pode decidir realizar um novo processo de tomada de decisão ou retreinar sua rede neural. Isto é baseado na analise de desempenho do agente, se o desempenho do agente for ou não adequado para a tarefa de classificação.

Na arquitetura interna dos agentes, as informações relacionadas à análise de sensibilidade, confiabilidade e média de treinamento dos agentescompõem a base de conhecimento compartilhada. Por outro lado, a informação sobre métodos de negociação anteriores, *thresholds*, regras internas e tudo que é individual para cada agente compõem a base de conhecimento do domínio.

3.3. Resumo do Capítulo

Nesse capítulo foi ilustrado o Sistema NeurAge, bem como três dos principais métodos de negociação abordados, além do novo método baseado em Teoria do Jogos proposto por Abreu (2005). Os métodos de negociação utilizados no sistema NeurAge são: Teoria dos Jogos, Leilão, Argumentação e Negociação baseado em Sensibilidade.

Capítulo 4

Ontologias para Negociação

Neste capítulo, são apresentadas alguns conceitos sobre ontologias, bem como as ontologias para negociação encontradas na literatura, focalizando nas negociações que serão tomadas como base para o principal objetivo desse trabalho que é adaptar ontologias existentes aos quatro métodos de negociação utilizados no sistema NeurAge. Ao final desse capitulo, é mostrado as metodologias, linguagens e ferramentas disponíveis para a construção de ontologias.

4.1. Ontologia

O termo Ontologia tem sido usado há alguns anos pelas comunidades de Inteligência Artificial e de Representação do Conhecimento. Recentemente, o uso de ontologia tem sido ampliado e popularizado como uma poderosa via de atingir a interoperabilidade de sistemas de informação, principalmente no que se refere a normalização terminológica. A possibilidade de compartilhamento do conhecimento foi a principal razão da utilização de ontologias no campo da Inteligência Artificial nos últimos anos. Ela é mais que um vocabulário padrão, pois assegura que os termos escolhidos sejam suficientes para especificar e definir conceitos, e, permitir relacionamentos adequados a partir da escolha terminológica realizada. É mais que uma taxonomia, pois inclui a expressão exata do domínio específico do conhecimento (Gruber 2003).

As técnicas para construção de ontologias têm se desenvolvido por iniciativas de diversas áreas. Muitas comunidades científicas utilizam ontologias.

- A comunidade de Inteligência Artificial, é focada nos sistemas baseados em conhecimento, a ontologia é a forma de capturar o domínio do conhecimento e posteriormente, o próprio conhecimento;
- A comunidade de linguagem natural usa ontologia para caracterizar o sentido e o significado das palavras;
- A comunidade de Banco de Dados usa ontologia para esquemas conceituais, para buscar a interoperabilidade semântica entre base de dados heterogêneos;
- Os métodos orientados a objetos usam ontologia para especificar softwares;
- Para sistemas de recuperação de informação, a ontologia especifica o significado dos conceitos a serem procurados;
- Mais recentemente, XML começa a ser usado para recuperação de informação e a ontologia auxilia na definição de meta-dados.

Ao longo dos anos, a indústria e alguns pesquisadores perceberam a importância da interoperabilidade entre diversas fontes de informação através da modelagem explícita dos conceitos usados na comunicação. Surgiu então a Web Semântica, com o objetivo

de tornar mais eficiente o processo dos dados trocados. A ontologia é um fator chave para permitir a interoperabilidade na Web Semântica (Berners-Lee et al 2001).

A pesquisa no campo de ontologias tem crescido nos últimos anos e, um dos motivos é a necessidade de dois ou mais agentes se comunicarem em um certo ambiente. Para tanto é preciso que os agentes utilizem uma terminologia comum que descreva tal domínio. Ontologia é um conjunto de especificações explícitas de significado, conceitos e relacionamentos aplicáveis a algum domínio específico (Gruber 2003). Desta forma, pode-se assegurar que dois agentes estejam utilizando a mesma linguagem durante o processo de comunicação. Uma ontologia define os conceitos que descrevem um determinado domínio, ou seja, é essencialmente uma hierarquia de conceitos de classes e subclasses com definições dos relacionamentos entre eles. Em outras palavras, a utilização de uma ontologia permite a definição de um contexto único, eliminando-se a ambigüidade.

Pragmaticamente, uma ontologia comum define o vocabulário com o qual perguntas e respostas serão trocadas entre agentes. É interessante notar que agentes podem compartilhar um mesmo vocabulário sem possuírem o mesmo conhecimento: cada um possui conhecimentos que os outros não têm e não se espera que um agente o qual se comprometa com uma ontologia seja capaz de responder a todas as perguntas que possam ser formuladas com o vocabulário compartilhado (Gruber 2003).

4.1.1 Ontologias para Negociação

A ontologia para negociação é um conceito novo, portanto as pesquisas sobre o assunto são poucas. Entretanto, está se tornando um elemento crucial na tecnologia de agentes, devido a suas características como autonomia e independência. Parte desta independência é a habilidade de requisitar ajuda a outros agentes na execução de uma tarefa (tal como encontrar informações na web). Os agentes requisitados para a ajuda podem ser um usuário ou uma organização diferente, porém não há nenhuma garantia que as entidades usarão a mesma terminologia ou compreenderão os mesmos conceitos (objetos, operadores, teoremas, regras) que o agente solicitante. Então, surgiu a área de ontologia para negociação a fim de que os agentes possam reconhecer e resolver os conflitos de ontologia (Bailin and Truszkowski 2001).

O objetivo da ontologia para negociação é permitir a cooperação entre agentes inteligentes para executarem uma tarefa, mesmo se seu conhecimento do domínio é

baseado em ontologias diferentes. Para esta finalidade, as abordagens práticas de ontologia para negociação, fornecem, geralmente um protocolo de ontologia para negociação e uma infra-estrutura do software a fim de suportar as tarefas da negociação. (Bailin and Truszkowski 2002).

Como já foi mencionado, foram encontradas diversas ontologias para negociação, esta seção apresenta algumas delas. Com o intuito de melhorar a interoperabilidade semântica nos SMAs, Su, Matskin and Rao (2003) usou uma ontologia explicativa compartilhada entre os agentes. Os papéis, no protocolo, são executados por um ou mais agentes, dentre eles o *Explanation Manager*, agente intermediário da explicação, que gerencia o protocolo de interação inicializando e finalizando o processo e enviando as mensagens aos outros agentes. Já Meng et al (2004) propõe uma nova arquitetura que usa NLP (Processamento de Linguagem Natural), recuperação de conhecimento e ontologia para classificar a oferta e demanda, e dividi-las em subconjuntos para aumentar a probabilidade de combiná-las. A arquitetura inclue cinco partes: *Business Spy Agent, Supply and Demand Analysis, Supply and Demand Classification, Matching and Negotiation*, e *Hidden Business Mining*. A transação comercial suportada pela arquitetura abrange quatro processos, tais como: *supply and demand identification, product brokering, merchant brokering* e *negotiation*.

Em Carabelea (2002), é aresentado uma generalização de um sistema multiagente em que os melhores acordos de cooperação são alcançados com o uso de argumentos usados na interação humana. A negociação é executada, usando tipos diferentes de argumentos que variam de quantitativos, tais como dinheiro ou objetos de comércio, aos argumentos qualitativos como promessas, recursos a promessas passadas e exemplos passados. Os modelos dos outros agentes são construídos e refinados, incrementalmente, durante a negociação. Esses modelos são usados para adaptar a estratégia da negociação de acordo com os desejos de outros agentes, preferências e características comportamentais durante as interações.

4.2 Descrição das Ontologias para Negociação utilizadas

Na literatura, existem várias ontologias, inclusive algumas delas são propostas para os métodos de negociação convencionais. Os métodos utilizados no NeurAge são adaptações desses métodos, então para evitar o desenvolvimento de uma ontologia do

início ao fim, surgiu a idéia de utilizar ontologias estudadas na literatura e adapta-las aos métodos de negociação usados no NeurAge. As três ontologias escolhidas para serem adaptadas aos quatro métodos de negociação são aquelas propostas em: Tamma, Wooldridge and Dickinson (2002), Sierra et al (1997) e Bailin and Truszkowski (2001). As Seções seguintes descrevem as ontologias abordadas.

4.2.1 Ontologia para Negociação Automatizada

A ontologia de negociação é construída com o intuito de encontrar aspectos comuns entre protocolos de negociação diferentes (Tamma, Wooldridge and Dickinson 2002). A classificação de London (Field, Stolze and Stroebel 2000) e o trabalho de Lomuscio, Wooldridge and Jennings (2000) ofereceram os atributos usados para descrever os conceitos. A estrutura da ontologia foi inspirada em Wurman, Wellman and Welsh (2001), ajudando a definir as relações que ligam os conceitos. Finalmente, Bartolini and Jennings (2002) ofereceu a noção das regras de negociação. Deve ser notado que nenhum dos esforços anteriores era uma ontologia como definido em Studer, Benjamins and Fensel (1998). A Figura 4.1 mostra a ontologia de negociação resultante desse processo de fusão usando um modelo de entidade-relacionamento para representar conceitos e relações.

Em particular, a classificação de London (Field, Stolze and Stroebel 2000) identificou o conjunto de conceitos que caracteriza a maioria dos protocolos de negociação e, portanto, não é restrita apenas a leilões. Os conceitos identificados pela classificação de London incluídos na ontologia são:

- *People:* corresponde ao conceito *Party* na ontologia de negociação mostrada na Figura 4.1;
- *Goods:* representado por *Object* na Figura 4.1*;*
- *Process*: já ilustrado na Figura 4.1.

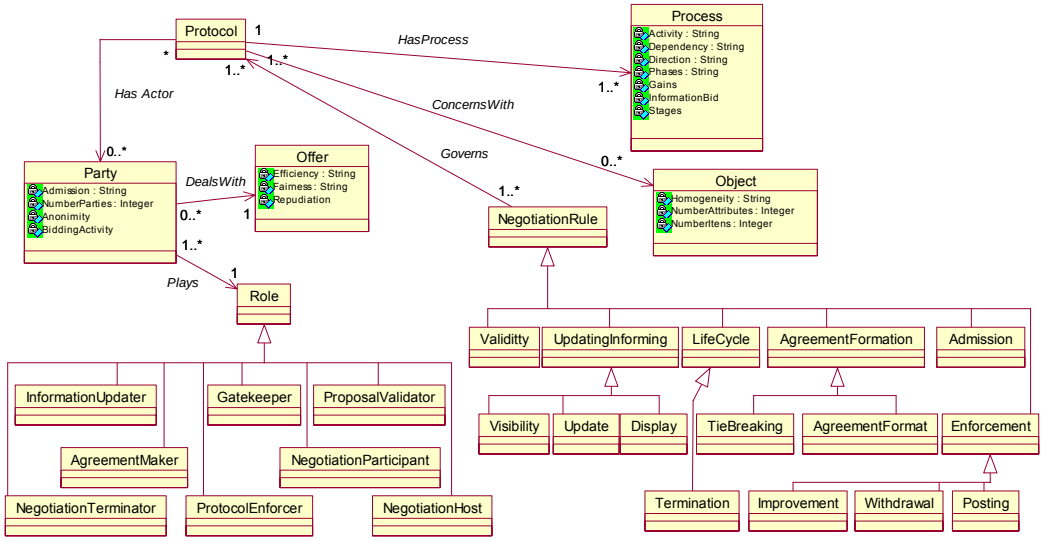

Figura 4.1: Um modelo Entidade-Relacionamento da ontologia de negociação

Na ontologia apresentada na Figura 4.1, o protocolo de negociação é definido em termos dos seguintes conceitos, cada um dos quais destaca um aspecto diferente de um protocolo de negociação:

- *Protocol*: define um protocolo genérico que define as "regras de encontro" que são seguidas pelos participantes da negociação durante um processo de negociação. As regras descrevem as condições que definem as interações entre agentes, os acordos que podem ser feitos e as seqüências de ofertas permitidas (Lomuscio, Wooldridge and Jennings 2000);
- *Party*: descreve um agente único (seja humano ou eletrônico) ou uma organização de agentes que participam em uma negociação. Vários agentes podem negociar e podem exercer diferentes papéis na negociação;
- *Process*: define a maneira de atingir um acordo sobre algum objetivo, modificando-se os atributos da negociação;
- *Object*: descreve os objetos da negociação, que são os bens materiais ou imateriais que são transferidos uma vez que o acordo seja atingido;
- *Offer*: apresenta uma combinação possível de valores associados aos atributos da negociação os quais representam uma expressão de vontade (por exemplo, comprar uma certa quantidade de produtos, receber os produtos em uma certa data, ou pagar um preço máximo pelos produtos);

- *Negotiation_Rule*: define o conjunto de regras que governam um protocolo de negociação específico. O protocolo genérico é paramétrico com respeito às regras de negociação que são aplicáveis ao tipo do mercado eletrônico modelado pelo protocolo. Na ontologia, isso significa que identificamos uma quantidade de regras de negociação, e a maneira pela qual elas são especificadas define um protocolo de negociação específico. As regras de negociação são divididas em categorias que serão descritas na seção 4.3.1.2.
- *Role*: Existem dois papéis principais na negociação - participante e *host*. O primeiro que deseja alcançar o acordo, enquanto o segundo é o papel responsável pela execução do protocolo e regras de negociação. O *host* da negociação é implementado por um agente que pode assumir vários papeis definidos na seção 4.3.1.1.

Os conceitos de nível mais baixo na ontologia de negociação especificam os papéis desempenhados pelos agentes envolvidos em um processo de negociação e as regras que descrevem os estágios e, as características de um protocolo, tais os agentes que são autorizados a ver as ofertas, como uma negociação termina etc.

O protocolo geral de negociação define a forma pela qual os agentes interagem durante a negociação. Esse protocolo foi baseado num modelo abstrato de negociação, formado pela análise do que é comum em diferentes formas de negociação. O protocolo pode ser especializado para um determinado mecanismo de negociação especificando, as regras de negociação, bem como adicionar novas regras ou excluir as atuais. Em Bartolini and Jennings (2002), foi definido uma linguagem para expressar as propostas de negociação. Um protocolo, que é governado por regras de negociação, é modelado através da relação *Protocol IsGovernedby Negotiation_Rule*, em que o conceito *Negotiation_Rule* é especificado pelos diferentes tipos de regras identificados em Bartolini and Jennings (2002).

Relações *ad hoc* descrevem como os conceitos identificados interagem para definir o domínio do protocolo de negociação. Por exemplo, um *Protocol HasActor Party* que modela o fato de que uma quantidade de agentes interagem em um protocolo de negociação. *Party Plays Role*, define que agentes desempenham papéis diferentes na interação. O conceito *Role* é especificado pelos papéis identificados em Bartolini and Jennings (2002).

O refinamento de um conceito é obtido, restringindo-se os valores associados com os atributos que descrevem os conceitos, ou adicionando-se novos atributos que associam a um conceito adicional mais características específicas. Por exemplo, o conceito *Protocol* é caracterizado pelo atributo *HasActor* cuja cardinalidade mínima é 2, isto é, pelo menos dois agentes precisam estar envolvidos em um protocolo. Entretanto, quando se define o Protocolo Inglês de Leilões, a cardinalidade mínima é restringida a 3, pois em um leilão inglês, precisam estar envolvidos pelo menos dois agentes e um leiloeiro (que são todos subconceitos de *Party*).

Na construção da ontologia, foi usado o editor de ontologias *WebOde* (Corcho, Fernández-López and Gómez-Pérez 2003), que é independente de formalismos de representação específicos, e foi traduzido para XML (Xml.org 2001) e DAML+OIL (Corcho, Fernández-López and Gómez-Pérez 2003).

Essa abordagem apresenta duas vantagens: uma é a flexibilidade, pois protocolos de negociação não necessitam ser codificados nos agentes, mas podem ser aprendidos dinamicamente, adquirindo-se a ontologia. E a outra é que a ontologia oferece a terminologia para raciocinar em termos de protocolos de negociação, seus componentes e as restrições que os regulam.

4.2.1.1 Papéis na Negociação

Como já foi mencionado, os agentes podem assumir dois papéis principais: *host* e participante. O agente cujo papel é o *host* também pode executar o papel de participante (exemplo, uma negociação um-para-um) ou não ser um participante (exemplo, leiloeiro em um leilão). Em alguns casos, o papel do *host* pode alternar entre entidades diferentes durante o progresso da negociação, como descrito em Bartolini and Jennings (2002):

- *Gatekeeper*: Verificar a política que governa a admissão à negociação.
- *Proposal validator*: Assegurar que uma proposta está bem formada seguindo a estrutura da negociação.
- *Protocol enforcer*: Certificar que as propostas dos participantes são postadas e encerradas de acordo com as regras da negociação.
- *Agreement maker*: Garantir que os acordos são formados de acordo com as regras de formação de acordo.
- *Information updater*: Notificar os participantes do atual estado da negociação.

- *Negotiation terminator*: Declarar que a negociação terminou.

4.2.1.2 Taxonomia das Regras de Negociação

Ao examinar os pontos da flexibilidade do processo abstrato da negociação descrito na Seção 4.2.1, foram identificadas as seguintes categorias de regras de negociação, junto com os papéis responsáveis pelas mesmas:

- **Regras para Admissão de Participantes**
  Definição: Permite se um agente participa ou não da negociação.
  Papel Responsável: *Gatekeeper*
  Admission Rule: Governa admissão de uma participante na negociação.
- **Regras para Propostas Válidas**
  Definição: Verifica se uma proposta está bem ou mal formada.
  Papel Responsável: *Proposal Validator*
  Validity Rule: Garantir que a proposta submetida deve ser condizente com o template da negociação.
- **Regras para Execução do Protocolo**
  Definição: Verifica a ordem de execução das propostas.
  Papel Responsável: *Protocol Enforcer*
  Posting Rule: Determina quando um participante pode fazer uma proposta.
  Improvement Rule: Especifica, dado um conjunto de propostas existentes, quais as novas propostas devem ser postadas.
  Withdrawal Rule: Especifica se e quando as propostas podem ser retiradas, e verifica o tempo de expiração das mesmas.
- **Regras de Atualização de Status e Informação de Participantes**
  Definição: Informar aos participantes aspectos da negociação.
  Papel Responsável: *Information Updater*
  Update Rule: Especifica como os parâmetros da negociação mudam na ocorrência de certos eventos.
  Visibility Rule: Especifica os participantes que podem visualizar uma proposta.
  Display Rule: Especifica se e como o updater da informação notifica os participantes que uma proposta foi submetida ou um acordo foi feito - transmitindo a proposta não-modificada ou um resumo da situação.
- **Regras de Formação de Acordo**

Definição: Define os acordos que serão formados entre as propostas.

Papel Responsável: *Agreement Maker*

Agreement Formation Rule: Determina, dado um conjunto de propostas, em que no mínimo duas são compatíveis, os acordos que serão formados.

Tie-breaking rule: Especifica a regra de formação de acordo aplicada após as outras regras de formação.

- **Regra do Ciclo de Vida da Negociação**

  Definição: Define quando nenhuma proposta pode ser postada.

