UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA MECÂNICA
COMISSÃO DE PÓS-GRADUAÇÃO EM ENGENHARIA MECÂNICA

Proposta de uma rede de compartilhamento de habilidades no ambiente da Manufatura

Autor: Olga Fernanda Nabuco de Araújo
Orientadores: **João Maurício Rosário (UNICAMP Brasil)**
Khalil Drira (LAAS-CNRS França)

10/03

UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA MECÂNICA
COMISSÃO DE PÓS-GRADUAÇÃO EM ENGENHARIA MECÂNICA
DEPARTAMENTO DE PROJETO MECÂNICO

Proposta de uma rede de compartilhamento de habilidades no ambiente da Manufatura

Autor: Olga Fernanda Nabuco de Araújo
Orientadores: João Maurício Rosário – Unicamp Brasil
Khalil Drira – Laas-CNRS França

Curso: Engenharia Mecânica
Área de Concentração: Mecânica dos Sólidos

Tese de Doutorado apresentada à comissão de Pós Graduação da Faculdade de Engenharia Mecânica, como requisito para a obtenção do título de Doutor em Engenharia Mecânica.

Campinas, 2003
SP – Brasil

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

Ar15p

Araújo, Olga Fernanda Nabuco de
Proposta de uma rede de compartilhamento de habilidades no ambiente da manufatura / Olga Fernanda Nabuco de Araújo. -- Campinas, SP: [s.n.], 2003.

Orientadores: João Maurício Rosário e Khalil Drira.
Tese (doutorado) - Universidade Estadual de Campinas, Faculdade de Engenharia Mecânica.

1. Aquisição de conhecimento (Sistemas especialistas). 2. Agentes inteligentes (Software). 3. Ontologia. 4. Grupos de trabalho – Processamento de dados. I. Rosário, João Maurício. II. Drira, Khalil. III. Universidade Estadual de Campinas. Faculdade de Engenharia Mecânica. IV. Título.

UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA MECÂNICA
COMISSÃO DE PÓS-GRADUAÇÃO EM ENGENHARIA MECÂNICA
DEPARTAMENTO DE PROJETO MECÂNICO

TESE DE DOUTORADO

Proposta de uma rede de compartilhamento de habilidades no ambiente da Manufatura

Autor: **Olga Fernanda Nabuco de Araújo**
Orientador: **João Maurício Rosário**

Prof. Dr. João Maurício Rosário, Presidente
Universidade Estadual de Campinas

Prof. Dr. Paulo Corrêa Lima
Universidade Estadual de Campinas

Prof. Dr José Reinaldo Silva
Universidade de São Paulo

Prof. Dr. Osvaldo Luiz Agostinho
Universidade Estadual de Campinas

Prof. Dr. Maurício Ferreira Magalhães
Universidade Estadual de Campinas

Dedicatória

Dedico este trabalho ao meu filho querido e à minha mãe que reza desde a minha graduação por este momento.

Agradecimentos

Vou começar logo os agradecimentos de modo a acomodar uma lista quase interminável, tanto eu devo a tantos. No início no meio e no fim tive a companhia e força de N. Senhora em várias de suas manifestações: Notre Dame du Taur, direto da Eglise du Taur em Toulouse, N. Sra. De Fátima a luz que ilumina a caverna que você acha que está, N. Sra. De Lourdes com sua água benfazeja, N. Sra. Da Medalha Milagrosa da qual não me separo sempre a cata de inspiração e N. Sra Desatadora de Nós de fundamental importância para meus embaraços mentais. Sem Ela, nada teria sido possível, a começar pelo meu orientador, Professor João Maurício Rosário, depositar sua confiança no meu trabalho e me enviar para outro laboratório que ele achou ser útil na minha formação. Continuando com a recepção em Toulouse pelo professor Drira que por sua vez abriu as portas do Laas proporcionando uma experiência inigualável de trabalhar no projeto DSE, de engenharia cooperativa distribuída (bem no alvo!). A intervenção fundamental da minha amiga Janette Cardoso me acolhendo em Toulouse. Aqueles brasileiros incríveis da comunidade do Laas e a própria comunidade do Laas, rica em conhecimento e calor humano. E o que dizer dos meus amigos do CenPRA? Fundamentais. O Edeneziano Dantas e seu trabalho de revisão dedicando horas do seu tempo a verificar de que modo um assunto ficaria melhor explicado e a adequação do vocabulário. E o Mauro Koyama, codificando o meu projeto no ambiente que ele desenvolveu para a tese dele mesmo mas aprimorou e caprichou para a minha tese. Sem dizer do incentivo dos amigos Rosana, Fátima e Marcius Fabius do CenPRA e meus amigos, Cíntia e Miro, da Unicamp As orações da minha mãe e o carinho do meu filho Rodrigo. Sinto-me particularmente abençoada por ter o privilégio de ter vocês ao meu lado e do meu lado.

O problema não é inventar. É ser inventado hora após hora e nunca ficar pronta a nossa edição convincente.

Carlos Drummond de Andrade

Resumo

ARAÚJO, Olga Fernanda Nabuco, Ferramenta para Aplicação em rede de Compartilhamento de Habilidade no ambiente da Manufatura, Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, 2002, 130 p. Tese (Doutorado).

Considerando o ciclo de vida do produto no âmbito da Manufatura, a fase de concepção e especificação congrega conhecimentos de várias áreas. Estes conhecimentos estão dispersos por pessoas que na maioria das vezes não possuem um vocabulário em comum. Corre-se então o risco de perder oportunidades porque foi subestimado o potencial em torno ou mesmo porque não se consegue estabelecer correlação entre temas. O trabalho aqui apresentado tem como objetivo organizar o vocabulário informal de especialistas em domínios categorizados do conhecimento de forma a poder se estabelecer uma correlação entre eles e conscientizar o usuário da potencial rede de habilidades que ele está inserido . Ontologias são utilizadas para organizar e relacionar os domínios do conhecimento enquanto agentes representam o usuário e recuperam a informação que foi inferida como necessária para o usuário. O sistema possui um processo automático de atualização e prevê que o usuário seja poupado de tarefas que normalmente considera estranhas ao seu trabalho. Os resultados demonstram a potencialidade deste tipo de sistema de recomendação podendo ser estendido tanto a outros domínios quanto a outros usos.

Palavras-chave:

– rede de habilidades, ferramenta para compartilhamento de conhecimento, ontologia para manufatura, ferramenta orientada a agentes, sistema de recomendação para manufatura, modelo de referencia para compartilhamento de conhecimento, taxonomia para manufatura.

Abstract

ARAÚJO, Olga Fernanda Nabuco, Ferramenta para Aplicação em rede de Compartilhamento de Habilidade no ambiente da Manufatura, Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, 2002, 130 p. Tese (Doutorado).

The manufacturing product life cycle in its conception and designing phases congregates knowledge coming from diverse domains. This knowledge is spread throughout individuals that most of time are not aware of each other vocabulary. In this case opportunities can be lost either because the potential around them is underestimated or correlating themes is burdensome. The work presented here is a tool intended to organize the informal vocabulary used by communities of practice in formal domain categories in order to establish a correlation between them and advice the user of the potential skills net he/she belongs to. Ontologies are used to relate domains and user's profile while agents capture user's preferences and provide useful information. The tool updates general information unburdening the user from tasks she/he is unfamiliar to. Experiments demonstrate the potential of this kind of tool envisaging its broader use and application in other domains.

Keywords:

- skills net, knowledge sharing tools, manufacturing ontology, agent-oriented tools, manufacturing referral system, knowledge sharing framework, manufacturing taxonomy.