  Papel Responsável: *Negotiation Terminator*

  Termination Rule: Especifica quando nenhuma proposta pode ser feita (por exemplo, depois de um certo tempo e/ou, após um período sem postagem de proposta).

4.2.1.3 Protocolo Geral de Negociação

O protocolo de negociação mostra como os papéis interagem através do uso de diagrama de atividades UML (Booch, Rumbaugh, and Jacobson 1999). A Figura 4.2 mostra o funcionamento completo do protocolo, iniciando o processo com a admissão dos participantes cujos requisitos foram aceitos pelo *Gatekeeper*, ou caso contrário, com a admissão rejeitada.

Os participantes se comunicam por meio de mensagens, a negociação continua conforme definido em *Termination rule*. O encerramento pode ocorrer após um acordo formado ou depois de decorrido um certo tempo. O participante submete uma proposta, e o agente no papel de *Proposal validator* verifica se a proposta está sintaticamente bem formada. Se a proposta não é válida, ela é rejeitada. Se a proposta passar no primeiro estagio de validação, o agente executando o papel de *Protocol enforcer* checa se a mesma satisfaz as regras de negociação. Estas regras definem a forma pela qual a negociação deve ocorrer e pode incluir restrições em quando uma proposta pode ser feita e os requisitos semânticos nas propostas válidas (requisitos que devem ser melhorados baseados nas propostas anteriores). Se a proposta passar nesta segunda etapa de validação, o conjunto de propostas e as estruturas de dados associados são atualizados e os participantes são notificados. *Visibility rule* e *display rule* definem quem será notificado e a estrutura da notificação.

Um processo de formação de acordo pode acontecer a qualquer momento durante a negociação, conforme *Agreement formation rule*, definidas antes do início da negociação. O agente (responsável pelo papel *agreement maker*) procura, no conjunto de propostas, determinar os acordos a serem executados. Os acordos podem ocorrer quando duas ou mais partes fazem propostas compatíveis. Se este for o caso, *Agreement formation rules* determinam, exatamente, a proposta escolhida e o exemplo de acordo final que será usado. Ao aplicar *Agreement formation rule*, o agente no papel de *information updater* notifica os participantes.

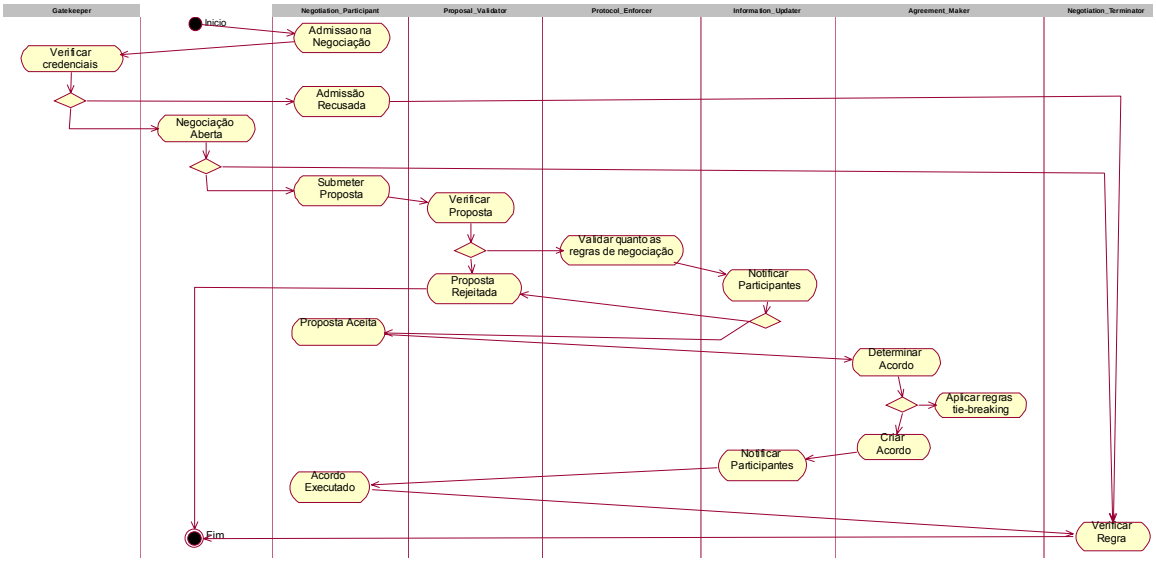

Figura 4.2: Diagrama de Atividades do Protocolo

4.2.2 Negociação Baseada em Argumento

Em sistemas multiagentes, os agentes, freqüentemente, não têm controle inerente sobre um outro agente e assim a única maneira que um pode influenciar o comportamento do outro é através da persuasão. Em alguns casos, o persuadido pode exigir um argumento que o convença a agir na maneira desejada pelo persuasor. Entretanto, em outros casos, o persuadido pode, inicialmente, se recusar a aceitar a proposta e deve ser persuadido para mudar sua opinião, objetivos ou preferências de modo que a proposta, ou algum variante disso seja aceito. A exigência mínima para negociação é que os agentes possam fazer propostas uns aos outros.

Um outro nível de negociação ocorre quando os receptores não têm apenas a escolha de aceitar ou de rejeitar propostas, mas têm a opção de fazer ofertas contrárias para

alterar os aspectos da proposta que são insatisfatórios. A ontologia proposta, em Sierra et al (1997), apresenta um modelo de negociação baseado em argumentos nos quais os agentes podem justificar ou argumentar junto com as propostas (contrárias) indicando porque devem ser aceitos.

O cenário utilizado, para ilustrar os princípios e conceitos do modelo proposto em Sierra et al (1997), é motivado pelo projeto ADEPT (Jennings et al 1996) que desenvolve um sistema multiagente para controlar um processo de negócio da *British Telecom* (BT). Em termos gerais, o processo recebe como entrada uma requisição de serviço do cliente e como saída gera uma citação especificando quanto custaria construir uma rede para realizar esse serviço.

4.2.2.1. Modelo de Negociação

O modelo descreve o processo de negociação entre múltiplos agentes para se alcançar um acordo. A negociação é atingida através da troca de ilocuções em uma linguagem de comunicação compartilhada *CL*. A troca de ilocuções é direcionada pelas necessidades e objetivos individuais dos agentes – algo que não será parte desse modelo de negociação. No entanto, essa negociação necessita de uma notação compartilhada no intuito de utilizar as ilocuções em *CL*, e um simples protocolo de negociação (Sierra et al 1997). Estas notações se relacionam:

1. aos elementos que são relevantes para a negociação - na forma de conceitos e valores que podem desenvolver como resultados da negociação.
2. à racionalidade dos agentes participantes – em termos de preferência de relacionamentos ou funções de utilidade que permitem aos agentes avaliarem e compararem diferentes propostas.
3. à capacidade de deliberação dos agentes participantes – na forma de um estado interno no qual o agente pode registrar o histórico da negociação, bem como a evolução dos seus elementos teóricos nas quais suas decisões são fundamentadas.
4. ao mínimo de conhecimento compartilhado de ilocuções aceitáveis – capturado quando é recebida uma ilocução que deve ser interpretada quando um agente escutar, e pela estrutura explícita das condições que permitem um agente usar (ou gerar) uma ilocução em um certo tempo.

4.2.2.2 Ontologia para Negociação

Esta Seção apresenta a ontologia para negociação definida em Sierra et al (1997). Em uma negociação, cada agente tem um único identificador cujo conjunto é denotado por *Agentes*. Para modelar a característica de persuasão, assumiu-se que entre os agentes há uma relação social e geral. Essa relação pode ser modelada como uma função binária do conjunto de funções sociais, denotado como *Papéis*. No cenário BT, por exemplo, *Papéis* poderiam ser: {*Cliente, Empreiteiro, Chefe, Colega*}. Os agentes trocam ilocuções em uma linguagem de negociação comum *CL* definido sobre um conjunto de partículas ilocucionárias expressas numa linguagem lógica comum *L*. A natureza precisa de *L* não é importante, porém deve conter:

1. Variáveis: representam os conceitos da negociação, que precisam ser restringidos a diferentes valores durante a negociação.
2. Constantes: representam valores para os conceitos na negociação. A constante "?" representa a ausência de valor, e pode ser usada em propostas definidas entre agentes.
3. Igualdade: especifica o valor de um conceito na negociação.
4. Conjunção: define sentenças complexas.

As características acima são necessárias para expressar os tipos de sentenças envolvidas nas propostas de negociação discutidas neste trabalho (Sierra et al 1997). A seguir é apresentado um exemplo de sentença:

$$(Price = £10) \wedge (Quality = High) \wedge (Penalty = ?)$$

onde `*Price*`, ` *Quality*` e ` *Penalty*` são os conceitos da negociação e são representadas como variáveis, `*£10*`, ` *High*` e `*?*` são valores para esses conceitos e logo para as constantes;`=` denota igualdade; e ` ∧ ` denota conjunção. Contudo, a linguagem definida não é expressiva o suficiente para descrever todos os aspectos da negociação. Em particular, o "raciocínio" e a "discussão" são imprescindíveis para que os agentes expressem suas preferências entre as ofertas. As ofertas são formuladas em *L*, portanto a forma mais simples de representar preferências entre fórmulas seria uma relação de segunda-ordem em *L*. Porém, isto significaria que *L* seria uma lógica de mais alta ordem, com problemas computacionais dessas lógicas. Baseado nisso, as preferências são expressas como uma meta-linguagem *ML*, com os seguintes requisitos:

1. Funções de Citação: para representar fórmulas em *L* como termos em *ML*.

2. Uma meta-predicado de preferência: Para expressar preferências entre fórmulas em *L*.

Por exemplo, dada as sentenças, *Price = £10* e *Price = £20* em *L*, pode-se expressar uma preferência da primeira para segunda como:

$$Pref(equal(\ulcorner Price \urcorner, \ulcorner £10 \urcorner), equal(\ulcorner Price \urcorner, \ulcorner £20 \urcorner)$$

onde `*equal*` é a citação em *ML* do predicado `=` em *L*, e `*Pref*` representa o meta-predicado de preferência. Entretanto, nesse trabalho será usada a representação mais compacta $\ulcorner Price = £10 \urcorner$.

A linguagem comum de comunicação, *CL*, é composta por um conjunto de partículas ilocucionárias necessárias para modelar o conjunto de atos ilocucionários utilizados na negociação baseada em argumento. Os atos ilocucionários podem ser divididos em dois conjuntos, I_{nego} correspondente a partículas de negociação (usadas para fazer ofertas e contra-ofertas) e I_{pers} correspondentes a partículas persuasivas (usadas na argumentação).

I_{nego} = {**`ofertar,requisitar,aceitar,rejeitar,retirar-se`**}
I_{pers} = {**`apelar,ameaçar,recompensar`**}

A estrutura de negociação entre dois agentes consiste numa seqüência de ofertas e contra-ofertas contendo valores para conceitos. Essas ofertas podem ser pares de conjunções 'conceito = valor' ou podem ser acompanhadas por argumentos persuasivos (**`apelar,ameaçar,recompensar`**). A seleção de três partículas persuasivas no conjunto I_{pers} é resultado da análise do domínio e da literatura de persuasão.

O processo de negociação é baseado na troca de mensagens, permitindo por meio da argumentação modificar as crenças do outro agente. Para tanto, nessa ontologia, os agentes utilizam as seguintes atividades: **`ofertar, requisitar, aceitar, rejeitar`** e **`retirar-se`**. Com exceção de **`retirar-se`** as estruturas das partículas são semelhantes entre si, **`atividade`** (*a,b,δ,t*). Onde:

a - agente persuasor da negociação
b - agente persuadido da negociação
δ - oferta que o agente *a* deseja efetuar com o agente *b*.
t - tempo da negociação

Ao sair da negociação, o agente *a* manda a seguinte mensagem ao agente *b*: **retirar-se**(*a,b,t*), seja por que o acordo foi alcançado entre os agentes da negociação ou após um certo tempo *t*.

Os tipos de apelo seguem a estrutura abaixo, no entanto, podem diferenciar variando φ em *L* ou *ML* ou pela variação [*not*]φ em *CL* – *not* φ significa que a ação φ não ocorre. A estrutura do ato ilocucionário apelar é **apelar**(*a,b*,ξ, [*not*]φ,*t*), onde:

a - agente persuasor da negociação;
b - agente persuadido da negociação;
ξ - fórmula em *L* ou *ML*;
φ - argumento utilizado pelo agente *a* para convencer o agente *b* em suporte a fórmula ξ;
t - tempo da negociação.

As estruturas das outras partículas persuasivas são semelhantes entre si: **ameaçar**(*a,b*,[*not*]ψ_1, [*not*]ψ_2,*t*) e **recompensar**(*a,b*,[*not*]ψ_1, [*not*]ψ_2,*t*), onde:

a - agente persuasor da negociação;
b - agente persuadido da negociação;
ψ_1 e ψ_2 – fórmula em *CL*;
t - tempo da negociação.

As estruturas de **ameaçar** e **recompensar** são recursivas desde que as formulas ψ_1 e ψ_2 sejam ilocuções em *CL*. Essa definição recursiva permite que um conjunto de ações possíveis suportem a persuasão. Por exemplo, um agente *A* pode ameaçar um agente *B* que irá informar ao chefe de *B* da incompetência de *B* se *B* não aceitar o negócio.

ameaçar(*A,B*, *not* **aceitar**(*B*,*A*,*time* = 24h,t_2),
apelar(*B*,*Chefe_de_B,B=incompetente*,
not **aceitar**(B,A, *time* = 24h,t_2), t_3), t_1)

Após introduzir todos os componentes, será descrita a seguir uma estrutura de diálogo para uma negociação persuasiva (Sierra et al 1997).

Definição 1. Uma Estrutura de Diálogo é uma tupla DF = <*Agentes, Papéis, R, L, ML, CL, Tempo*>, onde:

1. *Agentes* é um conjunto de identificadores de agentes.
2. *Papéis* é um conjunto de identificadores de papel.
3. *R : Agentes x Agentes* → *Papéis*, determina um papel para cada par de agentes,se existir.

4. *L* é uma linguagem lógica que satisfaz os requisitos mencionados acima. *Acordos(L)* denota o conjunto de possibilidades de fórmulas conjuntivas em *L* entre conceitos e valores, isto é, $x_1 = v_1 \wedge ... \wedge x_n = v_n$. $Acordos_{?\text{-}free}(L) \subset$ *Acordos(L)* excluindo '?' como um valor aceitável em um acordo.
5. *ML* é uma metalinguagem de *L* satisfazendo os requisitos mencionados acima.
6. *CL* é uma linguagem de comunicação entre agentes. Sendo $a,b \in Agentes$ e $t \in Tempo$, é definido como:
   (a) se $\delta \in Acordos(L)$ então **requisitar**$(a,b,\delta,t) \in CL$.
   (b) se $\delta \in Acordos_{?\text{-}free}(L)$ então **ofertar**(a,b,δ,t), **aceitar**(a,b,δ,t), **rejeitar**$(a,b,\delta,t) \in CL$.
   (c) **retirar-se**$(a,b,t) \in CL$
   (d) se $\psi_1, \psi_2 \in CL$, $\xi \in L \cup ML$, e $\varphi \in L \cup ML \cup CL$ então **ameaçar**$(a,b,[not]\psi_1,[not]\psi_2,t)$,**recompensar**$(a,b,[not]\psi_1,[not]\psi_2,t)$, **apelar**$(a,b,\xi,[not]\varphi,t) \in CL$.
7. *Tempo* é um conjunto discreto ordenado de instantes.

É importante notar que o tempo, que aparece como último argumento nas ilocuções, será omitido quando não houver ambigüidade.

4.2.3. Ontologia para Negociação entre Agentes de Informação

A ontologia proposta em Bailin and Truszkowski (2001) tem como objetivo especificar e apresentar uma ontologia para negociação em que os agentes resolvam conflitos semânticos em tempo real sem intervenção humana. O software resultante fornece agentes com meios de chegar a uma linguagem comum para conversar. A "linguagem comum" não se refere apenas à sintaxe, mas também ao significado de termos trocados entre os agentes.

Os termos trocados consistem, inicialmente, do conteúdo da consulta e nos identificadores do documento para os resultados da consulta. Ambos podem ser vistos como palavras-chaves que descrevem o documento desejado (consulta) ou encontrado (resultados). A pesquisa indica que há três processos envolvidos neste tipo de negociação:

- *Esclarecimento* do significado as palavras-chaves
- *Explicação* da relevância dos resultados da consulta

- *Desenvolvimento* de uma ontologia de agente baseado no esclarecimento e relevância da explicação

4.2.3.1. Objetos do Protocolo

O protocolo de ontologia para negociação (ONP) é baseado em um pequeno conjunto de tipos de objetos e as operações destes objetos (Bailin and Truszkowski 2001). Os tipos de objetos são apresentados na Tabela 4.1. **Consultas** são requisições para encontrar documentos e/ou URLs que são relevantes para um conjunto de palavra-chaves. **Declinação** expressa a má-vontade de um agente para responder uma consulta (por razões de capacidade ou potencialidade), enquanto **Rejeição** é a expressão do descontentamento de um agente com os **Resultados da Consulta**.

Tipo de Objeto	Abreviação
Consulta	Que
Resultados da Consulta	Qre
Agradecimento	Ack
Rejeição	Rej
Declinação	Dcl
Capacidade	Cap
Confirmação de Interpretação	Cfi
Esclarecimento	Cla
Explicação de Relevância	Exr
Explicação do Fato	Exf
Ontologia	Ont

Tabela 4.1: Objetos do Protocolo da Ontologia para Negociação

Confirmação de Interpretação é a confirmação do agente A da tentativa de compreensão do agente B da mensagem enviada pelo agente A. **Esclarecimento** é a forma pela qual um agente torna explícito o significado de uma mensagem previamente enviada.

Um agente recebe os resultados da consulta como um conjunto de URLs e palavras-chaves de documentos. As palavras-chaves são usadas, pelo agente, para avaliar a relevância dos resultados da consulta. Se o agente não entender porque uma particular URL é relevante, pode-se requisitar ao agente, que retornou os resultados, uma **Explicação de Relevância**. Quando o agente recebe uma explicação de relevância, mas não pode entende um fato particular usado na explicação, ele pode solicitar ao servidor de agentes uma **Explicação do Fato**.

4.2.3.2 Estados e Transições do Protocolo

Os estados do protocolo correspondem à execução de uma ou outra operação em um particular tipo de objeto. As operações são mostradas na Tabela 4.2. Nem toda combinação de uma operação e um objeto representa um estado significante.

Operação	Abreviação
Espera por	Wtg
Requisitar	Req
Receber Requisição para	Rrf
Receber	Rcd
Processar	Pro
Enviar	Fwd
Retornar	Ret
Interpretar	Int
Interpretar Requisição para	Irf
Avaliar	Evl
Desenvolver	Evo

Tabela 4.2. Operações do Protocolo da Ontologia para Negociação

A Tabela 4.3 apresenta os estados do ONP, usando as abreviações listadas nas Tabelas 4.1 e 4.2. Por exemplo, um agente pode **Esperar por** uma **Capacidade** (se ele tiver requisitado uma), mas não pode **Esperar por** uma **Declinação**. Similarmente, um agente pode **Receber** uma **Declinação** de outro agente, mas não pode **Receber** uma **Ontologia**.

	Que	Qre	Ack	Rej	Dcl	Cap	Cfi	Cla	Exr	Exf	Ont
Wtg	x	x	x			x	x	x	x	x	
Req						x	x	x	x	x	
Rrf						x	x	x	x	x	
Rcd	x	x	x	x	x	x	x	x	x	x	
Pro	x										
Fwd	x										
Ret		x	x	x	x	x	x	x	x	x	
Int	x	x				x					
Irf						x					
Evl		x				x					
Evo											x

Tabela 4.3 Estados do Protocolo de Ontologia para Negociação

O comportamento dos agentes, que compartilham o protocolo, é determinado pelas transições entre os possíveis estados. Existem 46 estados definidos no protocolo, as transições são selecionadas a partir de uma matriz de um par de estados.

4.3. Construção de Ontologias

Existem situações em que é necessário construir ontologias para: compartilhar a estrutura da informação comum entre pessoas ou softwares, reutilizar o conhecimento do domínio, tornar explícitos fatos consensuais, separar o domínio do conhecimento do conhecimento operacional ou analisar um domínio (Noy and Guinness 2001). Nas seções seguintes, uma breve revisão de literatura é apresentada sobre as metodologias, linguagens e ferramentas disponíveis para a construção de ontologias.

4.3.1. Metodologias disponíveis para a construção de ontologias

Antes de se iniciar a construção de uma ontologia, é importante fazer uma boa investigação a respeito da metodologia a ser utilizada, pois, para se garantir que a ontologia criada, poderá ser reutilizada, posteriormente, em outros sistemas, deve-se adotar o uso de uma metodologia que garanta sua portabilidade. Existem várias metodologias disponíveis na literatura, tais como: Uschold & King, Grüninger e Fox, Kactus, Methontology, Sensus e On-to-knowledge (Corcho, Fernández-López and Gómez-Pérez 2003).

Dentre as metodologias citadas anteriormente, a FIPA recomenda a *Methontology*, para uma tarefa de construção de ontologia. Desenvolvida no Laboratório de Inteligência Artificial na Universidade Técnica de Madri (UPM), *Methontology* (Corcho, Fernández-López and Gómez-Pérez 2003) constrói uma ontologia por reengenharia sobre outra, utilizando o conhecimento do domínio. Incluindo: (a) a identificação do processo do desenvolvimento da ontologia, consultando tarefas (planejamento, especificação, aquisição de conhecimento, conceitualização, documentação, etc.) que se deve realizar ao construir ontologias; (b) um ciclo de vida baseado nos protótipos de desenvolvimento, que identifica os estágios através do qual a ontologia passa durante sua existência; e (c) a própria metodologia especifica as etapas para executar cada atividade, as técnicas usadas, os produtos de saída e como eles devem ser avaliados.

As metodologias citadas possuem abordagens e características diversas. Não parece provável a unificação das propostas em uma única metodologia. Para verificar a utilidade das metodologias e compará-las, é necessário avaliar a ontologia resultante da aplicação de cada metodologia.

4.3.2. Linguagens disponíveis para a construção de ontologias

Antes de iniciar a implementação da ontologia, é importante decidir as necessidades da ontologia em termos de expressividade e serviços de inferência, visto que nem todas as linguagens existentes representam os componentes e o raciocínio da mesma maneira. As linguagens para construção de ontologias mais representativas descritas recentemente na literatura são: FLogic(*Frame Logic*), Loom, OCML, RDF, SHOE, OIL, DAML + OIL e OWL (Corcho, Fernández-López and Gómez-Pérez 2003).

Entre as linguagens mencionadas, a mais representativa é a RDF (*Resource Description Framework*), desenvolvida pelo W3C (*World Wide Web Consortium*). Foi criada para facilitar o intercâmbio de informações, que podem ser interpretadas por máquinas, entre aplicativos via web, além de adicionar semântica formal, representar metadados e facilitar a representação do conhecimento. A especificação RDF é dividida em duas partes principais: RDF e RDF-*Schema*. A primeira define como descrever recursos através de suas propriedades e valores, enquanto a segunda define propriedades específicas, restringindo sua utilização (Corcho, Fernández-López and Gómez-Pérez 2003). A combinação dessas duas partes é conhecida normalmente como RDF(S). Como extensões de RDF(S) surgiram: OIL (*Ontology Interchange Language*), DAML (*DARPA Agent Markup Language*) +OIL e OWL (*Web Ontology Language*).

Se o objetivo é criar uma ontologia que realize raciocínios complexos utilizando conceitos, taxonomias e relações binárias, a escolha de uma linguagem especifica é crucial para desenvolver uma aplicação baseada em ontologia.

4.3.3. Ferramentas disponíveis para a construção de ontologias

Nos últimos anos, o número de ambientes e ferramentas de construção de ontologias tem crescido bastante. Essas ferramentas fornecem suporte ao processo de desenvolvimento da ontologia e ao uso da ontologia final. Em geral, as ferramentas utilizam linguagens de representação para a construção das ontologias, as quais foram apresentadas na seção anterior. As ferramentas para construção de ontologia mais

relevantes na literatura são: *Ontolingua Server*, Ontosaurus e *WebOnto* (Corcho, Fernández-López and Gómez-Pérez 2003).

Nos últimos anos, uma nova geração de ambientes de engenharia de ontologias foi desenvolvida para integrar a tecnologia de ontologias em sistemas de informação reais. São construídos como ambientes ou aplicação integrados que fornecem suporte tecnológico a maioria das atividades do ciclo de vida da ontologia. Essas ferramentas são arquiteturas baseadas em componente, em que novos módulos podem facilmente ser adicionados para fornecer mais funcionalidades ao ambiente. Por se tratar de uma tarefa dispendiosa, qualquer apoio na construção de ontologias pode representar ganhos significativos (Corcho, Fernández-López and Gómez-Pérez 2003). Entre estes ambientes, os mais citados são Protégé 2000, WebODE e OntoEdit. O editor Protégé (Protégé 2005) foi utilizado nesse trabalho para modelar as ontologias adaptadas aos métodos de negociação utilizados no NeurAge. A próxima seção apresenta o Protégé.

Finalmente, com o surgimento da Web Semântica aumentou o número de ferramentas para o desenvolvimento das ontologias DAML+OIL e RDF(S). De fato, as ferramentas anteriores (Protégé, WebODE e OntoEdit) permitem importar e exportar as ontologias DAML+OIL e RDF(S). Há também ferramentas isoladas que criam ontologias DAML+OIL, as mais representativas são: OILEd (uma ferramenta baseada em DL) e DUET (um plugin baseado em UML para *Rational Rose*).

Atualmente, existem ontologias com outros propósitos: especializadas em combinar ontologias (Chimaera, Protégé-PROMPT), tradução de ontologia entre linguagens (Ontomorph), anotação de páginas web baseado em ontologia (COHSE, OntoMat, SHOE *Knowledge Annotator*), avaliação de ontologia (OntoAnalyser, ONE-T, ODEClean), mecanismos de consulta RDF (*RDFSuite*, Sesame, Inkling, Jena, etc.), etc. Porém, esse estudo está focado apenas nas ferramentas de desenvolvimento de ontologia(Corcho, Fernández-López and Gómez-Pérez 2003).

4.3.3.1 Protégé

Conforme mencionado, a ferramenta utilizada para a formalização das ontologias foi o Protégé (Protégé 2005). A popularização da construção de ontologias, principalmente pelas comunidades de pesquisa em Inteligência Artificial, motivou o desenvolvimento de ferramentas para auxiliar esta tarefa. O Protégé possui uma interface gráfica (Figura 6.1) que a torna de fácil utilização. A versão 2000 foi totalmente desenvolvida em Java,

garantindo total portabilidade para qualquer ambiente e inclui uma API (interface de programação) para estender-se o ambiente a fim de que suporte aplicações em domínios específicos.

Além da modelagem do conhecimento através da definição de classes (organizadas hierarquicamente) e relações entre elas, o Protégé oferece uma interface para a introdução de dados (instâncias) específicos para a criação de uma base de dados. Uma ontologia no Protégé consiste basicamente de:

- Classes: conceitos do domínio abordado que constituem uma hierarquia taxonômica;
- Slots: descrevem propriedades de classes e instâncias;
- Facets: descrevem propriedades de slots e permitem a especificação de restrições nos valores dos slots;

Há dezenas de plug-ins disponíveis para estender as funcionalidades do Protégé, permitindo, entre outras coisas, inferências e buscas na ontologia e sua gravação em diversos formatos, tais como RDF, OWL e DAML+OIL. Isso permite a integração e troca de ontologias com outras ferramentas. Dentre os plug-ins do Protégé, destaca-se o beangenerator (Van Aart et al 2002) que foi utilizado para exportar o código da ontologia modelada para a plataforma JADE. O capitulo 6 apresenta brevemente o *beangenerator*.

4.3.4. Conclusões

A Figura 4.3, adaptada de Corcho, Fernández-López and Gómez-Pérez (2003), apresenta as relações entre as principais metodologias, ferramentas e linguagens ilustradas nas subseções anteriores. A partir deste estudo, Corcho, Fernández-López and Gómez-Pérez (2003) extraiu algumas conclusões:

- Não há correspondência entre metodologias de construção e ferramentas de ontologia, com exceção de *Methontology* e WebODE, *e On-To-Knowledge* e OntoEdit. Como não há suporte tecnológico para a maioria das metodologias existentes, elas não podem ser, facilmente, aplicadas na tarefa de construção da ontologia. De fato, a maioria das ferramentas se direciona a poucas atividades do ciclo de vida da ontologia: projeto e implementação.

- Atualmente, não é necessário implementar ontologias, manualmente, pois a maioria das ferramentas gera ontologias em linguagens diferentes.

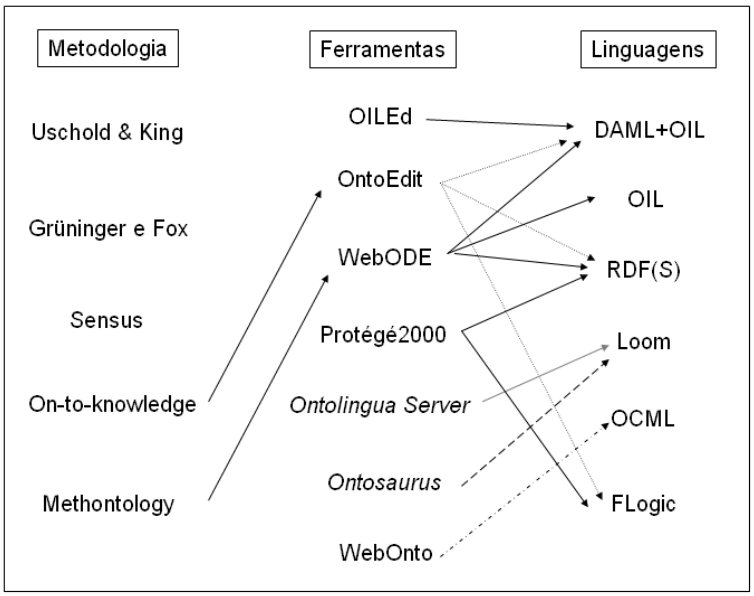

Figura 4.3: Metodologias, ferramentas e linguagens de ontologia

4.4. Resumo do Capítulo

Inicialmente, alguns conceitos sobre ontologia e ontologia para negociação são apresentados, como também algumas ontologias para negociação encontradas na literatura que serão utilizadas para a concepção deste trabalho. Após o estudo de algumas ontologias, foram escolhidas as propostas em Tamma, Wooldridge and Dickinson (2002), Sierra et al (1997), Bailin and Truszkowski (2001). As ontologias para negociação foram descritas, focalizando nas negociações que serão tomadas como base para o principal objetivo desse trabalho: adaptar ontologias existentes aos quatro métodos de negociação utilizados no sistema NeurAge.

Capítulo 5

Adaptação das Ontologias aos Métodos de Negociação do Sistema NeurAge

Este capítulo ilustra as alterações realizadas nas ontologias descritas em Tamma, Wooldridge and Dickinson (2002), Sierra et al (1997), Bailin and Truszkowski (2001) de acordo com o funcionamento dos quatro métodos de negociação utilizados no sistema NeurAge.

5.1. Ontologias adaptadas aos métodos de negociação do NeurAge

Como já foi mencionado, o objetivo desse trabalho é adaptar algumas ontologias já existentes na literatura, tais como as propostas em Tamma, Wooldridge and Dickinson (2002), Sierra et al (1997), Bailin and Truszkowski (2001), aos métodos de negociação utilizados no sistema NeurAge.

Os quatro métodos de negociação diferentes utilizam como base ontologias diferentes. As mesmas necessitam ser adaptadas em sua composição para que seja possível trabalhar com as informações fornecidas pelos agentes do sistema NeurAge, principalmente, no que se refere aos métodos de negociação utilizados por este sistema.

5.1.1. Adaptação da Ontologia para o Método de Negociação Baseado na Teoria Dos Jogos

Como já foi visto no Capítulo 4, a ontologia proposta em Tamma, Wooldridge and Dickinson (2002) utiliza um modelo de entidade-relacionamento para representar conceitos e relações. Em virtude do funcionamento do método de negociação baseado em teoria dos jogos, algumas mudanças são necessárias na ontologia original da Figura 4.1. As seções a seguir ilustram tais alterações.

Inicialmente, o modelo entidade-relacionamento foi transcrito para o diagrama de classes da UML (Booch, Rumbaugh, and Jacobson 1999) e então adaptado ao método de negociação baseado em teoria dos jogos. Dentre algumas alterações, ilustradas na Figura 5.1, destacam-se: incluir conceitos e atributos, bem como relacionamento entre conceitos, excluir regras de negociação, dentre outras.

5.1.1.1 Adaptação dos Conceitos da Ontologia

Inicialmente, as classes, atributos e relacionamentos foram traduzidos para português. Para o método baseado em teoria dos jogos, foi necessário alterar quatro conceitos: *Party, Offer, Object* e *Role.* Como já mencionado, o conceito *Party* descreve o agente que participa da negociação. Este conceito foi renomeado para Agente. As alterações foram necessárias porque o método em seu funcionamento utiliza informações como o custo da ação de um agente mudar de classe ou não, posição na lista de confiança, valores de *payoff*, esses atributos não existiam no conceito *Party* original. Portanto, se torna clara a necessidade da inclusão dos atributos Custo, *Rank* e *Payoff* ao conceito Agente.

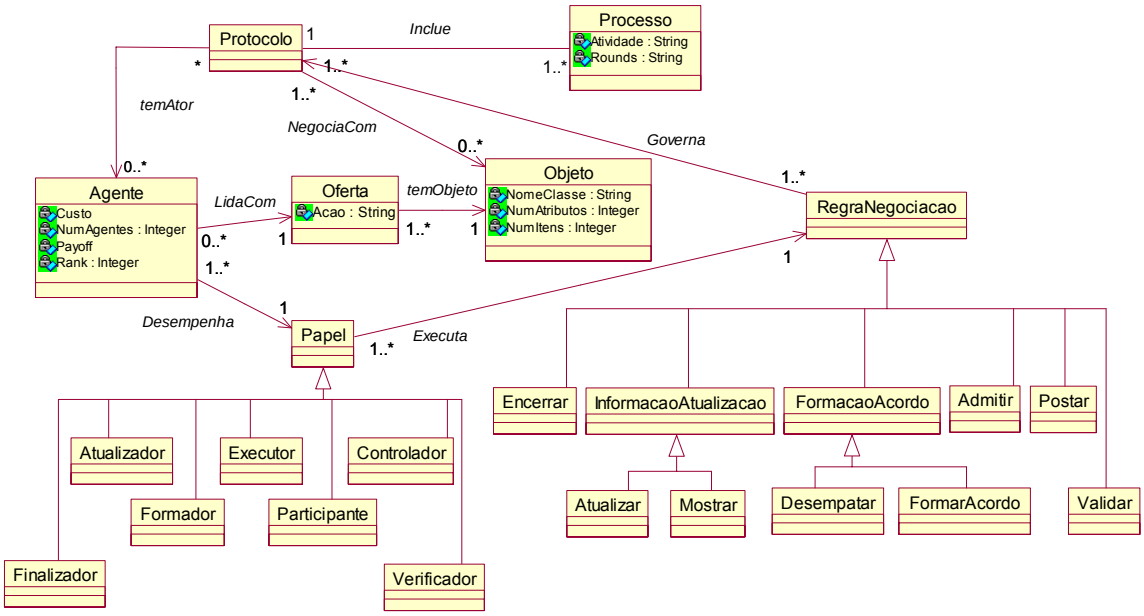

Figura 5.1: Diagrama de Classes da Ontologia adaptada a Teoria dos Jogos.

O conceito **Oferta** significa a proposta de um agente no processo da negociação, que também foi modificado, acrescentando-se o atributo **Ação** que é essencial ao método de negociação baseado na teoria dos jogos, pois ele determina se o agente vai mudar a classe escolhida como vencedor ou manter a classe vencedora. Além disso, esse atributo deve conter dois valores, que são os seguintes: manter a classe escolhida (**manter**) ou mudar para a classe escolhida pelo outro agente (**mudar**).

O objeto da **Oferta** a ser negociado é a classe vencedora determinada pelo agente com o valor da confiabilidade mais alta, logo foi acrescentado o atributo **nomeClasse** a classe **Objeto**.

O conceito **Papel** especifica os diferentes papéis que os agentes do sistema NeurAge podem assumir. Esse conceito também foi alterado ao incluir alguns papéis e retirar outros, estas alterações serão explicadas na seção seguinte.

5.1.1.2 Adaptação dos papéis da Ontologia

Originalmente na Figura 4.1, ao analisar o conceito *Role* o agente pode assumir de um até oito papéis (relacionamento 1:N). No entanto, em virtude do funcionamento do NeurAge, achou-se conveniente retirar o papel *Negotiation_Host* visto que no sistema não há um agente que faça o papel de mediador da negociação. O papel *Infrastructure_Provider* também foi retirado, pois ele é responsável em fornecer a

estrutura da negociação cujo estilo é *blackboard*, e os agentes do NeurAge se comunicam via mensagens trocadas entre eles.

5.1.1.3. Adaptação dos Relacionamentos entre os Conceitos da Ontologia

No que diz respeito ao relacionamento entre conceitos, foram acrescentados dois relacionamentos, enquanto dois foram excluídos. Na ontologia original da Figura 4.1, os conceitos *Offer* e *Object* não se relacionam. Porém, há a necessidade de relacionamento entre estes conceitos no sistema NeurAge. Portanto, foi acrescentado na ontologia da Figura 5.1 o relacionamento **temObjeto** entre os conceitos **Oferta** e **Objeto**. No sistema NeurAge uma oferta pode conter de um até N objetos, porém um objeto só pode pertencer a apenas uma oferta.

Os relacionamentos, da ontologia original, *ParticipatesIn* e *IsResponsibleFor* contém, basicamente, o mesmo sentido e ligam os mesmos conceitos Papel e *NegotiationRule*. Portanto, achou-se conveniente uni-los em um só conceito: **Executa**.