Índice Geral

Índice de Figuras **8**

Índice de Tabelas **10**

Nomenclatura **11**

Capítulo 1 ..... **13**

Introdução **13**

1.1 Contexto 13

1.2 Objetivos 14

1.3 Premissas 14

1.4 Motivação 15

1.5 Estrutura da Tese 16

Capítulo 2 ..... **18**

Estado da Arte: Ontologia, Agentes e Máquina de Aprendizado **18**

2.1 Introdução 18

2.2. Ontologia 20

2.2.1 Aplicações de Ontologias 20

2.2.2 Importância de Ontologias 21

2.2.3 Representação do Conhecimento empregando Ontologias 21

2.2.4 Linguagens para Representação de Ontologias 22

2.2.5 Linguagens de Armazenamento de Ontologias 25

2.2.6 Ferramentas para construir ontologias 30

2.2.7 Projetos de ontologia 36

2.3. Agentes 38

2.3.1 Características de agentes 39

2.3.2 Tipos de Agentes 40

2.4 Máquina de Aprendizado – redes *Bayesianas* 44

2.4.1 KEA – Algoritmo para extração de frases-chave 44
2.5 Conclusões 46

Capítulo 3 47

Requisitos e Arquitetura do Sistema Proposto 47
3.1. Introdução 47
3.2. Definição dos critérios para escolha de requisitos 48
3.3. Requisitos de Cooperação 51
3.3.1 Visão do Usuário 51
3.3.2 Visão do Sistema 55
3.4. Requisitos de Coordenação 58
3.4.1 Visão do Usuário 58
3.4.2 Visão do Sistema 59
3.5. Requisitos de Comunicação 61
3.5.1 Visão do Usuário 61
3.5.2 Visão do Sistema 64
3.6. Quadro de Referência dos requisitos para compartilhamento de conhecimento 64
3.7. Sistema de Recomendação: Especificação e Detalhamento da Arquitetura 66
3.7.1. Especificação da Funcionalidade 66
3.7.2. Arquitetura Funcional do Sistema 68
3.8. Detalhamento da Arquitetura do Sistema 70
3.8.1. Papel do agente *Sheik* 72
3.8.2. Papel do agente *Erudite* 74
3.8.3. Modelo de Entendimento 75
3.9. Conclusão 75

Capítulo 4 78

Descrição do projeto do Sistema Multi-agente 78
4.1 Introdução 78
4.2. Componentes do Sistema 79
4.2.1. Modelo do Agente Sheik 80
4.2.2. Modelo do agente *Erudite* 85
4.3. Módulo Prototipador de Agentes 87

4.3.1. Descrição da Metodologia 88
4.3.2. Vista de Implementação 91
4.4. Conclusão 93

Capítulo 5 95

Descrição da ontologia 95
5.1 Introdução 95
5.2 Critérios para criação de ontologia 96
5.3. O ambiente *Protégé* para construção de ontologias 97
5.3.1 Arquitetura do ambiente 97
5.3.2 Os componentes do sistema: vista do meta-modelo 97
5.3.3 Componente Gerenciador de Perfil 101
5.3.4. Componentes do Quadro de Referência 103
5.3. Conclusão 104

Capítulo 6 106

Avaliação do sistema e trabalhos relacionados 106
6.1 Introdução 106
6.2 Panorama dos sistemas de recomendação 107
6.3 Quadro comparativo entre os sistemas 113
6.4 O sistema implementado e seus resultados 114
6.4 Contribuições 121
6.5 Conclusão 121

Capítulo 7 123

Conclusão e Propostas Futuras 123
7.1 Metodologia para formação de rede de habilidades 124

Apêndice 1 130

Metodologia de Programação Orientada a Agentes 130

Referências Bibliográficas Capítulo 1 134

Referências Capítulo 2 136

Referências Bibliográficas Capítulo 3 141

Referências Bibliográficas Capítulo 4 145

Referências Bibliográficas Capítulo 5 146

Referências Bibliográficas Capítulo 6 147

Índice de Figuras

Figura 2.1 Espectro de Ontologia (Fonte: (McGuiness, 2002)) 22

Figura 2.2 Um exemplo de rede (Nardi, 2002) 24

Figura 2.2 Declaração em RDF 28

Figura 2.3 Modelo RDF 29

Figura 2.4 Arquitetura do ambiente de desenvolvimento Protégé 32

Figura 3.1 Fases sob análise da cooperação tecnológica do ciclo de vida do produto 51

Figura 3.2 Alvo da Cooperação Tecnológica (Cagliano, 2000) 52

Figura 3.3 Critérios para análise da forma de organização da cooperação (Cagliano, 2000) 53

Figura 3.4 Caracterização dos requisitos identificados versus sistema proposto 67

Figura 3.5 Arquitetura Funcional do Sistema de recomendação Proposto 69

Figura 3.6 Agente *Sheik* e suas funcionalidades 70

Figura 3.7 Agente *Erudite* e suas funcionalidades 70

Figura 3.8 Relacionamento entre a área do conhecimento e os agentes 71

Figura 4.1 Arquitetura do Sistema Multi-agente 79

Figura 4.2 Módulos do sistema e seus relacionamentos (Diagrma de *Pakages* UML) 80

Figura 4.3 Interação entre o usuário e o agente *Sheik* (Diagrama de Utilização UML) 81

Figura 4.4. Classes do agente Sheik (Diagrama Lógico UML) 82

Tabela 4.1 Tabela de documentação da classe *AgentSheik* 83

Figura 4.5. Máquina de Estado do agente *Sheik* (Diagrama de estados UML) 84

Figura 4.6 Comportamento do agente *Erudite* (Diagrama de Utilização UML) 86

Figura 4.7 As Classes do Agente *Erudite* (Diagrama Lógico UML) 87

Figura 4.8 Sistema SMA e a comunicação entre os agentes 89

Figura 4.9 Decomposição em SMA 89

Figura 4.10 Cenário C1 90

Figura 4.11 Dedução das máquinas de estado 91

Figura 4.12 Gerenciador de dados do usuário (*Profile Manager*) 92

Figura 4.13 Interface para configuração de dados estáticos (*Register*) 93

Figura 5.1 Ontologia do sistema de recomendação 98

Figura 5.2. A taxonomia de Manufatura definida como classe no *Protégé* 99

Figura 5.3 Classe PersonProfile implementada no Protégé 100

Figura 5.4 Gerenciador de Perfis 102

Figura 5.5. Classe Perfil (*Profile*) implementada no *Protégé* 103

Figura 5.6 Diagrama do Componente *Framework* 104

Figura 6.1 Acesso ao núcleo do Protégé via interface programática 116

Figura 6.2 Agente *Sheik* demanda informação do agente *Erudite* 117

Figura 6.2 O agente *Erudite* responde à demanda 117

Figura A.1 Relacionamento entre os modelos 131

Índice de Tabelas

Tabela 2.1: Glossário de Representação do Conhecimento (Farquar, 1997) 23

Tabela 2.2. Características de agentes (Fonte: Recticular Systems, 1999) 43

Tabela 3.1 Relação entre as dependências e os processos de gerenciamento 59

Tabela 3.4 Visão x Requisitos para Compartilhamento de Conhecimento 65

Tabela 3.3. Esquema para o papel do agente *Sheik* 73

Tabela 3.4. Esquema para o agente *Erudite* 74

Tabela 4.2 Máquina de Estado State/Activity 84

Tabela 6.1 Comparação entre os sistemas de recomendação 114

Nomenclatura

CSCW – *Computer Supported Cooperative Work*

DAML – *Darpa Agent Markup Language*

DFD – Diagrama de Fluxo de Dados

DL – *Description Logic*

DSE – *Distributed Systems Engineering*

DTD – *Document Type Definition*

EADS – *European Aeronautic Defence and Space*

FIPA – *Foundation for Intelligent Physical Agents*

FRS – *Frame Knowledge Representation System*

HTML – *Hyper Tex Markup Language*

IST – *Information Societies Technology*

KADS – *Knowledge Analysis and Design System*

KIF – *Knowledge Interchange Format*

KQML – *Knowledge Query*

OIL – *Ontology InferenceLayer*

OKBC – *Open Knowledge Base Connectivity*

OWL – *Web Ontology Language*

PIPEFA – Plataforma Industrial para Pesquisa, Ensino e Formação em Automação

RDF – *Rsource Description Framework*

SMA – Sistema Multi-Agente

UML – *Unified Modelling Language*

URI – *Uniform Resource Identification*

URL – *Uniform Resource Location*

UTF – *Unicode Transformation Format*

W3C – *World Wide Web Consortium*

X ML – e*Xtended Markup Language*

Capítulo 1

Introdução

Este capítulo contextualiza redes de compartilhamento de habilidades como tema desta tese, descreve os objetivos, premissas e a sua estrutura.