5.1.1.4. Adaptação das Regras de Negociação da Ontologia

As regras de negociação ilustradas na ontologia original da Figura 4.1, são todas especializações do conceito *NegotiationRule*. Contudo, algumas regras foram retiradas e outras foram modificadas com base no funcionamento do método baseado em teoria dos jogos, esta seção apresenta tais modificações. As regras *Agreement Formation Rule* e *Update Rule* foram adaptadas, enquanto *Improvement Rule, Visibility Rule* e *Withdrawal Rule* foram excluídas do processo de negociação.

Quando a negociação chega ao fim, um acordo de negociação é formado entre dois agentes negociadores. Ao contrário da regra de negociação original, as **Regras de Formação de Acordo** serão definidas antes da negociação começar. Pois, o número de acordos é pequeno - apenas quatro possibilidades (os dois agentes mudarem de opinião ou não mudarem e um mudar e o outro não mudar) – o que torna o processo de negociação mais rápido. Por exemplo, se ambos os agentes escolherem diferentes ações, o acordo é descartar o agente que escolheu mudar a ação, enquanto o agente que manteve a ação é considerado o vencedor e ainda está no processo de negociação. Porém, sua confiança é substituída pelo valor da confiabilidade da célula escolhida.

Uma regra significante na ontologia da Figura 4.2 é a **Regra de Atualização**, cujo objetivo é explicitar como os parâmetros da negociação se modificam na ocorrência de certos eventos. No método baseado em teoria dos jogos, se dois agentes escolhem a mesma ação de mudar ou não a classe escolhida, ambos continuam no processo de negociação, mas seus valores de confiança são trocados pelos valores de *payoff*. Neste caso, esta regra foi adaptada para que quando for utilizada altere o valor de confiança dos agentes.

Como já foi mencionado, algumas regras foram retiradas da ontologia original. Dentre elas, *Improvement Rule* que especifica dado um conjunto de propostas existentes quais as novas propostas devem ser postadas. No entanto, a **Regra de Postagem** já testa se um round já chegou ao fim, informando que novas propostas podem ser postadas. Portanto, para evitar ambigüidades entre regras optou-se por retirar *Improvement Rule.*

A **Regra de Visibilidade**, ilustrada na ontologia original da Figura 4.1, não é pertinente no método de negociação baseado em teoria dos jogos, pois todos os parâmetros são visíveis aos demais agentes participantes da negociação. Por isso, não há a necessidade dessa regra existir, pois o seu intuito é determinar os agentes que podem visualizar determinados parâmetros de uma nova proposta.

A **Regra de Encerramento** especifica quando uma proposta não deve ser feita, ou seja, esta regra define que uma negociação termina quando o método de negociação está inativo por um tempo maior que o especificado ou quando um acordo entre os agentes é formado. Essa regra é semelhante a *Withdrawal Rule*, cujo objetivo é especificar se e quando as propostas de negociação podem ser retiradas, e verificar o tempo de expiração das mesmas. Portanto, para evitar redundância nas regras se optou por não utilizar *Withdrawal Rule* no sistema NeurAge.

As demais regras de negociação ilustradas na ontologia da Figura 4.2 continuam: Regra de Admissão, Regra de Validação, Regra de Formação de Acordo, Regra de Atualização, Regra de Postagem, Regra de Mostragem, Regra de Desempate e Regra de Encerramento.

5.1.1.5. Estrutura da Negociação

A estrutura da negociação proposta em Tamma, Wooldridge and Dickinson (2002) é baseada em Bartolini and Jennings (2002), e expressa como uma coleção de fatos, declarando os campos que devem aparecer na proposta e quais são os opcionais, bem como os tipos de cada campo e restrições a seu valor. No método de negociação baseada na teoria dos jogos, uma proposta consiste em um agente A negociar com um agente B, se o agente B deve mudar ou não a classe escolhida em relação ao padrão de entrada. Baseado nisto, a estrutura de uma proposta necessita de três informações: identificador da proposta, o agente que propõe a ação e objeto da ação, que pode ser mudar ou manter a classe escolhida de acordo com o valor da confiabilidade. A estrutura da proposta deve ser:

```
(Identificador da proposta)
(Agente A que propõe a Ação A)
(Objeto da proposta=Acao_Mudar ou Acao_NãoMudar de acordo com a
confiabilidade)
```

Inicialmente, o agente deve ser admitido ou não no processo de negociação. Essa etapa é governada pela **Regra de Admissão**, na qual um agente só é admitido se ele estiver na próxima posição na ordem decrescente dos resultados dos vencedores e quando o round anterior chegar ao fim. Por exemplo, tem-se quatro agentes A,B,C e D na seguinte ordem decrescente de confiança A,C,D e B. Os dois primeiros agentes A e C entram em processo de negociação, enquanto estão negociando, o próximo da lista é o agente D. Ao sair um agente da negociação, D entra no processo e quem está na próxima posição de negociar é o agente B. A estrutura a seguir, exemplifica a **Regra de Admissão**:

```
...
Se ((roundiniciado=false) E (Posição_Agente=próximo))
Então AdmissaoAceita;
      roundiniciado=true;
...
```

No método baseado em teoria dos jogos, o resultado de um round é a escolha da ação baseada na matriz de *payoff* entre dois agentes. No final do round, existem três possíveis situações: um agente vencedor, ambos continuam ou ambos saem do processo

de negociação. O teste para verificar se o round acabou é realizado pela **Regra da Postagem**, a seguir um exemplo dessa regra:

```
...
Se    (Numero de Agentes da negociação <= 2)
Então roundiniciado=false;
      round++;
...
```

O intuito da **Regra da Validação** é garantir que a proposta de negociação está bem formada, ou seja, a estrutura da proposta deve ser condizente com a estrutura mostrada, inicialmente, nesta seção. O modelo da regra vem a seguir, exemplificando que, se apenas o campo **Objeto** da Proposta for vazio, a proposta de negociação é rejeitada:

```
...
Se    ((Identificador da proposta=falso) OU
      (Agente que propõe a Ação=vazio) OU
      (Objeto da Proposta=vazio))
Então Proposta MalFormada;
Senão IniciarNegociação;
...
```

Ao iniciar a negociação, é definida a **Regra de Formação de Acordo**. Sua estrutura deve conter o identificador do acordo, os agentes negociadores e finalmente a execução do acordo com as ações previstas de mudar ou manter a classe escolhida. Por exemplo, a regra a seguir especifica que se o agente A e o agente B escolhem a mesma ação, ambos continuam no processo de negociação, mas seus valores de confiança são trocados pelos valores de *payoff*.

```
...
IdentificadorAcordo=x;
Se    ((AgenteA(payoffA,ConfiancaA)=Ação_Mudar) E
       (AgenteB(payoffB,ConfiancaB)=Ação_Mudar))
Então ConfiancaA=payoffA;
      ConfiancaB=payoffB;
      Continuarnegociação;
...
```

Como já foi mencionado, a **Regra de Atualização** descreve como os parâmetros da negociação se modificam na ocorrência de certos eventos. Neste método de negociação, o parâmetro a ser alterado é o valor de *payoff* dos agentes. A estrutura da regra é mostrada a seguir:

```
...
AgenteA(payoffA,ConfiancaA);
Se (IdentificadorAcordo=x)
Então ConfiancaA=payoffA;
      ConfiancaB=payoffB;
...
```

O objetivo da **Regra de Mostragem** é apresentar o agente vencedor, ou seja, o agente que tem a maior confiabilidade. A estrutura da regra é mostrada logo abaixo:

```
...
Se (Confiablidade_AgenteA > Confiablidade_AgenteB)
Então AgenteA=Vencedor;
      Ação=Ação_AgenteA;
...
```

Quando se tenta vários acordos em uma negociação e não obtém sucesso em nenhum, ou após um número pré-definido de rounds não conseguir chegar a um acordo, aplica-se a **Regra de Desempate**. O exemplo abaixo ilustra se o número de rounds atingiu um valor x significa que o agente A é o agente vencedor.

```
...
Se (rounds=x)
Então AgenteA=Vencedor;
...
```

A negociação termina quando o método está inativo por um tempo maior que o especificado ou, quando um acordo é formado. Isto é verificado pela **Regra de Encerramento**, cujo exemplo é ilustrado a seguir:

```
...
Se    ((Negociação> Tempo_Especificado) OU
      (Identificador do Acordo=True))
Então Negociação=Fim;
```

5.1.2 Adaptação da Ontologia Baseada em Argumento para o Método de Argumentação

Como já foi mencionado, as ontologias para negociação descritas no Capítulo 4 devem ser adaptadas em sua estrutura, a fim de que os agentes do sistema NeurAge possam utilizá-las, formalmente, no processo de negociação. A ontologia para negociação proposta em Sierra et al (1997) utiliza o modelo de persuasão como forma de

negociação entre agentes. Um dos métodos de negociação utilizados pelo sistema NeurAge é a argumentação entre seus agentes, portanto optou-se por adaptar aquela ontologia a esse método.

O diagrama mostrado na Figura 5.2 representa a estrutura e relações das classes que a ontologia proposta em Sierra et al (1997) devem possuir. Como o trabalho original não apresentava tal diagrama, surgiu a necessidade de se modelar os elementos que compõem a ontologia buscando facilitar o seu processo de implementação.

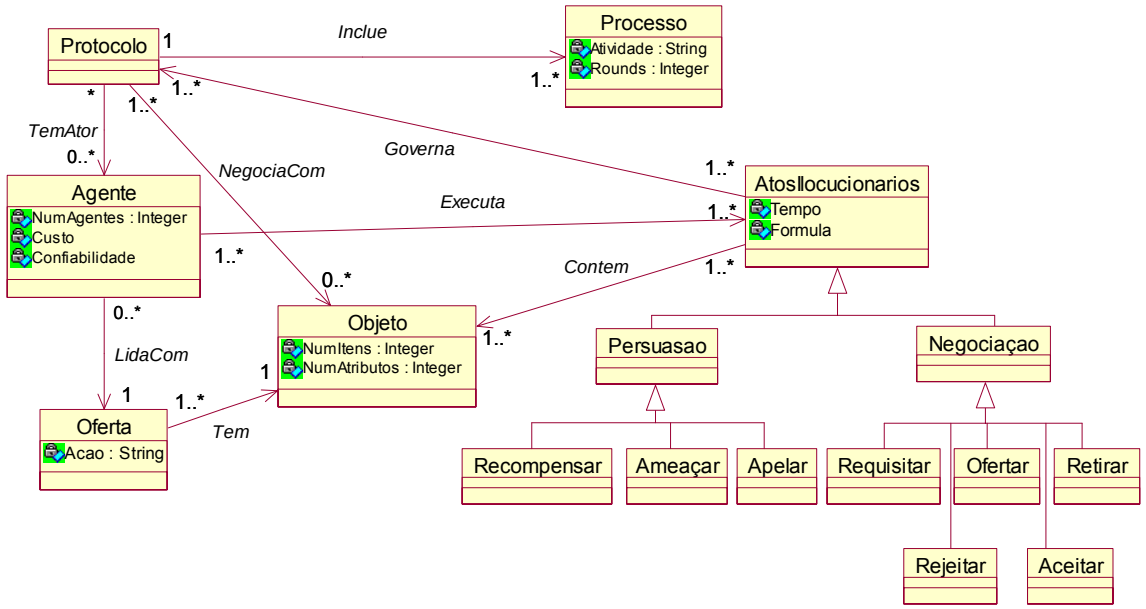

Figura 5.2: Diagrama de Classes da Ontologia baseada em argumento

Conforme explicado na Seção 4.3.2.2, a negociação ocorre por troca de atos ilocucionários, divididos em dois conjuntos:

I_{nego} = {**`ofertar,requisitar,aceitar,rejeitar,retirar-se`**} e

I_{pers} = {**`apelar,ameaçar,recompensar`**}. Baseado no funcionamento do método de argumentação do sistema NeurAge, não se sentiu a necessidade de acrescentar e/ou excluir alguma partícula de negociação e persuasão. Por outro lado, a estrutura da partícula **`requisitar`**(a,b,δ,t) necesitou ser modificada,onde:

a - agente persuasor da negociação;
b - agente persuadido da negociação;
δ - oferta que o agente a deseja efetuar com o agente b;
t - tempo da negociação.

O elemento δ foi retirado, pois um agente *a* ou *b* solicita que se inicie o processo de negociação e não está requisitando um acordo especifico. As demais partículas permanecem com sua estrutura inicial.

Os agentes negociam através de ilocuções em *CL*, iniciando com uma proposta de acordo com a partícula **`ofertar`** ou **`requisitar`**. Em seguida, são trocadas contra-propostas (que os agentes podem **`rejeitar`**) e algumas ilocuções persuasivas. No protocolo de negociação ilustrado na Figura 5.4, a persuasão é descrita por *Argumentar* que pode ser construída com uma das seguintes partículas: **`ofertar`**, **`ameaçar`**, **`recompensar`**, **`apelar`**, entre os agentes A e B. A negociação é encerrada quando é expressa uma ilocução de encerramento, isto é, **`aceitar`** ou **`rejeitar`**. As ilocuções **`aceitar`** e **`rejeitar`** sempre se referem a última proposta. O termo *tempo* foi omitido nas ilocuções.

A seguir é ilustrado um possível processo de negociação. Um agente *a* oferece ao agente *b* que ele mude sua classe para a classe escolhida pelo agente *a*. Porém, o agente *b* rejeita a oferta do agente *a* e oferece outra proposta ao agente *a*. O agente *a* aceita a proposta do agente *b* e se retira da negociação. O processo descrito, anteriormente, está descrito na seqüência de passos apresentado na Figura 5.3.

`Inicio Negociação`
`ofertar`(*a,b,mudar,t*)
`rejeitar`(*b,a,mudar,t*)
`ofertar`(*b,a,mudar,t*)
`aceitar`(*a,b,mudar,t*)
`retirar-se`(*a,b,t*)
`Fim Negociação`

Figura 5.3: Exemplo do processo de negociação entre dois agentes *a* e *b*

O diagrama de atividades apresentado na Figura 5.4 foi definido a partir da máquina de estados do protocolo ilustrado em Sierra et al (1997). A escolha por esse tipo de diagrama se deve ao fato de que o mesmo mostra o fluxo de controle de uma atividade para outra, modelando os aspectos dinâmicos do sistema.

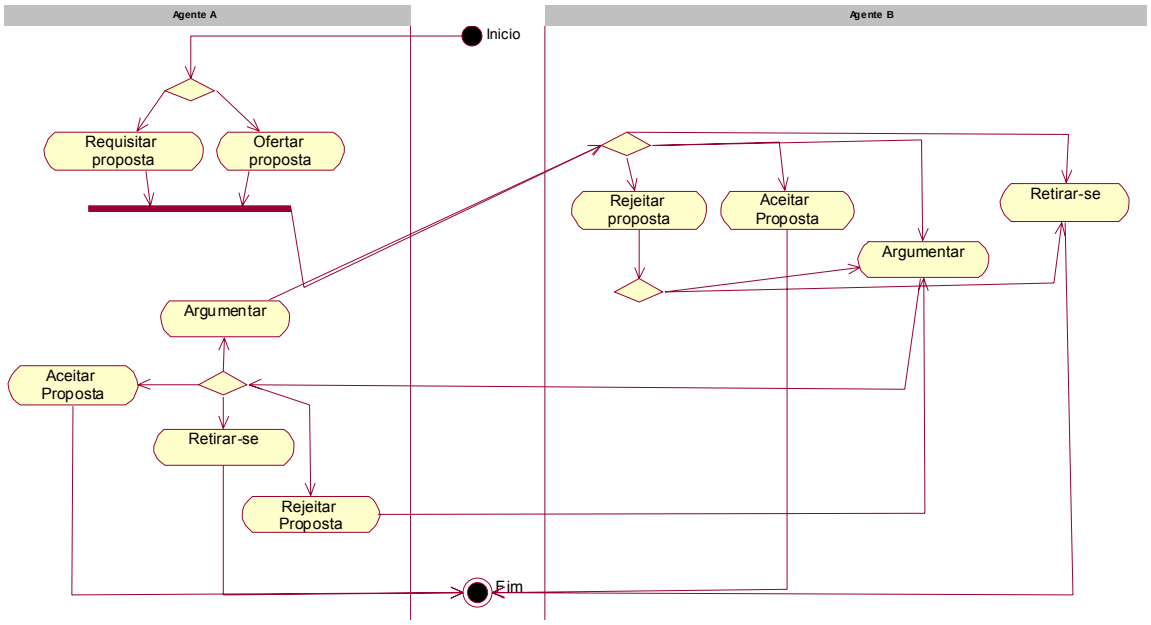

Figura 5.4: Diagrama de Atividades do Protocolo

5.1.3 Adaptação da Ontologia entre Agentes de Informação ao Método de Negociação Leilão

Assim como em Tamma, Wooldridge and Dickinson (2002) e Sierra et al (1997), a ontologia proposta em Bailin and Truszkowski (2001) necessita ser alterada em seu funcionamento, para que seja possível trabalhar com as informações fornecidas pelos agentes do sistema NeurAge, especialmente, no que se refere aos métodos de negociação utilizados por este sistema. Como já foi visto no Capítulo 4, a ontologia proposta em Bailin and Truszkowski (2001) é baseado em um conjunto de objetos, operações, estados e transições do protocolo de negociação, que necessitam ser adaptados a fim de trabalhar com o método de negociação de leilão utilizado no NeurAge. Foram feitas algumas adaptações na ontologia, como: incluir e retirar objetos, acrescentar operações e, conseqüentemente, os estados do protocolo também foram alterados.

A Figura 5.5 ilustra o diagrama de classes da ontologia proposta em Bailin and Truszkowski (2001) adaptada ao método de leilão utilizado no sistema NeurAge. Essa estrutura não foi definida no trabalho original, porém com o intuito de tornar a ontologia mais clara e objetiva, optou-se por descrever e introduzir o diagrama, apresentando os tipos de objetos e as operações que os agentes executam no protocolo de negociação.