1.1 Contexto

A fase de concepção e especificação de um produto, no âmbito da Manufatura, congrega conhecimentos de várias áreas. Estes conhecimentos estão dispersos, em grande medida nas pessoas, as quais na maioria das vezes não possuem um vocabulário em comum. Corre-se então o risco de perder oportunidades por subestimar-se o potencial de cooperação em torno dos temas de projeto ou mesmo por não estabelecer correlação entre eles. Torna-se assim fortemente desejável o compartilhamento de conhecimentos e para auxiliá-lo, o apoio de um sistema computacional que viabilize a interação entre as pessoas (Nihtilä, 1999; Wasco, 2000). Esse compartilhamento tem como principal objetivo evitar a duplicação de esforços e disponibilizar as melhores práticas. Outro objetivo insere-se no processo de aprendizagem no qual pessoas aprendendo umas com as outras, estabelecem contatos que resultam numa rede de excelência (Smith, 2001).

Comunidades de prática, comunidades de conhecimento, comunidades técnicas, ecologias de conhecimento, redes profissionais, redes de conhecimento cooperativas (*Deloitte Research*, 2001) e redes de melhores práticas são diferentes nomes pelos quais as pessoas reconhecem seus parceiros no mundo real em uma rede de excelência (Smith, 2000). Redes de habilidades representam os indicadores dos elementos de uma potencial comunidade de prática apoiados pela tecnologia de informação.

O indicador é um sistema de recomendação. Este tipo de sistema baseia-se na opinião que um especialista emite sobre um assunto específico. A fim de preservar o especialista de uma

sobrecarga, em termos de demanda de opiniões, um sistema computacional considerado como um sistema de recomendação, colhe a opinião do especialista e, quando perguntado, o fornece a opinião do especialista.

1.2 Objetivos

Este trabalho tem como objetivos:

a) propor um quadro conceitual de referência que possibilite conceber e estruturar o sistema de recomendação. Este quadro foi idealizado a partir do resultado de estudos sobre cooperação; ele estratifica a funcionalidade referente à identificação, aquisição, consulta e localização de conhecimento;

b) propor e descrever um sistema computacional que recomende a um usuário um grupo de pessoas cujos conhecimentos e interesses sejam semelhantes;

c) permitir a identificação e classificação do vocabulário informal, utilizado pelos usuários do sistema, segundo um modelo que explicite formalmente os seus domínios do conhecimento;

d) construir dinamicamente a rede de habilidades segundo o vocabulário informal inferido.

1.3 Premissas

A concepção desta tese considera as seguintes premissas:

a) o conhecimento é um dos recursos mais valiosos numa organização (Wasko, 2000). Para que seja útil a toda comunidade, necessita ser classificado, armazenado e disponibilizado (Smith, 2000);

b) a forma de representação, armazenamento e troca para informações proposta pela *web* semântica (Bernes-Lee, 2001);

c) a forma que as pessoas utilizam para saberem sobre os conhecimentos práticos referentes a seus trabalhos por intermédio de outras pessoas, incluindo as participantes de comunidades de prática, redes de conhecimento cooperativas, de uma ou mais organizações;

d) a tecnologia de informação, considerada um facilitador que de forma prática possibilita capturar, compartilhar e disseminar informação e conhecimento nas organizações.

1.4 Motivação

O projeto de um novo produto envolve especialistas de diversas áreas do conhecimento. Estes especialistas podem estar dispersos em empresas de porte distintos, diferentes empresas, centros de pesquisa e universidades.

Alguns projetos tratam exatamente deste aspecto, da busca e localização destes especialistas e de uma possível classificação dos mesmos segundo estas áreas de trabalho. O projeto MakeIt (MakeIt, 2000), e o projeto Genial (Radeke, 1999), por exemplo, pesquisaram as características e possíveis soluções para este problema em pequenas e médias empresas. O método consistiu em levantar as competências principais das empresas envolvidas criando um modelo que outras empresas pudessem aderir. Como resultado foi construída uma base de conhecimento contendo as competências das empresas e gerenciado pelas próprias empresas. A atualização e inclusão de novas informações eram realizadas pelos próprios usuários do sistema.

Num outro nível, o projeto DSE (*Distributed Systems Engineering*, 2001), tinha como objetivo oferecer uma plataforma para trabalho cooperativo. As empresas consorciadas contratantes do projeto eram a EADS (*European Aeronautic Defence and Space Company*) e a Alenia Spazio (http://www.alespazio.it/) que tinham como tarefa a construção de uma nave de abastecimento da estação orbital internacional. Além dos especialistas das empresas consorciadas que estavam envolvidos no projeto da nave, estavam envolvidos especialistas que formavam um comitê de avaliação do projeto. Durante dois meses, antes de iniciar a fase de manufatura, o projeto parava e aguardava que estes especialistas fornecessem suas avaliações. O projeto DSE tinha como objetivo fornecer suporte tecnológico para esta atividade que envolvia o comitê de avaliação, os projetistas e a alta gerência das empresas envolvidas.

O comitê tinha como características a neutralidade, em termos das companhias interessadas financeiramente no projeto e também o conhecimento nas áreas que compreendiam a construção deste tipo de artefato. Eram escolhidos pela confiança que as companhias depositavam nos seus julgamentos. Este comitê era usualmente escolhido pelos gerentes da EADS e Alenia e novas indicações eram fornecidas pelos próprios constituintes do comitê.

Um dos problemas do consórcio consistia justamente na seleção dos especialistas. Outro residia no banco de dados das corporações, com diferentes formatos e com conteúdos que muitas vezes não correspondiam com a informação desejada

Outro projeto, desta vez um estudo da 3M (Figueroa, 2000) que constatou existirem ao mesmo tempo, na empresa, projetos semelhantes e em diversas fases de execução.

Um outro motivo que suscitou o interesse foi o fato que palavras indicando uma determinada especialidade podem mudar por razões do mercado, para diferenciar marcas de produto, podem ter diferentes significados dependendo da comunidade de onde surgiu.

Os problemas acima considerados foram: a procura por especialistas, dificuldade de manutenção de bancos de dados, dispersão da informação, pulverização de recursos humanos, entre outros. Uma solução foi mostrada entretanto, a indicação de pessoas por outras pessoas, correspondendo a uma ligação de confiança a partir do conhecimento existente entre elas.

Os projetos acima citados influenciaram esta tese na busca pela solução destes problemas tendo como foco plataformas computacionais que apoiassem a atividade das pessoas. Nem todos os problemas citados são solucionados aqui mas algumas contribuições foram dadas. A constituição de uma rede que relacionasse especialistas sob alguma área do conhecimento e direcionasse essa informação de maneira automática para pessoas com esta carência de informação e a atualização automática de novos termos criados por uma comunidade relacionados com uma área do conhecimento, constituem o foco deste trabalho

1.5 Estrutura da Tese

Esta tese está organizada em sete capítulos. Este é o primeiro e contextualizou e delimitou o trabalho.

O capítulo 2 apresenta o estado da arte e as teorias nas quais este trabalho está fundamentado: ontologias, compreendendo seus protocolos e ambientes de desenvolvimento; teoria de agentes e suas características; e máquina de aprendizado.

O capítulo 3 identifica os requisitos que um sistema de recomendação deve satisfazer de modo a suportar o compartilhamento de conhecimento, propõe e descreve a arquitetura deste sistema segundo a teoria de agentes.

O capítulo 4 detalha a arquitetura do sistema multi-agente, descrito em UML (*Unified Modeling Language*) sua lógica e funcionamento; apresenta a implementação utilizando o programa Prototipador de Agentes.

O capítulo 5 detalha a arquitetura da ontologia descrita em UML e sua implementação utilizando o ambiente de desenvolvimento *Protégé*.

O capítulo 6 apresenta uma comparação do sistema de recomendação com outros projetos semelhantes a este trabalho, em termos das técnicas utilizadas.

O capítulo 7 descreve as conclusões e perspectivas para continuidade do trabalho.

Capítulo 2

Estado da Arte: Ontologia, Agentes e Máquina de Aprendizado

Este capítulo descreve a fundamentação teórica e o suporte tecnológico investigados para consecução deste trabalho. A descrição está estruturada em três seções: ontologias, utilizadas para representar e compartilhar conhecimento; agentes, empregados como uma classe especial de *software* adequado para representar um usuário por intermédio de uma vista específica; e a técnica de máquina de aprendizado aplicado à extração de frases-chave, que oferece mecanismos para aquisição de conhecimento.

2.1 Introdução

O compartilhamento de conhecimento entre pessoas, pertencentes ou não a uma comunidade, pode ser realizado de várias maneiras; algumas delas são: trazer para seus participantes a literatura recomendada por especialistas: artigos, livros, endereços da Internet etc.; depoimentos sobre como solucionar um problema específico que técnicos especializados compartilham em pequenas comunidades; listas de discussão sobre assuntos específicos, nas quais pessoas de uma comunidade resolvem problemas uns dos outros, podendo inclusive ser auxiliadas por um *staff* regular. Mesmo tão disperso e diverso, o conhecimento pode ser organizado em domínios que são definidos por assunto, viabilizando assim sua representação, aquisição, armazenagem, recuperação, fusão e/ou busca utilizando-se recursos computacionais.

Pode-se organizar a representação do conhecimento empregando-se uma taxonomia que classifique-o por assuntos, ou desenvolvendo-se um meta-modelo que formalize-o em itens inter-relacionados ou mesmo ambos. Uma taxonomia é uma classificação de elementos em um domínio específico do conhecimento onde se estabelecem relacionamentos entre os elementos, que podem ser objetos ou conceitos. A orientação a objetos é uma das abordagens que auxiliam a criação de um meta-modelo. Essa abordagem oferece conceitos e recursos analíticos que

permitem expressar o conhecimento em classes, interrelacioná-las e organizá-las hierarquicamente, facilitando assim sua compreensão. Os relacionamentos entre as classes é realizado por intermédio de mecanismos de abstração os quais conferem significado ao conhecimento. Juntos, classes e relacionamento, tornam um meta-modelo um elemento comum de entendimento para ambos: indivíduo e máquina.

A utilização conjunta de uma taxonomia e da hierarquia e mecanismo de herança da orientação a objetos tornam possível a organização de um vocabulário representativo de uma área de conhecimento. Este "vocabulário" especializado define conceitos muito mais do que palavras e é chamado de ontologia, proveniente da Inteligência Artificial. As ontologias fornecem os meios para o compartilhamento de conhecimento porque os conceitos usados na construção de uma ontologia podem ser compartilhados pelos que possuem necessidades similares. Mais importante ainda, segundo Chandrasekaran (1999), é o fato que a ontologia constitui o cerne de qualquer sistema de representação do conhecimento e a análise efetuada no processo de construção de uma ontologia torna clara a estrutura do conhecimento.

Outro elemento pertencente à área de Inteligência Artificial considerado neste capítulo é o sistema de multi-agentes. Este componente possui qualidades que o torna adequado para aplicações onde a distribuição de tarefas entre os diversos componentes auxilia a alcançar um objetivo. O sistema multi-agente possui características como cooperação, autonomia e pode acrescentar capacidades à sua operação, como aprendizado de máquina. O sistema multi-agente realiza a coordenação do sistema proposto nesta tese e para tal utiliza ontologias e máquina de aprendizado como recurso.

As máquinas de aprendizado são aqui representadas por uma rede Bayesiana, que é um programa utilizado para reconhecimento de padrões. O reconhecimento de padrões é utilizado no processo de aquisição de conhecimento onde um agente, lançando mão deste recurso, vai compondo os interesses de um usuário. Este trabalho foi concebido de maneira que o mecanismo de aquisição minimize automaticamente a participação do usuário no fornecimento da informação referente a seu conhecimento. As seções seguintes apresentam os principais conceitos que fundamentam esta tese.

2.2. Ontologia

Ontologia ou ontologia? As duas definições seguintes sintetizam o que se depreende deste termo e ampliam a compreensão sobre o assunto; elas foram propostas por Guarino (1995):

a) Ontologia (letra maiúscula): é o ramo da Filosofia que trata a natureza e a organização da realidade;

b) Ontologia (letra minúscula): (sentido 1): uma teoria lógica que oferece uma representação explícita, parcial de uma conceituação; (sentido 2) : sinônimo de conceituar.

A ontologia descreve formalmente um conhecimento por intermédio de conceitos e inter-relações que existem em um domínio específico, tal como uma unidade de negócio, área de estudo ou pesquisa. A ontologia especifica o ponto de vista comum selecionado, delimita a faixa do fenômeno sob estudo e define a terminologia usada para adquirir conhecimento do domínio em questão. A ontologia caracterizará seu domínio representando crenças, objetivos, hipóteses e previsões em acréscimo a fatos simples.

A importância de uma ontologia advém da possibilidade de se compartilhar a compreensão sobre um determinado domínio do conhecimento. Ao possibilitar o compartilhamento por exemplo, empreendimentos podem ter a mesma base de informação aumentando a integração entre as diferentes áreas e empresas que formam um negócio (Ushold, 1997; Staab , 2001; Izumi, 2001; Flett, 2001). Como também aplicação em *web* semântica (*semantic web*), que é uma "extensão da *web* na qual informação possui um significado bem definido aumentando a possibilidade de computadores e pessoas trabalharem cooperativamente" (Bernes-Lee, 2001).

2.2.1 Aplicações de Ontologias

Ontologias são úteis à compreensão da linguagem natural de dois modos, segundo Chandrasekaran (1999): no primeiro, o domínio do conhecimento às vezes possui um papel crucial auxiliando a desfazer ambigüidades**,** identificar as categorias semânticas que estão envolvidas na compreensão do discurso naquele domínio. Para este uso as ontologias possuem um papel de um dicionário de conceito. A ontologia esclarece a estrutura do conhecimento e, uma vez construída, pode ser reutilizada eliminando a necessidade de repetir o processo de análise de conhecimento.

Considerações adicionais referentes a aplicações de ontologias mostram que ontologias simples podem ser aplicadas para organizar informação possível de ser classificada sob definições, facilitando navegação e fornecendo uma taxonomia para várias aplicações. Exemplos podem ser vistos como www.dmoz.com, www.google.com. Ontologias estruturadas provêem informação detalhada sobre um assunto específico, oferecendo suporte a buscas e consultas a bases de dados projetadas a partir delas. Essas ontologias também fornecem meios à interoperabilidade e recuperação de informação com restrições em buscas e consultas.

2.2.2 Importância de Ontologias

A importância de uma ontologia tem sido reconhecida pela tecnologia da informação, a qual abrange: engenharia do conhecimento, sistemas de bases de dados, representação do conhecimento, modelagem qualitativa, integração da informação, gerenciamento de conhecimento, projeto de sistemas baseados em agentes, em todas as categorias que possa assumir. Também mencionado por McGuiness (McGuiness, 2002) "ela pode ser estendida com a ênfase na *web* e também incluir áreas de definição de meta-dados de uso geral (Dublin Core 1999), busca aprimorada de ontologia (por exemplo, eCyc (http://www.e-Cyc.com/) e FindUR (McGuinness 1998)). Possivelmente a maior delas, comércio eletrônico (por exemplo, Amazon.com, *Yahoo Shopping*, etc".