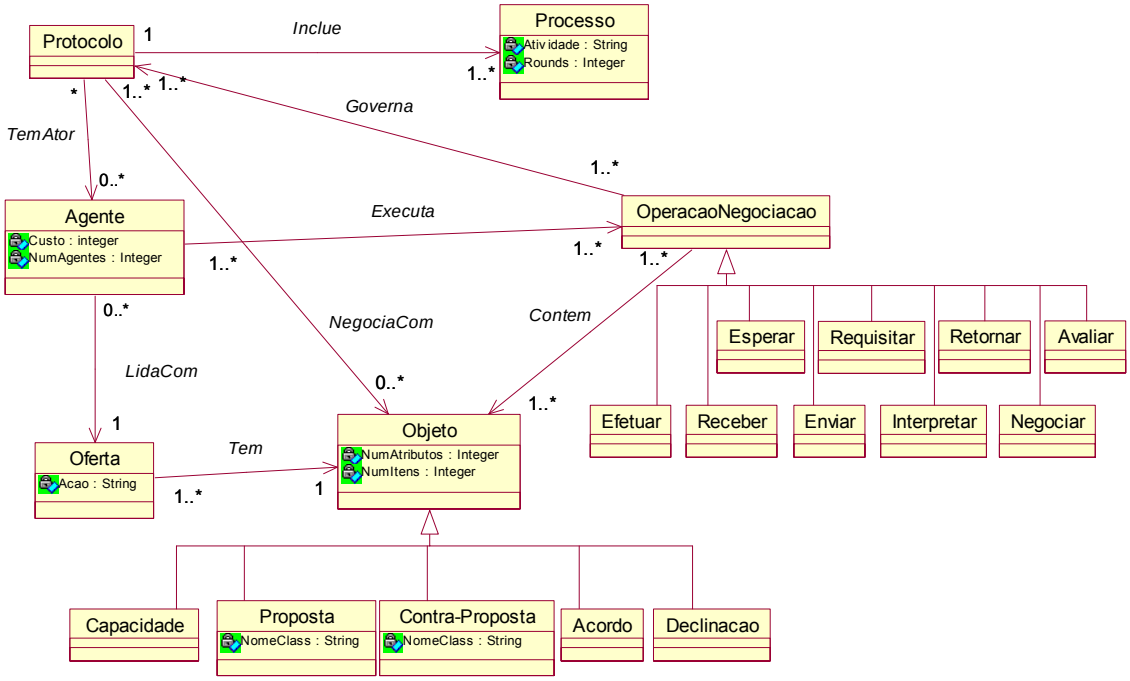

Figura 5.5: Diagrama de Classes da Ontologia adaptada ao Leilão

5.1.3.1 Objetos do Protocolo

Inicialmente, o protocolo de negociação continha 11 objetos, porém baseado no funcionamento do sistema NeurAge foram retirados nove objetos. Por exemplo, os agentes do NeurAge não realizam uma **Consulta** e sim uma **Proposta**, logo o objeto **Resultados da Consulta** também é desnecessário. **Rejeição** que expressa a insatisfação de um agente com os resultados da consulta foi trocado por **Contra-Proposta**, retornado pelo agente A se não concordar com a proposta do agente B. O agente é convidado a participar da negociação através de uma **Capacidade**, ele pode até não entrar na negociação, mas acata a decisão de todos os agentes participantes da negociação. E o **Acordo** é o objeto firmado entre os agentes da negociação. Os tipos de objetos são apresentados na Tabela 5.1.

Tipo de Objeto	Abreviação
Proposta	Pro
Contra-Proposta	Cpr
Declinação	Dcl
Capacidade	Cap
Acordo	Aco

Tabela 5.1: Objetos do Protocolo da Ontologia para Negociação

5.1.3.2 Estados e Transições do Protocolo

Como já foi visto, o protocolo em sua versão inicial era composto por onze operações. No entanto, para que a ontologia se adaptasse ao sistema NeurAge foi necessário o acréscimo e a exclusão de algumas operações reduzindo esse número para nove. As operações **Receber** e **Receber Requisição para** foram fundidas em uma só operação **Receber**, por questões de tornar a ontologia mais clara. **Avaliar**, nesse contexto, significa analisar a **Proposta** e/ou **Contra-Proposta** enviada pelo outro agente participante da negociação. A operação **Efetuar** consiste no agente A realizar um acordo com um agente B. A Tabela 5.2 apresenta as operações realizadas pelo protocolo.

Operação	Abreviação
Espera por	Esp
Requisitar	Req
Receber	Rec
Enviar	Env
Retornar	Ret
Interpretar	Int
Avaliar	Avl
Efetuar	Efe
Negociar	Neg

Tabela 5.2. Operações do Protocolo da Ontologia para Negociação

A Tabela 5.3 apresenta os possíveis estados que os agentes do NeurAge podem assumir, usando as abreviações listadas nas Tabelas 5.1 e 5.2. Por exemplo, em um leilão o agente pode **Enviar** uma **Proposta** ao outro agente, mas não pode **Enviar** uma **Contra-proposta**. Ou então, um agente pode **Receber** uma **Declinação**, mas não pode **Receber** um **Acordo**.

	Pro	Cpr	Dcl	Cap	Aco
Esp	x	x		x	
Req				x	
Rec	x	x	x	x	
Env	x				
Ret		x	x	x	
Int	x	x		x	
Avl	x	x		x	
Efe					x
Neg	x				

Tabela 5.3 Estados do Protocolo de Ontologia para Negociação

Na ontologia original, existiam 46 estados definidos no protocolo, porém, após as adaptações feitas na ontologia, os agentes do sistema NeurAge podem assumir ate 20 estados.

Em seu trabalho, Bailin and Truszkowski (2001) não ilustraram o diagrama de atividades da ontologia para negociação. Portanto, como uma forma de esclarecer o fluxo de atividades em um único processo e mostrar como as atividades são dependentes uma das outras, se optou por acrescentar esse diagrama apresentado na Figura 5.6.

Inicialmente, os agentes esperam por uma requisição de capacidade. Ao receber essa requisição, eles interpretam, avaliam e então retornam a capacidade, indicando se vai participar ou não da negociação. Com o leilão aberto, os agentes enviam suas propostas aos demais participantes. Após a interpretação da proposta, o agente pode retornar uma declinação ou avaliar a proposta. Baseado nessa avaliação, o agente efetua o acordo ou então retorna uma contra-proposta. O processo de interpretar e avaliar uma contra-proposta é semelhante ao da proposta. Finalmente, a negociação termina quando os agentes efetuam o acordo e negociam entre si.

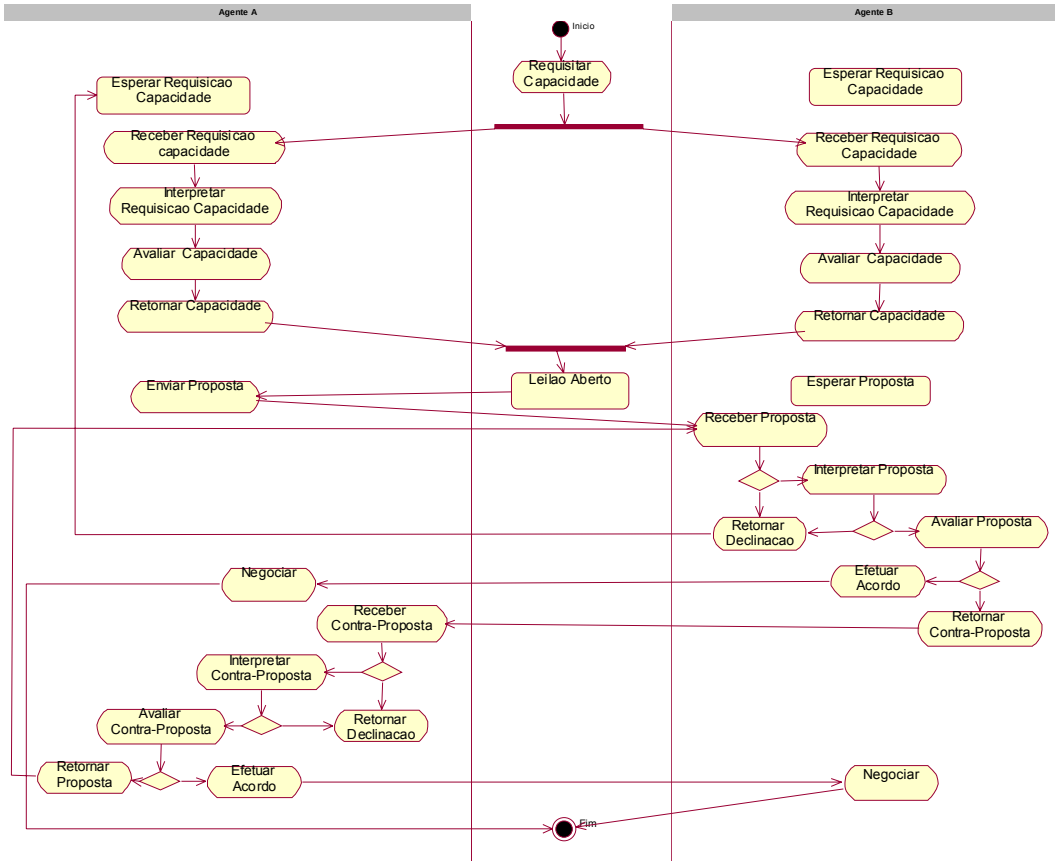

Figura 5.6: Diagrama de Atividades do Protocolo de Negociação

5.1.4.Adaptação da Ontologia entre Agentes de Informação ao Método de Negociação Baseado em Sensibilidade

Dentre as três ontologias para negociação encontradas na literatura: Tamma, Wooldridge and Dickinson (2002), Sierra et al (1997), Bailin and Truszkowski (2001), esta última foi a escolhida para ser adaptada ao método de negociação baseado em sensibilidade (Abreu 2005), em virtude dos conceitos ilustrados naquela ontologia serem os que mais se assemelham ao funcionamento do método.

Conforme estudado na Seção 4.3.3, a ontologia é baseada em um conjunto de tipos de objetos, operações, estados e transições do protocolo de negociação. O primeiro passo é modificar esta ontologia, para que a mesma possa ser utilizada em conjunto com o método de negociação baseado em sensibilidade. As alterações na ontologia consistem em: acrescentar e/ou excluir objetos e operações e, conseqüentemente, os estados do protocolo também foram alterados.

A estrutura apresentada na Figura 5.7 consiste no diagrama de classes da ontologia proposta em Bailin and Truszkowski (2001), adaptada ao método de negociação baseado em sensibilidade utilizado no sistema NeurAge. A importância de acrescentar este diagrama consiste em descrever os vários tipos de objetos no sistema e o relacionamento entre eles, proporcionando uma visão mais ampla dos conceitos ilustrados na ontologia. É importante observar que a as Figuras 5.5 e 5.7 são bastante semelhantes, haja vista ser a mesma ontologia adaptada a métodos de negociação diferentes: leilão e baseado em sensibilidade, respectivamente. No entanto, as classes **Objeto** e ***AgentAction*** sofreram algumas alterações que serão explicadas na seção seguinte.

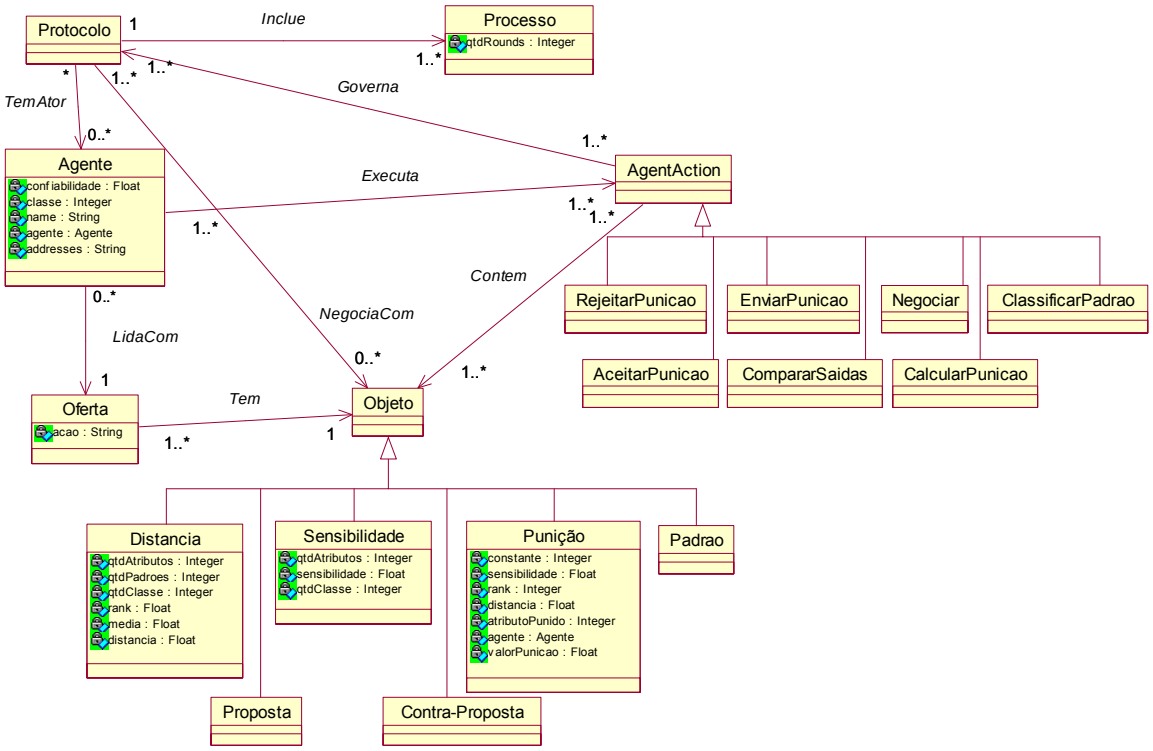

Figura 5.7: Diagrama de Classes da Ontologia adaptada ao método sensibilidade

5.1.4.1 Objetos do Protocolo

As modificações necessárias para que a ontologia se adapte ao método baseado em sensibilidade são semelhantes às que foram realizadas para o método de leilão na Seção 5.1.3. Entretanto, houve a necessidade de alterar a classe **Objeto**, fundamentado nas informações necessárias para o funcionamento do método baseado em sensibilidade.

Dentre as alterações realizadas, uma delas é incluir o objeto **Punição** o qual é enviado de um agente A ao agente B para diminuir sua confiabilidade. Outros objetos como **Distancia** e **Sensibilidade** foram acrescentados na ontologia adaptada de Bailin and Truszkowski (2001). Os tipos de objetos são apresentados na Tabela 5.4.

Tipo de Objeto	Abreviação
Padrao	Pad
Proposta	Pro
Contra-Proposta	Cpr
Punição	Pun
Distancia	Dis
Sensibilidade	Sen

Tabela 5.4: Objetos do protocolo de ontologia adaptada ao método de sensibilidade

5.1.4.2 Estados e Transições do Protocolo

Da mesma forma que, para o método de leilão, os estados e as operações da ontologia necessitam ser alterados fundamentados no funcionamento do método baseado em sensibilidade. Durante a execução do método baseado em sensibilidade é necessário **Calcular** algumas variáveis como análise de sensibilidade, média dos atributos, diferença entre atributos do padrão de entrada e a média. Esses valores são necessários para o cálculo do objeto punição, que será enviado ao agente, como uma forma de convencê-lo de que seu resultado não está satisfatório. As operações da ontologia que foram adaptadas ao método baseado em sensibilidade são ilustradas na Tabela 5.5.

Operação	Abreviação
Classificar	Cls
Comparar	Com
Negociar	Neg
Calcular	Cal
Enviar	Env
Aceitar	Act
Rejeitar	Rej

Tabela 5.5. Operações do protocolo de ontologia adaptada ao método de sensibilidade

Utilizando as abreviações demonstradas nas Tabelas 5.4 e 5.5 é possível definir na Tabela 5.6 os prováveis estados que os agentes do NeurAge podem assumir. Por exemplo, no método baseado em sensibilidade, o agente pode **Calcular** uma **Punição**, mas não pode **Calcular** uma **Proposta**. Ou ainda, um agente pode **Negociar** uma **Proposta**, mas não pode **Negociar** uma **Punição**.

	Pad	Pro	Cpr	Pun	Dis	Sen
Cls	x					
Com					x	
Neg		x	x			
Cal				x	x	x
Env		x	x	x		
Act		x	x	x		
Rej		x	x	x		

Tabela 5.6 Estados do protocolo de ontologia adaptada ao método de sensibilidade

Na ontologia original, existiam 46 estados definidos no protocolo, porém após as adaptações feitas na ontologia os agentes do sistema NeurAge podem assumir até 16 estados.

A fim de descrever a seqüência de atividades da ontologia, a Figura 5.8 apresenta o diagrama de atividades que fornece suporte ao comportamento condicional e paralelo das atividades executadas no protocolo.

Inicialmente, são calculados a análise de sensibilidade para os atributos do padrão, a média dos atributos, a diferença entre os atributos do padrão e sua média e a ordenação dos atributos em ordem decrescente da diferença calculada por classe. Se todos os agentes derem o mesmo resultado, não há negociação e o processo é encerrado. Caso contrário, a negociação começa com os agentes mostrando aos outros agentes que seus resultados não são tão bons. A partir daí, de acordo com a ordenação dos atributos, escolhe-se um agente para começar a negociar. Então, uma mensagem é enviada ao agente, sugerindo uma diminuição (punição) na sua confiabilidade. O processo continua até que todos os atributos sejam analisados ou que o usuário especifique uma quantidade de rounds. O agente classificador com o mais alto grau de certeza em sua resposta é dito ser o mais capacitado para classificar o padrão e a sua saída é considerada a saída correta.

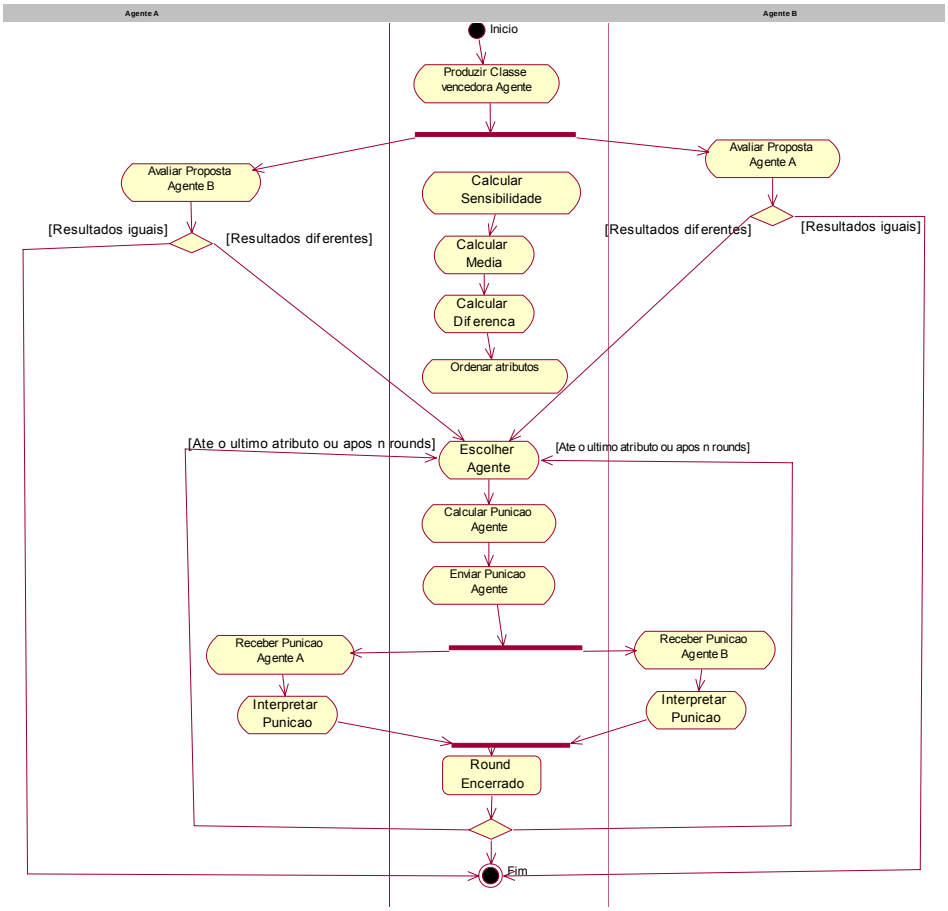

Figura 5.8: Diagrama de Atividades do Protocolo de Negociação do NeurAge

5.2 Resumo do Capítulo

Neste capítulo, foram apresentadas as adaptações nas ontologias propostas em Tamma, Wooldridge and Dickinson (2002), Sierra et al (1997), Bailin and Truszkowski (2001), associadas aos quatro métodos de negociação utilizados no sistema NeurAge. Dentre algumas alterações, destacam-se: a inclusão/exclusão de alguns conceitos/objetos da ontologia, a adição de um relacionamento entre dois conceitos, inclusão/exclusão de papéis e regras de negociação. Também foram mostrados diagramas de classes e do fluxo de atividades de cada ontologia, com o intuito de fornecer uma maior compreensão dos métodos de negociação para as quais as ontologias foram adaptadas.