A figura 2.1 apresenta o espectro de ontologia como visto por McGuiness (2002). A linha horizontal mostra o verdadeiro potencial dos significados da ontologia, desde catálogos até *frames* com restrições de valor e relacionamentos lógicos. À medida que a necessidade de expressar informação aumenta, ampliam-se os atributos que descrevem propriedades, a fim de melhor representar os conceitos associados ao mundo real ou a pessoa e computador.

2.2.3 Representação do Conhecimento empregando Ontologias

A pesquisa e desenvolvimento de conceitos, formalismos etc. para representação de conhecimento predominam nas áreas de engenharia do conhecimento, processamento de linguagem natural, sistemas cooperativos de informação, integração de sistemas inteligentes e gestão do conhecimento. Atualmente, as ontologias são um tópico popular de pesquisa dessas áreas por fornecerem um entendimento comum e compartilhado de um domínio que pode ser comunicado entre pessoas e aplicativos. As ontologias possuem uma função similar a de um

esquema de base de dados contudo, diferenciam-se dele nos seguintes aspectos, de acordo com Fensel (2000),:

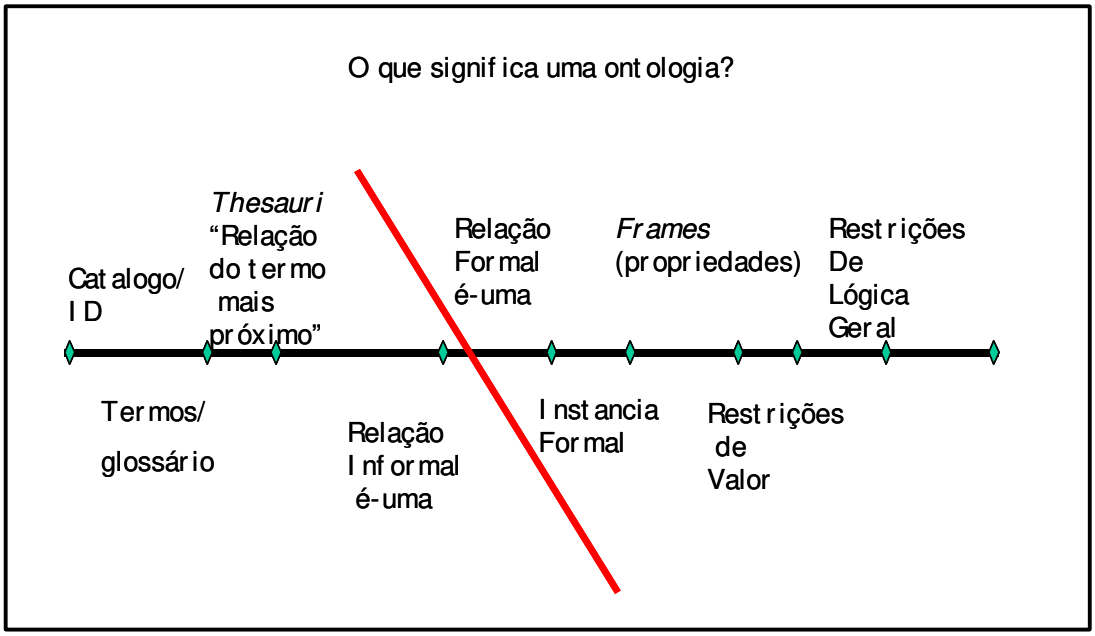


Figura 2.1 Espectro de Ontologia (Fonte: (McGuiness, 2002))

– Uma linguagem para definir ontologias é sintática e semanticamente mais rica do que as abordagens comuns para descrição de bases de dados;

– A informação descrita por uma ontologia consiste em textos de linguagem natural semi-estruturada e não na forma de tabelas;

– Uma ontologia necessita ser uma terminologia consensual e compartilhada porque é usada para compartilhamento e troca de informação;

– Uma ontologia oferece uma teoria de domínio e não a estrutura de um repositório de dados.

2.2.4 Linguagens para Representação de Ontologias

A fim de representar o conhecimento, utilizam-se ambientes computacionais que permitem desenvolver ontologias. Os recursos de edição desses ambientes incluem linguagens que possibilitam expressar significados, heranças e relacionamentos do domínio de conhecimento sob consideração.

Com o objetivo de clarificar a terminologia de editores de ontologia e sistemas baseados no conhecimento, serão apresentados os conceitos usados para representar a maioria dos construtores de ontologia, Sistema de Representação de *Frame* (FRS), de acordo com Peter Karp (1993). FRS são conhecidos por uma variedade de nomes, incluindo rede semântica, sistemas de *frame*, lógicas de descrição, rede estrutural de herança, grafos conceituais, e inferidor terminológico. Eles são a base da representação do conhecimento onde o *frame* é uma estrutura de dados usada tipicamente para representar um objeto simples, uma classe de objetos relacionados ou um conceito geral (predicado).

a) *Frames* - São tipicamente arranjados numa hierarquia taxonômica na qual cada *frame* é ligado a um *frame*-pai. Uma coleção de *frames* numa ou mais hierarquias de herança formam uma base de conhecimento. *Frames* possuem componentes chamados *slots* que descrevem propriedades ou atributos do que está sendo descrito pelo *frame*. Definições de *slots* geralmente possuem outros componentes além dos *slots* nome, valor e restrição de valor, tal como o nome de procedimento que pode ser usado para computar um valor do *slot*, e uma justificativa (no sentido verdadeiro de manutenção) de como um valor de um slot foi computado. Estes diferentes componentes de um *slot* são chamados de suas facetas (*facets*). Farquar (1997), propôs um glossário que representa os conceitos usados na ontologia de *Frame*, tendo ampliado seu uso incluindo editores de ontologia.

Tabela 2.1: Glossário de Representação do Conhecimento (Farquar, 1997)

Frame	Qualquer objeto, incluindo classes, instâncias e relações
Relação	Uma relação entre um ou mais objetos
Classe	Um conjunto distinto de objetos
Instância	Um membro de uma classe
Função	Uma relação onde o último argumento é determinado unicamente pelos outros
Slot	Uma relação binária
Slot Próprio	Um slot pertencente a um frame
Template Slot	Um slot associado com uma classe, mas com valores para instâncias
Faceta	Uma relação ternária num frame, slot, valor
Restrição	Qualquer declaração que restrinja a interpretação possível de um frame
Axioma	Qualquer declaração considerada verdadeira

b) Redes Semânticas – São estruturas cognitivas formatadas como rede mapeando o conhecimento e a seqüência lógica do sistema sob consideração. Sistemas de frame e redes

semânticas podem ser vistos como estruturas baseadas em rede nas quais são representados seus indivíduos e relacionamentos.

c) Sistemas de Representação de Frame – Os que empregam classificação, às vezes são chamados deliberadores terminológicos por manterem relacionamentos entre um conjunto de termos – definições de classe ou conceito. O nome evoluiu para Descrição Lógica, significando a descrição das propriedades que suportam sistemas lógicos.

d) Descrição Lógica – É um formalismo destinado a representar conhecimento. Ela permite acrescentar o papel valor/restrições a relacionamentos is-a. O exemplo apresentado por Nardi (2000) ilustra este conceito. Na figura 2.2 podem ser vistos nós e arcos nos quais o relacionamento entre mãe e pai pode ser lido como Mãe is-a Pais. O relacionamento *is-a* indica um relacionamento de herança, onde o conceito mais especializado herda as propriedades do mais geral. Uma característica da Lógica de Descrição é sua capacidade de representar relacionamentos além da herança de propriedades. Entre Pai e Pessoa pode ser introduzida uma restrição que delimita a faixa de tipo de objetos que pode satisfazer a propriedade de relacionamento, chamada "papel". O papel possui também uma faixa de validade, expressa na forma (1, NIL), a qual delimita valores superiores e inferiores. O diagrama pode ser lido como "Pais são pessoas que possuem no mínimo um filho(a) e todo(a) filho(a) seu é uma pessoa".

Os aspectos das características-chave da Descrição Lógica residem nos construtores para estabelecer relacionamentos entre conceitos.

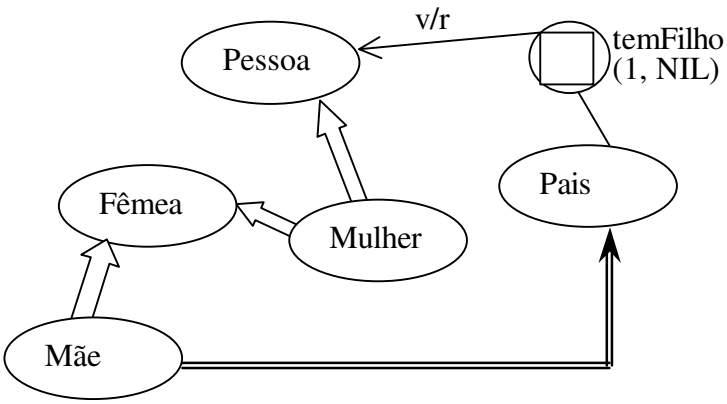


Figura 2.2 Um exemplo de rede (Nardi, 2002)

2.2.5 Linguagens de Armazenamento de Ontologias

Se para representar o conhecimento a comunidade usa um sistema baseado em *frame*, para armazená-lo, a maioria dos editores de ontologia e os baseados em conhecimento usam as seguintes linguagens: *Resource Description Framework* (RDF), *eXtended Markup L*anguage (XML), *Hypertext Markup Language* (HTML) e *Darpa Agent Markup Language* + *Ontology Inference Layer* (DAML+OIL).

Os quatro itens seguintes fornecerão uma descrição geral das linguagens e suas importâncias no compartilhamento do conhecimento. Ambas, HTML e XML, são subconjuntos do SGML, *Standard Generalized Markup Language*, (ISO 8879:1985), o padrão internacional para definir descrições de estrutura dos diferentes tipos de documentos eletrônicos. HTML define os tipos possíveis de serem usados num documento Web. XML permite a criação de tipos de documentos, graças a seu tipo específico e flexível de representação de documentos, como os que relativos à química e música. XML DTD (*Document Type Definition*)possui o formato para texto, possibilitando sua interpretação e acesso por diferentes navegadores. O *Schema* XML permite a representação de tipos diferentes de tipos, números, símbolos especiais (química, matemática, música e outros), a fim de ser interpretado por navegadores e outros aplicativos. *Darpa Agent Markup Language* - DAML é uma extensão de XML e RDF. Embora o W3C participe no comitê do DAML ela não é a mesma linguagem como definida pelo RDF, mas possui o mesmo propósito.

HTML – ***Hypertext Markup Language***

HTML é uma linguagem de marcação que se tornou o padrão mais utilizado para escrever páginas na Internet. Suas principais características são: ser gratuita, aberta, simples, possibilita a qualquer pessoa escrever uma página, é processável / compreensível por todos os navegadores (*browsers*); sua simplicidade promove o aumento do uso da Internet. Infelizmente, essa mesma simplicidade não pode atender a crescente demanda por novos serviços dessa rede.

Uma linguagem de marcação serve para marcar ou identificar palavras, utilizando para isso sinais como ‘<’ e ‘>’, num documento que indica sua estrutura lógica (como parágrafos) e fornece instruções para sua disposição na página para transmissão eletrônica ou apresentação. Gráficos vetoriais, transações comerciais, equações matemáticas, meta-dados de objeto, servidor de APIs, e outros tipos de informação estruturada são considerados documentos estruturados.

XML – *eXtended Markup Language*

Foi concebida para satisfazer os requisitos de projetistas de páginas da *web* destinados a marcação de uso específico na indústria, de troca de dados de formato neutro quanto ao fornecedor, editoração independente do meio, marketing direto, gerenciamento de *workflow* em ambientes autorais colaborativos, e o processamento de documentos *web* por navegadores inteligentes. Atualmente, essa linguagem é internacionalizada em termos de idiomas europeus e asiáticos, e compreensível por todos os processadores que utilizam o conjunto de caracteres Unicode, em ambas as codificações: UTF-8 (*Unicode Transformation Format* 8-bit) e UTF-16.

Seu projeto visa permitir o processamento rápido na porção cliente, sendo assim consistente com seu propósito básico de ser um formato para publicação eletrônica e troca de dados (971208 W3C *press release*).

Ela fornece uma maneira mais geral do que HTML para apontar documentos, da maneira semelhante a HTML, porém estendendo a capacidade de ligação (link) deste de ponto a ponto para endereçamento estruturado em árvore. O localizador pode ser um endereço da Internet – *Uniform Resource Locator* – URL, uma consulta (*query*) ou um apontador estendido (*Xpointers*, possibilita o endereçamento de partes individuais de um documento XML).

Esquemas (*Schema*) XML definem:

a) Mecanismos para restringir estrutura e conteúdo de documentos;

b) Mecanismos para habilitar herança de elemento, atributo e definições de tipos de dados;

c) Mecanismo para restrições e descrições específicas da aplicação;

d) Mecanismo para habilitar integração de esquemas estruturais com tipos primitivos de dados;

e) Tipificação de dados primitivos, incluindo Byte, Data, Integer, Sequence etc.;

f) Mecanismos para a criação de tipos de dados definidos pelo usuário.

Outro elemento importante da definição XML é o DTD, Definição do Tipo de Documento, do inglês, Document Type Definition, que utiliza a Sintaxe de Declaração XML para descrever formalmente um tipo específico de documento. Um DTD possui três elementos: declaração do elemento que define tags compostos e faixas de valores para tags elementares, declaração de atributo que define atributos de tags e finalmente, declaração de entidade.

DAML+OIL - *Ontology Inference Layer*

Essa linguagem é uma combinação da DAML (*Darpa Agent Markup Language*, esforço em tecnologias para *web* semântica, patrocinado pelo governo norte-americano) e OIL (*Ontology Inference Layer*, Projeto Europeu IST1999-10132 On-To-Knowledge) destinado a ser um esquema mais expressivo do que RDF (Ouellet, 2002). Ela foi adotada numa iniciativa conjunta entre Estados Unidos e Europa culminando em uma linguagem chamada DAML+OIL.

DAML+OIL é um conjunto de declarações RDF e XML. RDF foi concebida para possibilitar pessoas construírem suas próprias definições de meta-dados. Contudo, a menos que o produtor e o consumidor de informação tenham o mesmo entendimento comum, a informação não pode ser compartilhada. Segundo Zaychik (2001), RDF não é suficiente porque permite somente uma construção limitada de restrições só aplicável em termos de alcance e/ou propriedades do domínio, não possui representação de propriedades de propriedades (veja abaixo definições de propriedades em RDF), equivalência ou disjunção e não possui semântica definida. A linguagem DAML+OIL estende o RDF facilitando a construção de inferências.

A idéia-chave da linguagem consiste em dispor de dados na Web que sejam definidos e relacionados, de tal maneira que seus significados possam ser explicitamente interpretados por processos de software, ao invés de serem implicitamente interpretados por pessoas. Mais especificamente, os projetistas de DAML e OIL alegam que usuários necessitavam de uma linguagem acessível por uma dedução lógica automática (*automatic reasoning*).