Capítulo 6

Validação da Ontologia com o Método de Negociação Baseado em Sensibilidade

Neste capítulo, é mostrado inicialmente, como a ontologia foi validada, para tal será analisado sua respectiva modelagem, as classes desenvolvidas, os agentes implementados e as regras de negociação governadas pela ontologia. Na seqüência, são apresentados os experimentos que foram realizados e quatro cenários de uma tarefa de reconhecimento de padrões, exemplificando o funcionamento do sistema em conjunto com os conceitos definidos na ontologia. O objetivo é mostrar que o funcionamento do NeurAge está de acordo com o que foi definido na ontologia.

6.1. Modelagem da Ontologia

Dentre as quatro ontologias formalizadas, a escolhida para ser validada foi àquela adaptada ao método de negociação baseado em sensibilidade. Em virtude desse método ter sido proposto por Abreu (2005) e haver vários trabalhos relacionados a este método. A adaptação da ontologia de Bailin and Truszkowski (2001) para o método de negociação baseado em sensibilidade foi guiada pelo diagrama de classes ilustrado na Figura 5.7. O sistema NeurAge foi implementado por Santana (2005), utilizando o JADE e a ontologia foi modelada através do editor de ontologias Protégé (Protégé 2005). Então, com o intuito de analisar sua consistência, a ontologia foi exportada para a plataforma JADE através do plug-in *beangenerator* (Van Aart et al 2002). Em seguida, baseado no trabalho de Santana (2005) e utilizando alguns de seus conceitos, foram implementadas as regras de negociação da ontologia necessárias para que os agentes do NeurAge alcancem um acordo final entre eles.

6.1.1. Suporte JADE para Ontologia

No Protégé, uma ontologia pode ser armazenada em vários formatos, por exemplo, em RDF, XML, OWL ou em seu próprio formato. O último inclui dois arquivos: um *.pont* para classes e suas relações e outro *.pins* que contém instâncias. Os agentes JADE representam sua ontologia em objetos Java. Após uma vasta busca na literatura, em busca de como gerar a ontologia modelada no Protégé em código java, foi encontrado o plug-in *beangenerator.*

O plug-in foi desenvolvido por C.J. van Aart na Universidade de Amsterdam. Com esse plug-in uma ontologia modelada no Protégé pode ser desenvolvida e exportada para classes Java. Em particular, Java *Beans*. Um Java *Beans* é um tipo especial de classe Java, contém membros que podem ser escritos com uma operação set() e pode ser lido com uma operação get() ou is (Van Aart et al 2002).

6.2. Representação da Ontologia utilizando Protégé, *Beangenerator* e JADE

Como já foi mencionado, o Protégé armazena uma ontologia em vários formatos enquanto que no JADE os agentes representam a mesma como objetos Java. A ferramenta Protégé juntamente com o plug-in *beangenerator,* permite definir o domínio

da ontologia, utilizando uma interface gráfica do Protégé e o template JADE do *beangenerator*. O template *SimpleJADEAbstractOntology* é fornecido juntamente com o plug-in *beangenerator* e importado junto com sua estrutura de classes pré-definidas: *Concept*, *AgentAction* e *Predicate.* Assim, a ontologia deve ser alterada, acrescentando classes necessárias como subclasses desses construtores, conforme o funcionamento do método de negociação baseado em sensibilidade. A Figura 6.1 ilustra o template *SimpleJADEAbstractOntology* juntamente com os conceitos referentes ao funcionamento do método baseado em sensibilidade. Tais elementos são definidos a seguir:

- *Concepts:* Entidades que podem ser definidas em termos de slots, por exemplo Objeto,Proposta.
- *AgentActions:* Conceitos especiais que indicam ações a serem realizadas por alguns agentes, como ClassificarPadrao, AceitarPunicao.
- *Predicates*: Relacionamentos entre *concepts*, exemplo ParticipaEm, Executa, temAtor.

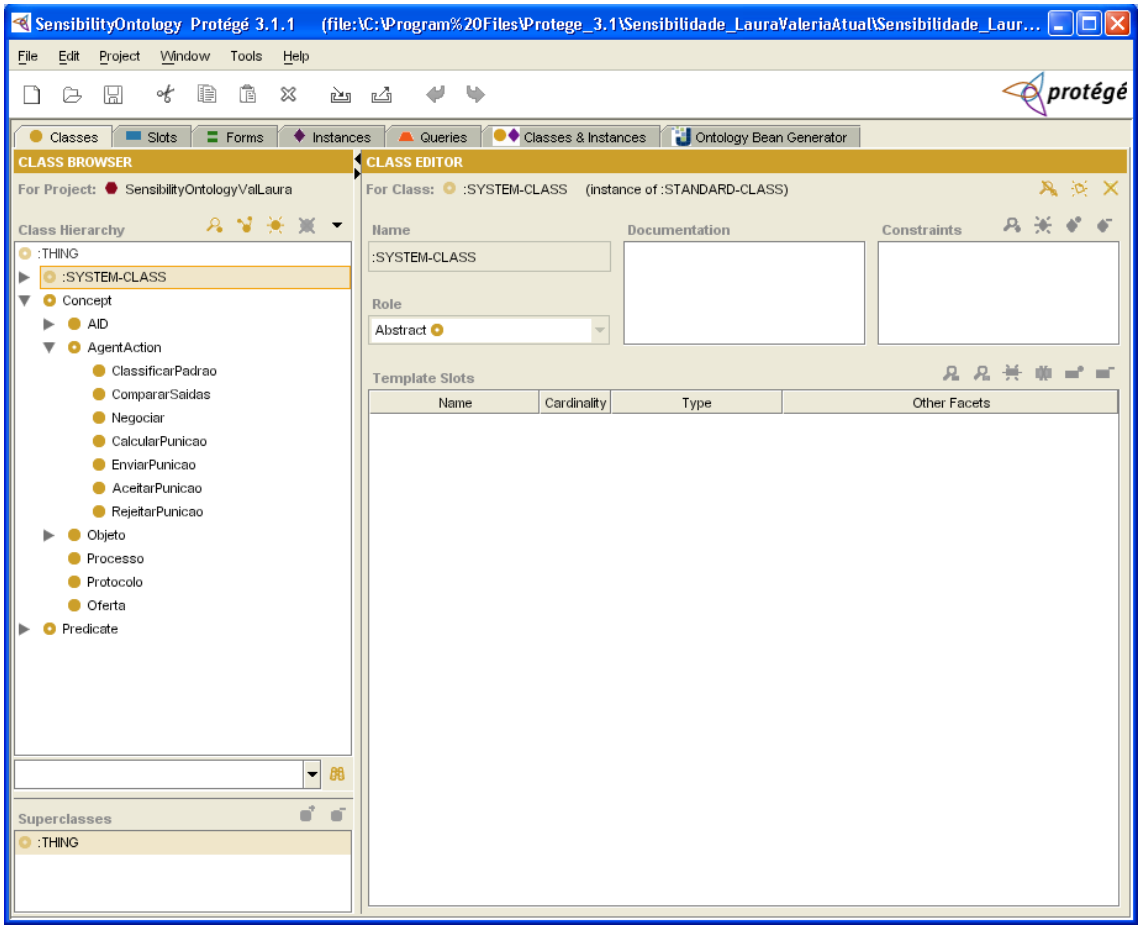

Figura 6.1: Taxonomia dos *Concept*, *AgentAction* e *Predicate*

As ontologias geradas pelo *beangenerator* definem apenas um vocabulário de domínio para troca de mensagens entre agentes, mas não definem um formato de armazenamento persistente para as instâncias dessa ontologia.

No JADE, o pacote *jade.content.onto* fornece as classes básicas *Ontology*, *Concepts*, *AgentActions* e *Predicates*. Para cada ontologia, é criada uma classe que estende a classe *jade.content.onto.Ontlogy*. Essa classe define basicamente os *Concepts, AgentActions* e *Predicates*, e as relações entre eles. A partir daí, é definida uma classe para cada *Concepts, AgentActions* e *Predicates* e para cada um de seus slots são definidos métodos "set" e "get" como apresentados na Figura 6.2.

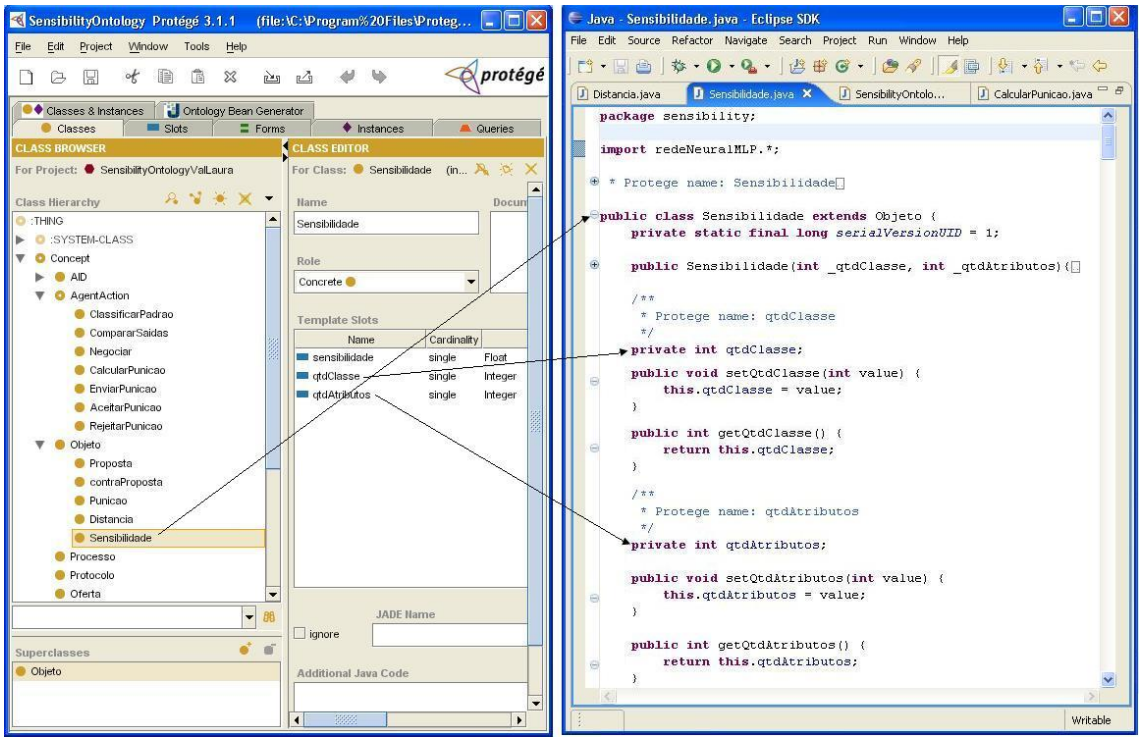

Figura 6.2 : Definição dos métodos set/get para slots

6.3 Implementação dos Agentes do NeurAge

O sistema multiagente NeurAge para classificação de padrões foi implementado por Santana (2005), utilizando a plataforma para desenvolvimento multiagente JADE. Um único tipo de agente foi implementado no sistema, chamado *Agente Classificador*.

O sistema é composto por N *agentes classificadores*. Esses agentes terão como funcionalidades:

- receber um padrão de entrada;
- gerar uma resposta sobre qual classe, dentre as classes possíveis, o padrão pertence, analisando os atributos ou características do padrão;
- ser capaz de negociar com os outros agentes, a fim de decidir quem é o mais apto a dar a resposta final do sistema, utilizando-se do método de negociação descrito no na Seção 3.2.4.

6.3.1. A Arquitetura dos Agentes

As classes do sistema que compõem a arquitetura do agente, foram baseadas no trabalho de Santana (2005), utilizando classes da plataforma JADE. As principais classes desenvolvidas, de acordo com a arquitetura descrita na Seção 5.1.4, foram:

- *AgenteClassificador.java*: Essa classe estende a classe *Agent.java* de JADE, onde está descrita a arquitetura do agente propriamente dito. Essa classe foi modificada no intuito de acessar as classes referentes à ontologia que serão descritos na próxima seção.
- *Controlador.java*: Descreve o módulo Controlador da arquitetura proposta, ou seja, comunica-se com a classe TomaDecisao para produzir um resultado. Essa classe é responsável também por mudar ou não, o resultado corrente após a execução da punição sobre a confiabilidade do agente.
- *TomaDecisaoMLP.java*: Descreve o módulo de tomada de decisão da arquitetura. Acessa a rede neural. É nessa classe, onde são feitos os cálculos sobre a sensibilidade dos atributos, suas médias e ordenação.

O classificador escolhido para a tarefa de reconhecimento de padrões foi rede neural do tipo MLP. O módulo da rede neural é constituído basicamente por três classes:

- *Neurônio.java*: Essa classe implementa os neurônios existentes nas camadas da rede. Os neurônios recebem as entradas da rede, passam-nas por uma função de ativação e o resultado é transmitido para os neurônios da próxima camada. O processo é repetido até chegar na camada de saída, onde as saídas dos neurônios serão as saídas da rede.
- *Camada.java*: A classe camada contém uma quantidade de neurônios, especificado de acordo com o problema.
- *NeuralNetwork.java*: Implementa a classe principal do módulo da rede Neural.

6.4. Classes da Negociação

O pacote da ontologia chamado *sensibility*, resultado da exportação do Protégé para o JADE através do *beangenerator*, contém as classes relativas aos *Concepts, AgentActions* e *Predicates* da ontologia modelada, porém nem todas as classes contêm implementação. As classes referentes ao conceito *AgentAction* não contêm nenhuma implementação pois sua função é alterar o comportamento do agente receptor da mensagem. No entanto, as classes relativas aos Objetos Distancia, Sensibilidade e Punicao foram modificadas com o intuito de calcular a distancia do atributo do padrão de entrada em relação a sua média para a classe vencedora, sensibilidade para os atributos do padrão testado e punição, respectivamente. A principal classe do pacote *sensibility* é *SensibilityOntology*, pois contém as definições de *Concepts, AID(s), Predicates, slots* e tipos dos *slots*. As Seções seguintes apresentam as classes referentes a cada um dos conceitos modelados na ontologia.

6.4.1 Concepts

A partir de *Concept* da ontologia modelada no Protégé, foram exportados os seguintes conceitos, juntamente com suas subclasses:

- AID: identificador do agente vencedor.
- *AgentAction*: ClassificarPadrao, CompararSaidas, Negociar, CalcularPunicao, EnviarPunicao, AceitarPunicao e RejeitarPunicao.
- Objeto: Proposta, contraProposta, Punicao, Distancia, Sensibilidade.
- Processo: Não contem sub-classes
- Protocolo
- Oferta: Não contem sub-classes

As classes referentes aos conceitos AID, Processo e Oferta contém apenas métodos get() e set() com o intuito de utilizar os valores dos seus respectivos atributos. A Figura A.1 ilustra um trecho da implementação da classe referente ao conceito `Oferta`.

Como já foi mencionado, as classes que implementam os conceitos Distancia, Sensibilidade e Punição foram modificadas com a intenção de calcular: a distância do atributo do padrão de entrada em relação a sua média para a classe vencedora, sensibilidade para todos os atributos do padrão testado e punição, respectivamente. Em

Santana (2005), esses cálculos eram realizados na arquitetura do próprio agente, porém nesse trabalho o objetivo é que a ontologia governe as regras da negociação, por isso esses cálculos foram acrescentados no pacote *sensibility* através dos conceitos de `Distancia`, `Sensibilidade` e `Punição`.

A Figura A.2 apresenta a declaração do método `calculaDistancia()` entre a distância do atributo do padrão de entrada em relação a sua média para a classe vencedora. A declaração do método `setSensibilidade()` é mostrado na Figura A.3. As classes que implementam os conceitos `Distancia` e `Sensibilidade`, são invocadas através de objetos instanciados na classe `TomaDecisaoMLP` da arquitetura do agente, conforme ilustra a Figura A.4.

A classe `Punicao` no pacote sensibility foi modificada a fim de inserir o cálculo de tal conceito. Parte de sua implementação é mostrada na Figura A.5. Esta classe é chamada, ao criar um objeto do tipo `Punicao` na classe `Controlador` da arquitetura do agente, como ilustrado na Figura A.6.

6.4.1.1. AgentAction

O conceito *AgentAction* é uma sub-classe de *Concepts* na modelagem da ontologia no Protégé, no entanto, as classes referentes aos conceitos seguem a mesma hierarquia de *Concepts* dentro do pacote *sensibility*. Conforme visto anteriormente, as classes dos conceitos *AgentAction* não contêm nenhuma implementação, pois sua função é alterar o comportamento do agente receptor da mensagem. No pacote *sensibility*, a classe do *AgentAction* `EnviarPunicao` contem o código ilustrado na Figura A.7.

6.4.2 Predicate

O conceito *Predicate* se refere aos relacionamentos entre os *Concepts* que compõem a ontologia, e suas classes não contem implementação, visto que sua função é alterar o comportamento do agente receptor da mensagem. A ontologia contém nove *predicates*, sendo eles: NegociaCom, Contem, LidaCom, Governa, Executa, Tem, TemAtor, Inclue e ParticipaEm. A Figura A.8 apresenta o código relativo ao *Predicate* `NegociaCom` do pacote *sensibilty*.

6.4.3 Inicialização dos agentes e da ontologia

Ao ser inicializado, o agente registra a linguagem e a ontologia que está utilizando. Essa operação é realizada através do pacote `jade.content` que contém um conjunto de classes que suportam ontologias e linguagens de conteúdo definidas pelo usuário. As linguagens de conteúdo das mensagens são diferentes da representação interna dos agentes (classes java no caso de JADE). Entretanto, a tradução, entre a linguagem da mensagem e as classes internas dos agentes, é automática, usando a classe `jade.content.ContentManager` que trabalha com classes codec (isto é, uma instância da interface incluída no pacote `jade.content.lang`) e de ontologia (isto é, uma instância da classe de `Ontology` incluída no pacote `jade.content.onto`) para realizar as traduções de linguagem. O código mostrado na Figura A.9 ilustra o uso de `ContentManager`.

Através de `getInstance`, o agente declara uma ontologia e chama a classe *SensibilityOntology* que é a classe principal da ontologia. Essa classe é gerada pelo *beangenerator* e não foi modificada, contendo definições de *Concepts*, AID(s), *Predicates*, slots e tipos dos slots, conforme mostra a Figura A.10.