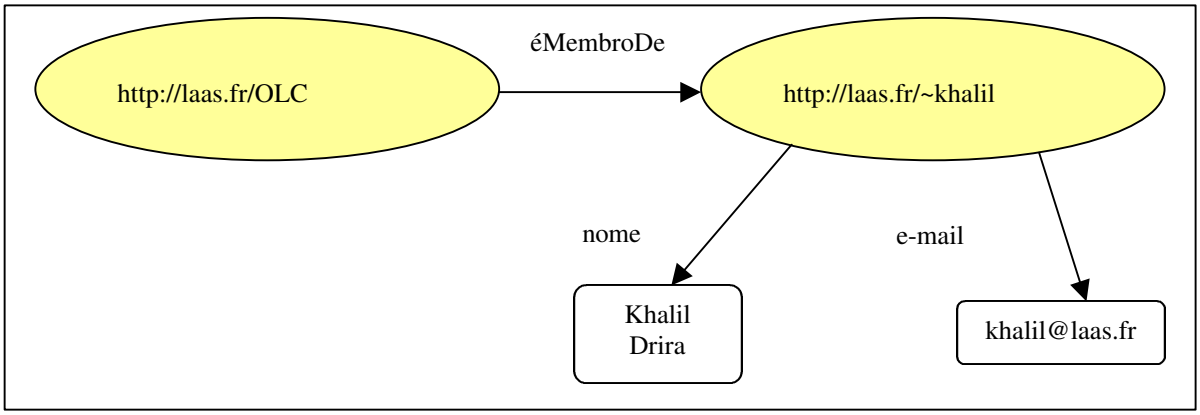
OIL é uma linguagem baseada em *frames* que amplia o RDFS (Esquema RDF) com um conjunto de primitivas que aumenta as possibilidades de descrições. DAML é uma extensão de RDF e parte das suas idéias são provenientes do OIL. Entrementes em novembro de 2001, o W3C deu início ao *Web Ontology Working Group* para definir uma linguagem para a *web* semântica. Este grupo está incumbido de utilizar DAML+OIL como seu ponto de partida, sendo mesmo uma revisão da linguagem, de modo que com esse aprendizado possa desenvolver uma outra linguagem chamada OWL (*Web Ontology Language*) (http://www.w3.org/TR/2002/WD-owl-features-20020729\). O grupo de trabalho do W3C faz circular *drafts* na comunidade com o intuito de revisar e divulgar seu trabalho.

RDF – *Resource Description Framework*

É uma linguagem de propósito geral destinada a representar informação tendo como foco a Web; ela propõe-se a suportar a interoperabilidade entre aplicativos que trocam informação na Web. O modelo RDF consiste de um Recurso, usando uma URL - URL, Propriedades, recursos usados como predicados de tuplas, eles descrevem recursos, e Declarações, sujeito, predicado e objeto, o conjunto tuplas, eles descrevem o conjunto de tuplas. A figura 2.2 mostra uma declaração RDF e a figura 2.3 apresenta um modelo RDF, ambos usando representação de grafos (RDF, 1999). Considerando a seguinte sentença:

Sujeito (Recurso)	http://www.laas.fr
Predicado (Propriedade)	IsMemberOf
Objeto (Literal)	“Khalil Drira”

A figura 2.2 representa uma declaração usando grafos na qual o recurso é um nó, o predicado é um arco, o objeto é o literal e a declaração “Khalil Drira” é membro de



http://laas.fr/OLC .

Figura 2.2 Declaração em RDF

A figura 2.3 pode ser lida como: a pessoa apontada pelo indicador ~khalil é nomeado Khalil Drira e possui o e-mail khalil@laas.fr. O recurso http://laas.fr foi descrito por esta pessoa.

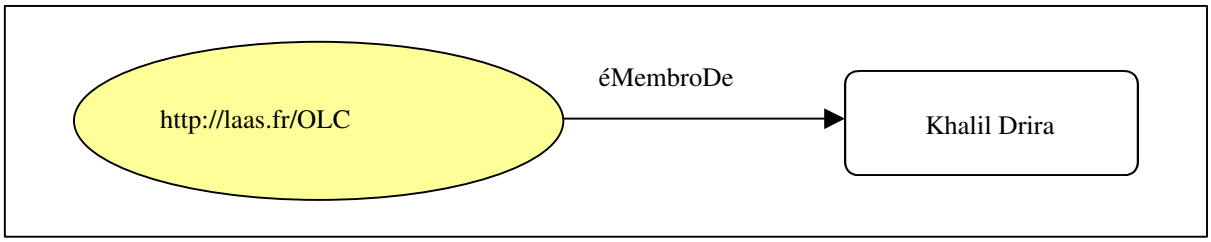


Figura 2.3 Modelo RDF

Como muitos projetos orientados a objeto e sistemas de modelagem, RDF tem um sistema de classe. As classes criadas com propósito específico ou domínio são chamadas Esquema (Schema). O Esquema descreve atributos e seus significados. Um Esquema pode definir as propriedades de um recurso, como também os tipos de recursos descritos (livros, pessoas, companhias etc.). A linguagem de especificação de esquema é influenciada pela linguagem de representação de *frame* (redes semânticas, lógica de predicado) como também linguagens de especificação de predicado e modelos de grafo de dados. Essa linguagem pretende ser menos expressiva porém mais simples que KIF.

KIF – *Knowledge Interchange Format*

É uma linguagem projetada para ser usada para a troca de conhecimento entre sistemas computacionais díspares (criados por diferentes programadores, em momentos distintos, em linguagens diferentes etc.) (Genesereth, 1992). Foi concebida para ser um formato de troca para ontologias e constitui uma extensão da linguagem de predicado de primeira ordem. Fornece meios para representar conhecimento sobre conhecimento. Isto permite o usuário explicitar decisões referentes a representação do conhecimento e introduzir nova construção de representação do conhecimento, sem modificar a linguagem.

O formato RDF possui muito em comum com essa linguagem: ambas possuem elementos de segunda ordem (fórmulas como termos em fórmula de meta-nível) e escopo global de propriedades.

Essa linguagem possui três tipos diferentes de expressões: termos, sentenças e definições. Termos são usados para denotar objetos no mundo sendo descrito; sentenças são empregadas para fatos sobre o mundo; e definições são utilizadas para definir constantes. Uma base de conhecimento é um conjunto finito de sentenças de definições. Seus quatro aspectos fundamentais são:

– semântica declarativa: possibilita entender o significado das expressões na própria linguagem;

– é logicamente compreensiva: sentenças arbitrárias são expressas em cálculo de predicado, diferindo assim das linguagens de bases de dados relacionais;

– provê recursos para a representação do conhecimento sobre a representação do conhecimento;

– em KIF estruturado, a noção de palavra é considerada como primitiva: significa que uma expressão tanto pode ser uma palavra quanto uma seqüência finita de expressões. Neste trabalho, usam-se parênteses para limitar os itens numa expressão composta, como descritos a seguir.

<palavra> ::= um objeto sintático primitivo

<expressão> ::= <palavra> | (<expressão>*)

O conjunto de todas as palavras da linguagem é dividido segundo uma categorização exaustiva. Suas sentenças são formadas empregando os operadores lógicos.

KIF é usada no projeto Ontolingua e também na PAL (*Protégé Axiomatic Language*) a Linguagem Axiomática utilizada no ambiente Protégé (ver seção 2.2.6).

2.2.6 Ferramentas para construir ontologias

Essas ferramentas são desenvolvidas considerando os conceitos de compartilhamento de conhecimento, aquisição de conhecimento, facilidades de interface homem-máquina, capacidades para consultar e incluir métodos de solução de problemas, linguagem e método para armazenar conhecimento etc. Esta subseção apresenta os conceitos de cada ferramenta citada nesta dissertação, facilitando assim a escolha da ferramenta usada neste trabalho.

a) Protégé 2000

– Essa ferramenta é um aplicativo desenvolvido pela área de informática de medicina da Universidade de Stanford e vem sendo aprimorado pela equipe que o desenvolveu e por centenas de usuários. Sua concepção, segundo uma estrutura funcional modular, torna-o um software escalável e extensível. Esse software é distribuído gratuitamente e possui mais de 5000 usuários registrados. Sua funcionalidade inclui um editor de ontologia e outro de base de conhecimento, sendo ambos baseados no modelo OKBC (Open Knowledge Base Connectivity, Chaudri, 1998). Oferece suporte a classe e hierarquia de classe com herança múltipla; gabarito e slots próprios;

especificação de facetas predefinidas e arbitrárias para slots, que incluem valores permitidos, restrições de cardinalidade, valores default e slots inversos; meta-classes e hierarquia de meta-classe.

– O Protégé é um ambiente para desenvolvimento de sistemas baseados em conhecimento e continuou evoluindo ao longo da última década (Gennari, 2002). O fato mais importante a respeito do Protégé é sua comunidade externa que desenvolve uma série de extensões ao próprio programa, depura o código e o utiliza para os mais diversos fins. Sem essa comunidade provavelmente ele seria mais uma entre tantas ferramentas do mesmo tipo já desenvolvidas mesmo mais poderosas que ele como o CommonKADS (Knowledge Analysis and Design System, Wielinga, 1992; Schreiber et al., 1994) e mais recentes (e com mais recursos) como o Oil (Fensel, 2000) e o DAML (Hendler). O Protégé projetou sua própria ontologia usando o formalismo de uma linguagem padrão baseada em frames, utilizando conceitos como classes, instancias, slots e facets. Muitas de suas idéias foram baseadas no sistema OKBC (Noy, 2000).

– O Protégé e o sistema KADS são ambos metodologias para projeto de sistemas baseados em conhecimento porém o KADS tem um ciclo mais completo ao definir quatro níveis de conhecimento: a camada domínio, para conhecimentos estáticos sobre o domínio; a camada de inferência, para conhecimento programático sobre a aplicação; a camada tarefa, para sequenciar um conjunto de inferências fornecendo uma solução de resolvedores de problemas completa; a camada estratégica para selecionar entre tarefas alternativas. No caso do Protégé a camada domínio foi realizada como ontologia domínio, enquanto as outras três camadas dependem da escolha e do desenvolvimento do método de solução de problema.

– A abordagem do Protégé inclui três classes de ontologias – domínio, método e aplicação. Ontologias tipo domínio definem o conceito relacionado com um domínio de aplicação. Tipo método especifica os requisitos dos métodos para solução de problemas (problem-solver methods) em termos de dados (a estrutura de entrada e saída de cada método). Tipo aplicação define conceitos que são específicos de uma aplicação ou implementação particular. Ontologias do tipo domínio e método podem ser reutilizadas em outras aplicações, são ontologias de alto nível, descrições gerais, enquanto ontologias de aplicação são específicas e sem intenção de reutilização. Estas diferentes ontologias podem ser integradas via mapeamentos que conectam determinados elementos através das ontologias.

– O Protégé também tem a preocupação de reutilização da ontologia. A idéia constitui na inclusão de ontologias. Os conceitos na ontologia incluída não podem ser modificados porém podem ser referenciados por outras classes e instâncias na base de conhecimento.

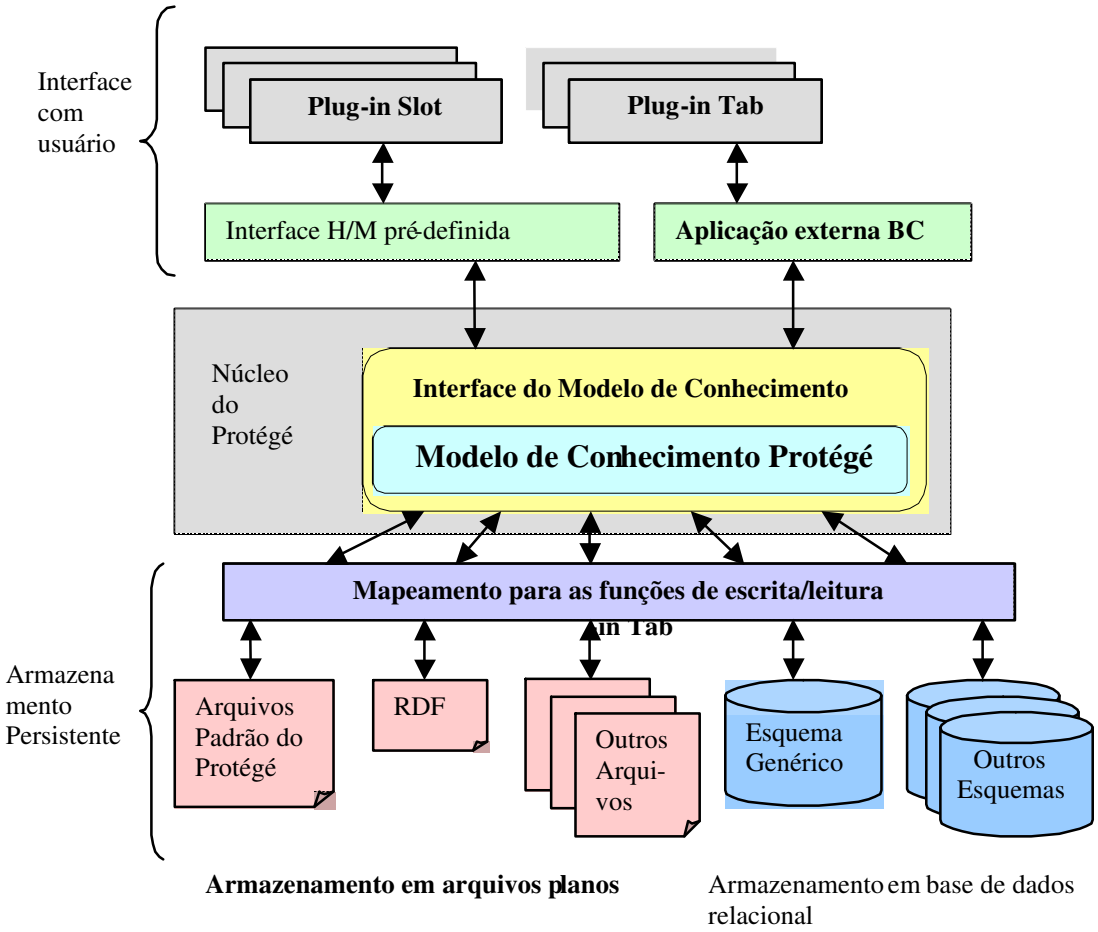


Figura 2.4 Arquitetura do ambiente de desenvolvimento Protégé

– A comunidade de desenvolvedores vem acrescentando funcionalidades a essa ferramenta e tornado-as disponíveis na forma de plugins. A maioria desses, pertence a uma das três seguintes categorias: (1) backends, permitem usuários armazenar e importar bases de conhecimento em vários formatos; (2) slot widgets, são usados para mostrar e editar valores de slot ou suas combinações em modos de domínio específico e tarefa específica; (3) tab plugins, são aplicativos baseados no conhecimento usualmente fortemente ligado a bases de conhecimento do Protégé.

– Esses plugins executam funções de máquina de inferência, gerenciamento de ontologias múltiplas, importação e integração on-line de grandes fontes de conhecimento, permitem expressar restrições em dados nas quais o formalismo de frame não é suficientemente expressivo,

vistas gráficas, imagens GIF (Graphical Interchange Format) de display, áudio e vídeo e suporte a armazenagem e importação de Esquema RDF, XML com DTD, XML com Esquema e OIL.

– A Figura 2.4 mostra a arquitetura do ambiente Protégé. Qualquer interação com os objetos que residem na base de conhecimento é feita através da interface do programa de aplicação (Application Programmers Interface – API). A flexibilidade do ambiente permite que os usuários/desenvolvedores criem plug-ins e os tornem disponíveis para a comunidade. Este é um exemplo de como compartilhamento de conhecimento pode ter efeito. Os desenvolvedores correspondem participando ativamente da lista de discussão, colocando disponíveis seus trabalhos e prestando os devidos esclarecimentos quando outros têm dúvidas sobre o mesmo. A maior parte do suporte é feita pela própria equipe desenvolvedora do Protégé.