Um dos principais passos do processo de negociação é o envio da punição. Para tanto, um agente A envia uma mensagem ao agente B com o conteúdo `EnviarPunicao`, quando o agente B recebe a mensagem e verifica que a ação é uma instância de `EnviarPunicao`, ele executa o código pertinente a essa ação. O processo anterior foi criado através do conceito *AgentAction* `EnviarPunicao`, presente no pacote *sensibility*, e pode ser descrito, em parte, na Figura A.11.

6.5. Análise do Funcionamento do Sistema

Esta Seção apresenta os experimentos que foram realizados, a base de dados utilizada e quatro cenários de uma tarefa de reconhecimento de padrões, exemplificando o funcionamento do sistema em conjunto com os conceitos definidos na ontologia. O objetivo é mostrar que o funcionamento do NeurAge está de acordo com o que foi definido na ontologia.

6.5.1. Trabalho Experimental

Com o intuito de validar a ontologia adaptada de Bailin and Truszkowski (2001) ao método de negociação baseado em sensibilidade, o sistema multiagente foi testado, utilizando cinco agentes classificadores. Esses agentes classificadores foram executados utilizando classificadores neurais MLP.

6.5.2. Base de dados utilizada

Nesse trabalho, foi utilizada uma base de dados de imagens, extraída de imagens de outdoors, tirada do UCI *repository* (Blake and Merz 1998). A base possui 2310 instâncias de sete conjuntos de imagens de outdoors, que são: *brickface, sky, foliage, cement, window, path* e *grass*. As imagens foram segmentadas à mão a fim de criar uma classificação para cada instância, onde cada instância tem uma região de 3x3 associada a si, da qual foram tirados dezenove atributos.

6.5.3. Cenário ilustrativo um do método de negociação

Esta Seção ilustra um cenário, com um número fixo de agentes, cujo objetivo é avaliar o funcionamento do sistema NeurAge, observando se o mesmo está utilizando as classes e conceitos definidos na ontologia. A seguir, é apresentado um exemplo de uma tarefa de reconhecimento de padrões, juntamente com a ordem de execução do sistema, mostrando que o funcionamento do método de negociação do NeurAge está de acordo com o que foi definido na ontologia. Considerando uma tarefa de reconhecimento com três possíveis classes (A, B e C), os padrões com cinco atributos (att1, att2, att3, att4 e att5) e o Sistema Multiagente composto por dois agentes (ag1 e ag2).

De acordo com o algoritmo de negociação ilustrado na Seção 3.2.4, o primeiro passo é apresentar o seguinte padrão de teste aos agentes: 0.7; 0.4; 0.34; 0.9; 0.22. A rede neural de cada agente produz suas saídas (confiabilidade), como mostra a Tabela 6.1.

Agente1 (ag1)		**Agente2 (ag2)**	
Classe	**Confiabilidade**	**Classe**	**Confiabilidade**
A	0.9	B	0.87
B	0.3	A	0.56
C	0.25	C	0.34

Tabela 6.1: Saída dos dois agentes

Inicialmente, são fornecidos ao sistema os padrões e a quantidade de rounds desejados. Os dois agentes classificadores são inicializados. Em seguida, através de *getInstance*, cada agente declara a ontologia e chama a classe *SensibilityOntology*. Na seqüência, os agentes produzem sua respectiva saída representando o grau de confiabilidade em cada uma das classes existentes, sendo a saída inicial aquela classe que tiver o maior grau de confiabilidade inicialmente.

Seguindo para o passo 2 do algoritmo, na classe *TomaDecisaoMLP* da arquitetura do agente, são criados os objetos conceitoSensibilidade e conceitoDistancia, dos conceitos Sensibilidade e Distancia da ontologia, respectivamente. A classe do conceito Sensibilidade contém o método setSensibilidade(), que segundo o passo 2a do algoritmo da na Seção 3.3.4, deve calcular a análise de sensibilidade para todos os atributos do padrão testado, variando-se o valor de cada atributo, um de cada vez, conforme mostra a Tabela 6.2. A análise de sensibilidade para os atributos de entrada é feita excluindo e/ou variando os valores de um atributo de um padrão, não utilizado no treinamento, e analisando a variação do desempenho do método de rede neural. Se um agente tiver uma sensibilidade alta para um determinado atributo, significa que ele é muito susceptível a pequenas variações nesse atributo, caso contrário, ele é pouco susceptível a essas variações. Esta análise é realizada para os atributos de todas as classes do problema de classificação.

De acordo com o passo 2b o método calculaMedia(), na classe implementada pelo conceito Distancia, calcula a média dos valores dos atributos em relação aos padrões de treinamento, os valores são apresentados na Tabela 6.3. Partindo para o passo 2c, o método calculaDistancia() implementa a diferença absoluta dos atributos do padrão de entrada do agente em relação a sua média, anteriormente calculada, para a classe vencedora, que podem ser vistos na Tabela 6.4. Finalmente, através de ordenaAtributos(), passo 2d do algoritmo, os atributos são ordenados decrescentemente da diferença calculada anteriormente (do atributo menos similar ao mais similar) por classe, o que é ilustrado na Tabela 6.5.

Após o agente produzir sua saída em relação ao padrão de entrada, de acordo com o passo 3 do algoritmo o *AgentAction* CompararSaidas compara as saídas dos agentes, caso haja divergências entre eles o processo de negociação é iniciado.

Supondo que houve divergência entre os agentes, em que o agente ag1 produz a classe A e a saída do agente ag2 é a classe B. O quarto passo é escolher um agente para

começar a negociar primeiro. Neste exemplo o agente ag1 inicia o processo de negociação. A saída do ag2 é classe B, o atributo att1 é o menos similar à média para a classe B (Tabela 6.5). A sensibilidade de ag2 sobre o att1 é 8% e a diferença de att1 para sua média na classe B é de 0.5 (Tabela 6.4). A punição é calculada através do conceito Punicao na classe *sensibility*, que é chamado através da criação do objeto conceitoPunir na classe *Controlador* da arquitetura do agente. Com uma constante igual a 20, o valor da punição é igual a 0.2, que é subtraído da confiabilidade do ag2 sobre a classe B. De acordo com o passo 4ii, o agente ag1 envia um *AgentAction* EnviarPunicao ao agente ag2. Como só existem dois agentes, o próximo passo desse primeiro round é que o ag2 ataque o ag1. A saída do ag1 é classe A, o att5 é o atributo menos similar à média da classe vencedora. A sensibilidade do ag1 para o att5 é 6.9% e a diferença para a média de treinamento é 0.51. Novamente, a classe Punicao é chamada através da criação do objeto conceitoPunir na classe *Controlador* da arquitetura do agente. O valor da punição é calculada usando a constante 20, produzindo um valor igual a 0.176, que é subtraído da confiabilidade do ag1 sobre a classe A. Como todos os agentes foram punidos o primeiro round é encerrado.

Esse processo continua até que uma quantidade qtdRounds (definida na classe do conceito Processo do pacote *sensibility*), todos os atributos tenham sido analisados ou uma quantidade de vezes especificada pelo usuário. Também, o usuário pode definir um número de rounds, definindo o número de atributos para serem pesquisados. A escolha de Ag1 para começar o primeiro round foi aleatória. Entretanto, experimentos iniciais têm mostrado que não existe diferença quando o Ag2 começa a negociação. Finalmente, ambos os agentes podem ser sugeridos para mudar seus resultados se, por exemplo, o grau de confiabilidade for menor que 0,25. Ao final do processo de negociação, aquele agente que tiver com o maior grau de confiança em sua resposta dá a resposta final do sistema.

O sistema implementado no cenário 1, explicado anteriormente, obteve um desempenho igual a 98.33%.

	att1	**att2**	**att3**	**att4**	**att5**
ag1	7.7%	0.9%	6.2%	5.3%	6.9%
ag2	8%	3.4%	10.7%	4.6%	9%

Tabela 6.2: Análise da sensibilidade dos atributos para os dois agentes

	att1	att2	att3	att4	att5
Classe A	0.4	0.5	0.25	0.84	0.73
Classe B	0.2	0.7	0.35	0.88	0.61
Classe C	0.1	0.47	0.46	0.91	0.75

Tabela 6.3: Média dos atributos em cada classes

	att1	att2	att3	Att4	att5
Classe A	0.3	0.1	0.09	0.06	0.51
Classe B	0.5	0.3	0.01	0.02	0.39
Classe C	0.6	0.07	0.12	0.01	0.53

Tabela 6.4: Distância absoluta dos atributos do padrão de teste para sua média

Classe A	att5	att1	att2	att3	att4
Classe B	att1	att5	att2	att4	att3
Classe C	att1	att5	att3	att2	att4

Tabela 6.5: Ordem decrescente dos atributos em relação às suas distâncias

6.5.4. Cenário ilustrativo dois do método de negociação

O cenário dois descreve uma situação onde a negociação inicia com quatro agentes e, em seguida um quinto agente entra durante a execução do sistema. Os cenários 1 e 2 são iguais, pois utilizam a mesma base de dados e o mesmo algoritmo de negociação, ilustrado na Seção 3.2.4. Entretanto, o número inicial de agentes é diferente, isso mostra que o sistema não é estático, permitindo a inclusão e/ou exclusão de agentes durante o processo de negociação. Nesse cenário o sistema obteve uma taxa de acerto igual a 98.07%.

6.5.5. Cenário ilustrativo três do método de negociação

No cenário três, é descrito um exemplo com um número fixo de quatro agentes. O cenário 3 é semelhante aos 1 e 2, utilizando a mesma base de dados e o mesmo protocolo de negociação baseado em sensibilidade, apresentado na Seção 3.2.4. A importância desse cenário é ilustrar que o desempenho de acerto, 97.98%, é menor que o obtido nos cenários 1 e 2, visto que o número de agentes durante a execução do sistema é menor.

6.5.6. Resultados e Análise de Desempenho

A fim de verificar se o sistema continua apresentando melhores resultados na tarefa de classificação de padrões, foram comparados os resultados de Santana (2005) com o do sistema utilizando ontologia. Inicialmente, foram realizados experimentos utilizando cinco agentes classificadores e, em seguida, se iniciou o sistema NeurAge com quatro agentes, introduzindo o quinto agente durante o processo de negociação. Como os experimentos foram realizados utilizando três cenários, em que o sistema finaliza a execução com 5 agentes, é importante comparar o desempenho com o número fixo de 4 agentes. Os resultados obtidos durante os experimentos são mostrados na Tabela 7.6

Método de Classificação	%acerto	Desvio Padrão
SMA_SemOntologia	98.33	1.11
SMA_Ontologia(4 agentes)	97.98	1.90
SMA_Ontologia(5 agentes)	98.33	1.11
SMA_Ontologia(4 agentes+1)	98.07	2.11

Tabela 7.6 Porcentagem de acerto do SMA ComOntologia X SemOntologia

No cenário onde o número de agentes é fixo, os sistemas apresentam os mesmos resultados de desempenho, porém no cenário dois em que o quinto agente entra, após o início da negociação, o sistema aumentou o percentual de erro em 0.26%. Isso se deve ao fato do quinto agente entrar no processo de negociação, quando aproximadamente metade dos padrões já terem sido testados. Conforme o esperado, o resultado da classificação do cenário 3 foi 97.98%, taxa menor que os demais cenários, pois o NeurAge contém menos agentes classificadores.

No sistema do cenário um, os agentes utilizaram 270 rounds, como cada round pune o agente baseado na sensibilidade de 1 atributo (270 /18 atributos = 15), cada atributo foi punido 15 vezes. Já no sistema que implementa o cenário dois, o quinto agente, adicionado durante o processo de negociação, entrou quando os outros quatro agentes estavam aproximadamente no round 135, que indica cerca da metade da negociação, executando o restante dos rounds.

Finalmente, os agentes executam o mesmo número de rounds. No entanto, a quantidade de rounds é determinada pelo usuário, que verifica o número de rounds que proporcionou melhores resultados. Foram feitos experimentos com 1 round, 18 rounds e sempre aumentando em múltiplos de 18, para que cada atributo fosse punido a mesma

quantidade de vezes, de acordo com a sensibilidade do agente ao atributo. Os experimentos mostraram que com 270 rounds o sistema obteve um desempenho de 98.33% para o cenário um e 98.07% para o cenário dois.

Isso nos mostra que utilizando ontologias na situação proposta, sistema multiagente para classificação de padrões, o sistema se tornou mais formal e mais modularizado, proporcionando um maior entendimento de sua implementação.

6.6. Resumo do Capítulo

Inicialmente, foi mostrada a modelagem da ontologia utilizando a ferramenta Protégé e o uso do plug-in *beangeneration* utilizado para exportar o código da ontologia para o JADE. Em seguida, se descreveu a implementação dos agentes do NeurAge e as regras de negociação utilizando a ontologia. Na seqüência, foram mostrados os experimentos realizados e uma breve descrição da base de dados utilizada. Foi apresentado, também, um exemplo de uma tarefa de reconhecimento de padrões com o intuito de ilustrar o funcionamento do sistema NeurAge em conjunto com os conceitos descritos na ontologia.

Capítulo 7

Conclusões

Por fim, nesse capítulo, mostra-se as conclusões desse trabalho, os resultados obtidos com a validação da ontologia utilizando o método de negociação baseado em sensibilidade. Finalmente, são apresentados os trabalhos futuros.

7.1. Conclusão

O sistema NeurAge (Abreu 2004), que serviu de base para o trabalho ora desenvolvido, foi proposto devido à necessidade de tornar o processo de tomada de decisão dos Sistemas Multi-Classificadores distribuído, autônomo e flexível. Em Santana (2005), o NeurAge foi implementado utilizando a plataforma para desenvolvimento multiagente JADE.

Com o surgimento do sistema NeurAge, sentiu-se a necessidade de utilizar protocolos de negociação mais específicos surgindo, assim, a idéia de adaptar métodos de negociação ao sistema multiagente para reconhecimento de padrões NeurAge. No entanto, os métodos de negociação do sistema NeurAge foram propostos de maneira informal. Assim, se percebeu que precisava formalizar os métodos de negociação, e uma maneira de formalizar a negociação é utilizando ontologias. Para tal, se fez um estudo dos métodos utilizados no sistema e, após uma pesquisa na literatura, foram encontradas várias ontologias. Então, para não iniciar o desenvolvimento de uma ontologia do início ao fim, se decidiu adaptar as ontologias já existentes aos métodos de negociação utilizados no sistema NeurAge. As adaptações nas ontologias para negociação encontradas é uma das contribuições desse trabalho.

Além de compreender o funcionamento dos quatro métodos de negociação utilizados no sistema NeurAge, outros objetivos foram estudar as ontologias para negociação propostas em Tamma, Wooldridge and Dickinson (2002), Sierra et al (1997), Bailin and Truszkowski (2001) e, adapta-las aos métodos de negociação utilizados pelo NeurAge. Assim, a ontologia proposta em Tamma, Wooldridge and Dickinson (2002) foi adaptada ao método baseado em Teoria dos Jogos, o método baseado em argumentação foi utilizado pela ontologia baseada em argumentação baseado em Sierra et al (1997) e, a ontologia descrita em Bailin and Truszkowski (2001) foi adequada aos métodos baseado em leilão e em sensibilidade, produzindo duas ontologias para negociação diferentes. Ao final das adaptações, foram obtidas quatro diferentes ontologias para negociação. Dentre algumas alterações, destacam-se: a inclusão/exclusão de alguns conceitos da ontologia, a adição de um relacionamento entre dois conceitos, inclusão/exclusão de papéis e regras de negociação.

Uma vez feita a adaptação da ontologia proposta em Bailin and Truszkowski (2001) ao método de negociação baseado em sensibilidade, fez-se a validação do sistema

NeurAge utilizando a ontologia. Inicialmente, foi mostrada a modelagem da ontologia, utilizando o editor de ontologias Protégé e o uso do plug-in *beangeneration* para exportar a ontologia para a plataforma JADE. Em seguida, se descreveu a implementação dos agentes do NeurAge e as regras de negociação utilizando a ontologia.

Após a validação da ontologia, foram ilustrados três cenários exemplificando uma tarefa de reconhecimento de padrões, descrevendo o funcionamento do sistema em conjunto com os conceitos definidos na ontologia. O sistema obteve o mesmo desempenho que o sistema implementado por Santana (2005) sem o uso de ontologia. Como resultado desta validação, podemos concluir que a utilização da ontologia formalizou o sistema NeurAge, tornando-o mais organizado e aberto, como também não dificultou na tarefa de reconhecimento de padrões. Portanto, acredita-se que o uso de ontologia foi benéfica ao sistema NeurAge.

7.2. Trabalhos Futuros

A seguir, serão apresentadas algumas sugestões de possíveis melhorias e extensões que podem ser realizadas, relativas ao estudo abordado nesta dissertação:

- Adaptar as demais ontologias aos métodos de negociação:
    - Adaptar a ontologia proposta em Tamma, Wooldridge and Dickinson (2002) aos métodos de negociação baseados em leilão, argumentação e em sensibilidade.
    - Utilizar a ontologia baseada em Bailin and Truszkowski (2001) nos métodos de negociação baseados em teoria dos jogos e argumentação.
- Validar as outras três ontologias desenvolvidas durante este traballho, bem como as construídas futuramente.
- Validar as ontologias em sistemas multiagentes :
    - maiores - por exemplo, um sistema composto por 25 agentes;
    - heterogêneos – O NeurAge é um sistema homogêneo, pois os agentes contém a mesma arquitetura interna e mesmo funcionamento. Portanto, um trabalho seria verificar o comportamento em sistemas em que os agentes possuam arquiteturas interna diferentes.

Referências Bibliográficas

Abreu, M., A. Canuto, A. Santos, V. Bezerra, F. Souza, and M. Gomes Junior. 2004. "Investigating the Use of an Agent-Based Multi-classifier System for Classification Tasks." *11th International Conference on Neural Information Processing.* 3316: 854-859.

Abreu, M. 2005. "Análise do Desempenho de Alguns Métodos de Negociação para Agentes Inteligentes Classificadores." Relatório Final de Graduação, Universidade Federal do Rio Grande do Norte.

Abreu, M., A. Canuto, and L. Santana. 2005. "A Comparative Analysis of Negotiation Methods for a Multi-neural." *Proceedings of Fifth International Conference on Hybrid Intelligent Systems (HIS'05)*. 451-456.

Abreu, M., A. Canuto, and M. Souto. 2005. "Uma análise comparativa de sistemas de classificação usando uma base de estrutura de proteínas." *Proceedings of Seventh Congresso Brasileiro de Redes Neurais*.

Bailin, S., and W. Truszkowski. 2001. "Ontology negotiation between scientific archives." *Proceedings of Thirteenth International Conference on Scientific and Statistical Database Management,* IEEE Press. 245-250.

Bailin, S., and W. Truszkowski. 2002. "Ontology Negotiation: How Agents Can Really Get to Know Each Other." *First International Workshop on Radical Agent Concepts* (WRAC). 320-334.

Bartolini, C., and C. P. N. Jennings. 2002. "A generic software framework for automated network." *Proceedings of the First International Conference on Autonomous Agent and Multi-Agent Systems*.

Bellifemine, F., G. Caire, T. Trucco, and G. Rimassa. 2003. JADE Programmer's Guide.Disponível em: http://sharon.cselt.it/projects/jade/doc/programmersguide.pdf. Acesso em: 01 de maio de 2006.

Berners-Lee, T., J. Hendler, and O. Lassila. 2001. "The semantic web." *Scientific American*. 284:34-43.

Bertolami, R., and H. Bunke. 2006. “Feature stream integration versus decision level combination in a multiple classifier system for text line recognition” *18th International Conference on Pattern Recognition.*

Bhattacharya, U., and B. Chaudhuri. 2005. "Fusion of Combination Rules of an Ensemble of MLP Classifiers for Improved Recognition Accuracy of Handprinted Bangla Numerals." *8th International Conference on Document Analysis and Recognition* (ICDAR'05). 322-326.

Blake, C.L., and C.J. Merz. 1998. *UCI Repository of machine learning databases*. Irvine, CA: University of California, Department of Information and Computer Science. Disponivel em: http://www.ics.uci.edu/~mlearn/MLRepository.html. Acesso em: 15 de maio de 2006.

Bonissone, P., N. Eklund, and K. Goebel. 2005. “Using an Ensemble of Classifiers to Audit a Production Classifier.” *Proceedings of the 6th International Workshop on Multiple Classifier Systems*. 376-386.

Booch, G., J. Rumbaugh, and I. Jacobson. 1999. *The Unified Modeling Language User Guide*. Addison--Wesley, Reading, Massachusetts, USA, 1 ed.

Brandão, A., L. Martimiano, and E. Moreira. 2004. “O Uso de Ontologia em Alertas de Vulnerabilidades.” *22º Simpósio Brasileiro de Redes de Computadores*. 75-86.

Brickley, D., and R.V. Guha. 2002. “RDF Vocabulary Description Language 1.0: RDF Schema.” W3C Working Draft. Disponível em: http://www.w3.org/TR/PR-rdf-schema. Acesso em: 13 de novembro de 2005.

Carabelea, C. 2002. “Adaptive Agents in Argumentation-Based Negotiation.” Springer Verlag LNAI series.

Canuto, A. M. de P. 2001. “Combining Neural Networks and Fuzzy Logic for Aplications in Character Recognition.” Doctor´s thesis, Department of Electronics, University of Kent.

Canuto, A., A. Campos, J. Alchiere, E. Moura, A. Santos, E. Santos, and R. Soares. 2005. “A personality-based model of agents for representing individuals in working organizations.” *IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT´05)*. 65-72.

Chawla, N. V., and K. Bowyer. 2005. “Ensembles in face recognition: Tackling the extremes of high dimensionality, temporality, and variance in data.” *Proceedings of IEEE Conference on Systems, Man and Cybernetics.*

Cho, S-B. 2004. “The evolution of conversation.” Disponível em: http://www.ee.furg.br/~silviacb/SIIG/apl.htm. Acesso em: 12 de outubro de 2005.

Cho, S-B. 1999. “Pattern Recognition with Neural Networks combined by Genetic. Algorithm.” *Fuzzy Sets and Systems*, 103(2): 339-347

Corcho, O., M. Fernández-López, and A. Gómez-Pérez. 2003. “Methodologies, tools and languages for building ontologies. Where is their meeting point?” *Journal Title: Data Knowledge Engineering*, 46:41 – 64.

Czyz, J., J. Kittler, and L. Vandendorpe. 2004. “Multiple classifier combination for face-based identity verification”. *Pattern Recognition* 37 (7): 1459—1469.

Evangelista, P., P. Bonissone, M. Embrechts, and B. Szymanski. 2005. "Unsupervised Fuzzy Ensembles Applied to Intrusion Detection" *Proceedings of the 13th European Symposium on Artificial Neural Networks*.

Ferber, J. 1999. *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence.* 1ª edição, Addison-Wesley Pub Co.

Field, S., M. Stolze, and M. Stroebel. 2000. “The London Classification: 1st e-negotiations Workshop—Negotiations Beyond Price.” In: DEXA Workshops.

Fipa.org. 2002. “The Foundation for Intelligent Physical Agents.” Disponível em: http://www.fipa.org/. Acesso em: 14 de março de 2006.

Fonseca, J. M. M. R. 2000. “Protocolos de Negociação com Coligações em Sistemas Multiagente – Uma aplicação à Gestão Distribuída de Recursos.” Doctor´s thesis. Universidade de Nova Lisboa, Portugal.

Freitas, R., and R. Torres. 2005. “OntoSAIA: Um ambiente Baseado em Ontologias para Recuperação e Anotação Semi-Automática de Imagens.” *20º Simpósio Brasileiro de Banco de Dados*. 60-79.

Gruber, Tom. 2003. “What is an Ontology?” Disponivel em: http://www-ksl.stanford.edu/kst/what-is-an-ontology.html. Acesso em: 12 de setembro de 2005

Hamdi, Mohamed. 2006. "MASACAD: A Multiagent-Based Approach to Information Customization." *IEEE Intelligent Systems*, 21: 60-67.

Haykin, S. 1998. *Neural Networks, A Comprehensive Foudation*. Prentice Hall, Second Edition.

Heutte, L., A. Nosary, and T. Paquet. 2004. "A multiple agent architecture for handwritten text recognition." *Pattern Recognition* 37(4): 665-674.

JADE, Cselt. 2005. Disponivel em: http://sharon.cselt.it/projects/jade. Acesso em: 23 de janeiro de 2006

JADE.2003. JADE Brasil-Nucleo de JADE Brasileiro. Disponível em: http://qos.tecnolink.com.br/doc/Ontologia.htm. Acesso em: 15 de setembro de 2005

Jennings, N.R., P. Faratin, M.J. Johnson, T.J. Norman, P. O`Brien, and W.E. Wiegand. 1996. "Agent-based business process management." *Proceedings of International Journal of Cooperative Information Systems*. 105-130.

Kajarekar, S., L. Ferrer, E. Shriberg, K. Sonmez, A. Stolcke, A. Venkataraman, and J. Zheng. 2005. "SRI's 2004 NIST speaker recognition evaluation system". *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP´05)*. 1:173-176.

Kraus, S., K.Sycara, and A.Evenchik. 1998. "Reaching agreements through argumentation: a logical model and implementation." *Artificial Intelligence*. 104: 1-69.

Lassila, O., and R. Swick. 1999. "Resource description framework (RDF) model and syntax specification." W3C Recommendation, disponivel em: http://www.w3.org/TR/REC-rdf-syntax/. Acesso em 01 de dezembro de 2005.

Lassila, O., and D. McGuinness. 2001. "The role of frame-based representation on the semantic web." *Electronic Transactions on Artificial Intelligence (ETAI) Journal: area The Semantic Web.*

Li, P., K. Chan, and S. Krishnan. 2005. "Learning a Multi-Size Patch-Based Hybrid Kernel Machine Ensemble for Abnormal Region Detection in Colonoscopic Images." *IEEE Computer Society Conference on Computer Vision and Pattern Recognition,* 2: 670-675.

Li, P. , K. Chan, S. Fu and S. Krishnan. 2005. “An Abnormal ECG Beat Detection Approach for Long-Term Monitoring of Heart Patients Based on Hybrid Kernel Machine Ensemble.” *Proceedings of 6th International Workshop Multiple Classifier Systems*. 346-355.

Lomuscio, A., M. Wooldridge, and N. Jennings. 2000. “A classification scheme for negotiation in electronic commerce.” In F. Dignnum and C. Sierra, editors, *Agent mediated electronic commerce: a European perspective.* 19-33. Springer-Verlag, Berlim.

Matsubara, Shigeo.2005. “Auction in dynamic environments: incorporating the cost caused by re-allocation.” *4th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS´05)*. 643 - 649

Medeiros, R., and A. Belchior. 2004. “Uma Ontologia para Engenharia de Requisitos de Software”. *3º Simpósio Brasileiro de Qualidade de Software*.

Meng, I-Heng, S. Tseng, and Y. Wei-Pang. 2004. “Integrated Business Agents for Processing Free Text Information.” *Journal of Information Science and Engineering* 20: 325-347.

Morency, L., C. Sidner, and T. Darrell. 2005. “Towards context-based visual feedback recognition for embodied agents.” *Proceedings of the Symposium on Conversational Informatics for Supporting Social Intelligence and Interaction (AISB'05)*. 69-72.

Mukhopadhyay, S. et al. 2003. “Large-scale Multi-agent Information Classification Using Dynamic Acquaintance Lists.” *Journal of the American Society for Information Science and Technology*. 54: 966-975.

Nadarajan, G., and Y. Chen-Burger. 2006. "An Ontology-Based Conceptual Mapping Framework for Translating FBPML to the Web Services Ontology." *Sixth IEEE International Symposium on Cluster Computing and the Grid (CCGRID'06)*. 158-165.

Noy, F., R. Fergerson, and M. Musen. 2000. “The Knowledge Model of Protégé-2000: Combining Interoperability and Flexibility”. *Proceedings of the 12th European Workshop on Knowledge Acquisition, Modeling and Management.* 17-32.

Noy, F., and D. Guinness. 2001. “Ontology development 101: a guide to create your first ontology.” Disponível em: http://ksl.stanford.edu/people/dlm/papers/ontology-tutorial-noy-mcguinness.doc. Acesso em: 2 de novembro de 2005.

Ontology.Org. 2000. Frequently Asked Questions. Acesso em: 14 de setembro de 2005.

Osborne, M. 2003. *An Introduction to Game Theory*. Oxford University Press, 2003.

Peng, S., S. Mukhopadhyay, R. Raje, M. Palakal, and J. Mostafa. 2001. “Comparison of Single-Agent and Multi-Agent Architectures for the Classification of Handwritten Text.” *Proceedings of 10th IEEE Heterogeneous Computing Workshop (HCW2001)*.

Protégé. 2005. Disponivel em: http://protege.stanford.edu/. Acesso em: 2 de fevereiro de 2006.

Qu, Y., X. Zhang, and H. Li. 2004. "OREL: an ontology-based rights expression language". *Proceedings of 13th international World Wide Web conference on Alternate track papers & posters*. 324-325.

Rech, L., R. Oliveira, and C. Montez. 2005. “Dynamic Determination of the Itinerary of Mobile Agents with Timing Constraints.” *IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT'05)*. 45-50

Rezende, S. O. 2003. *Sistemas Inteligentes: Fundamentos e Aplicações*. 1ª edição. Ed. Manole.

Ribeiro,I., R. Sousa, and M. Girardi. 2004. “Uma Ontologia para a Especificação de Sistemas de Padrões.” *4th Latin American Conference on Patterns Languages of Programming*. 281-290.

Santana, L. E. A. 2005. “Implementação de um Sistema Multi-Agente para Classificação de Padrões utilizando a Plataforma JADE.” Relatório Final de Graduação, Universidade Federal do Rio Grande do Norte.

Saranli, A., and M. Demirekler. 2001. "A Statistical Unified Framework for Rank-Based Multiple Classifier Decision Combination." *Pattern Recognition*, 34: 865-884.

Saranli, A., and M. Demirekler. 2000. "A Unified View of Rank-Based Decision Combination." *Proceedings of the 15th International Conference on Pattern Recognition*, 2:479-482.

Schüller, B., S. Reiter, R. Mueller, M. Al-Hames, M. Lang, and G. Rigoll. 2005. "Speaker independent speech emotion recognition by ensemble classification." *IEEE International Conference on Multimedia & Expo (ICME2005)*, 864-867.

Sierra, C.; N. R. Jennings, P. Noriega, and S. Parsons.1997. "A Framework for Argumentation-Based Negotiation." *Proceedings of 4th International Workshop on Agent Theories, Architectures and Languages (ATAL 97*),167-182.

Sousa, L., and J. Leite. 2005."Geração Automática de Dicionários Explicativos em Sistemas de Informações Geográficas usando Ontologia." *XXV Congresso da Sociedade Brasileira de Computação*.

Studer, R., V. Benjamins, and D.Fensel. 1998. "Knowledge engineering, principles and methods." *Data and Knowledge Engineering*, 25(1-2):161-197.

Su, X., M. Matskin, and J. Rao. 2003. "Implementing Explanation Ontology for Agent System." *Proceedings of IEEE International Conference on Web Intelligence*, 330-336.

Tamma, V., M. Wooldridge, and I. Dickinson. 2002. "An ontology for automated negotiation." *Proceedings of 1st International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS02).* 62-69

Tho, Q., S. Hui, A. Fong, and T. Cao. 2006. "Automatic Fuzzy Ontology Generation for Semantic Web." *IEEE Transactions on Knowledge and Data Engineering*, 18: 842-856.

Van Aart, C.J., R.F. Pels, C. Giovanni, and F. Bergenti. 2002. "Creating and Using Ontologies in Agent Communication." *Proceedings of 1st International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS02)*. 44-51

Vuurpijl,L., and L. Schomaker. 1998. "A framework for using multiple classifiers in a multiple-agent architecture." *Proceedings of third IEEE European Workshop on Handwriting analysis and recognition*. 1-6.

Wang, Fei-Yue. 2005. "Agent-Based Control for Networked Traffic Management Systems." *IEEE Intelligent Systems*, 20: 92-96.

Weiss, G. 1999. *Multiagent Systems. A Modern Approach to Distributed Artificial Intelligence*. Ed. MIT Press.

Wooldridge, M. 2002. *An Introduction to Multiagent Systems*. Department of Computer Science, University of Liverpool, UK: JOHN WILEY & SONS, LTD.

Wurman, P., M. Wellman, and W. Welsh. 2001. "A parametrization of the auction design space." *Games and economic behavior*. 35(1-2):304-338.

Xie, Xiao-Feng, and J. Liu. 2005. "A compact multiagent system based on autonomy oriented computing." *IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT'05)*. 38-44.

Xml.org. 2001. "The extensible markup language." Disponível em: http://www.xml.org.

Apêndice A

```
package sensibility;
import jade.content.*;
public class Oferta implements Concept {
      private static final long serialVersionUID = 1;
      private String acao;
      public void setAcao(String value) {
            this.acao = value;
      }
      public String getAcao() {
            return this.acao;
      }
}
```

Figura A.1: Implementação do conceito `Oferta`

```
public class Distancia extends Objeto {
     ...
     private void calculaDistancia(int padrao, double[][] _media) {
      ...
      }
}
```

Figura A.2: Declaração do método `calculaDistancia()`

```
public class Sensibilidade extends Objeto {
     ...
     public void setSensibilidade(MLPNeuralNetwork rede, int padrao) {
      ...
      }
}
```

Figura A.3: Declaração do método `setSensibilidade()`

```
public class TomaDecisaoMLP {
      ...
      Distancia conceitoDistancia;
      Sensibilidade conceitoSensibilidade;

      public TomaDecisaoMLP(String configRede, int qtdAtrib, int
agente, int[] fold) {
            ...
            conceitoDistancia = new Distancia(qtdClasse, qtdAtrib,
qtdPadroes, fold, rede);
            conceitoSensibilidade = new Sensibilidade(qtdClasse,
qtdAtrib);
            ...
      }
}
```

Figura A.4: Chamadas dos objetos `conceitoDistancia` **e** `conceitoSensibilidade`

```
public class Punicao extends Objeto {
     ...
     public void setPunicao() {
           this.punicao = (distancia * sensibilidade) / (rank *
               constante);
      }
...
}
```

Figura A.5: Cálculo do conceito `Punicao`

```
public class Controlador {
     ...
     Punicao conceitoPunir;
     ...
     public Controlador(String tipo, int contFold, int qtdAtrib, int
                agente, int[] fold) {
           ...
           conceitoPunir = new Punicao();
           ...
      }
}
```

Figura A.6: Criação do objeto `conceitoPunir`

```
package sensibility;
import jade.content.*;
public class EnviarPunicao implements AgentAction {
      private static final long serialVersionUID = 1;
}
```

Figura A.7: Implementação do conceito *AgentAction* `EnviarPunicao`

```
package sensibility;
import jade.content.*;
public class NegociaCom implements Predicate {
      private static final long serialVersionUID = 1;
}
```

Figura A.8: Implementação do *Predicate* `NegociaCom`

```
public class AgenteClassificador extends Agent{
...
//Registra a linguagem que será usada nesta aplicação
getContentManager().registerLanguage(codec);
//Registra a ontologia que será usada nesta aplicação
getContentManager().registerOntology(SensibilityOntology.getInstance());
...
}
```

Figura A.9: Registro da Linguagem e Ontologia utilizados pelo agente

```
Public class SensibilityOntology extends jade.content.onto.Ontology
implements ProtegeTools.ProtegeOntology {
...
private static Ontology theInstance = new SensibilityOntology();
      public static Ontology getInstance() {
            return theInstance;
      }
...
private SensibilityOntology() {
      ...
      {
      // adding Concept(s)
      ConceptSchema distanciaSchema = new ConceptSchema(DISTANCIA);
      add(distanciaSchema, sensibility.Distancia.class);}
      ...
      // adding AgentAction(s)
      AgentActionSchema negociarSchema = new
```

```
                AgentActionSchema(NEGOCIAR);
       add(negociarSchema, sensibility.Negociar.class);
      ...
      // adding AID(s)
      ConceptSchema vencedorSchema = new ConceptSchema(VENCEDOR);
      add(vencedorSchema, sensibility.Vencedor.class);
      ...
       // adding Predicate(s)
       PredicateSchema governaSchema = new PredicateSchema(GOVERNA);
       add(governaSchema, sensibility.Governa.class);
      ...
      // adding fields
      distanciaSchema.add(DISTANCIA_QTDCLASSE,
             (TermSchema) getSchema(BasicOntology.INTEGER),
              ObjectSchema.OPTIONAL);
      ...
      }
...
}
```

Figura A.10: Definições de *Concepts*, AID(s), *Predicates*, *slots* e tipos dos *slots*.

```
...
EnviarPunicao punir = new EnviarPunicao();
enviaMensagemOnto(punir, agentes[indiceAgentes], "punicao",
                         ACLMessage.PROPOSE, padraoAtual);
...
```

Figura A.11: Envio do *AgenteAction* `EnviarPunicao`

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Administração
Baixar livros de Agronomia
Baixar livros de Arquitetura
Baixar livros de Artes
Baixar livros de Astronomia
Baixar livros de Biologia Geral
Baixar livros de Ciência da Computação
Baixar livros de Ciência da Informação
Baixar livros de Ciência Política
Baixar livros de Ciências da Saúde
Baixar livros de Comunicação
Baixar livros do Conselho Nacional de Educação - CNE
Baixar livros de Defesa civil
Baixar livros de Direito
Baixar livros de Direitos humanos
Baixar livros de Economia
Baixar livros de Economia Doméstica
Baixar livros de Educação
Baixar livros de Educação - Trânsito
Baixar livros de Educação Física
Baixar livros de Engenharia Aeroespacial
Baixar livros de Farmácia
Baixar livros de Filosofia
Baixar livros de Física
Baixar livros de Geociências
Baixar livros de Geografia
Baixar livros de História
Baixar livros de Línguas

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo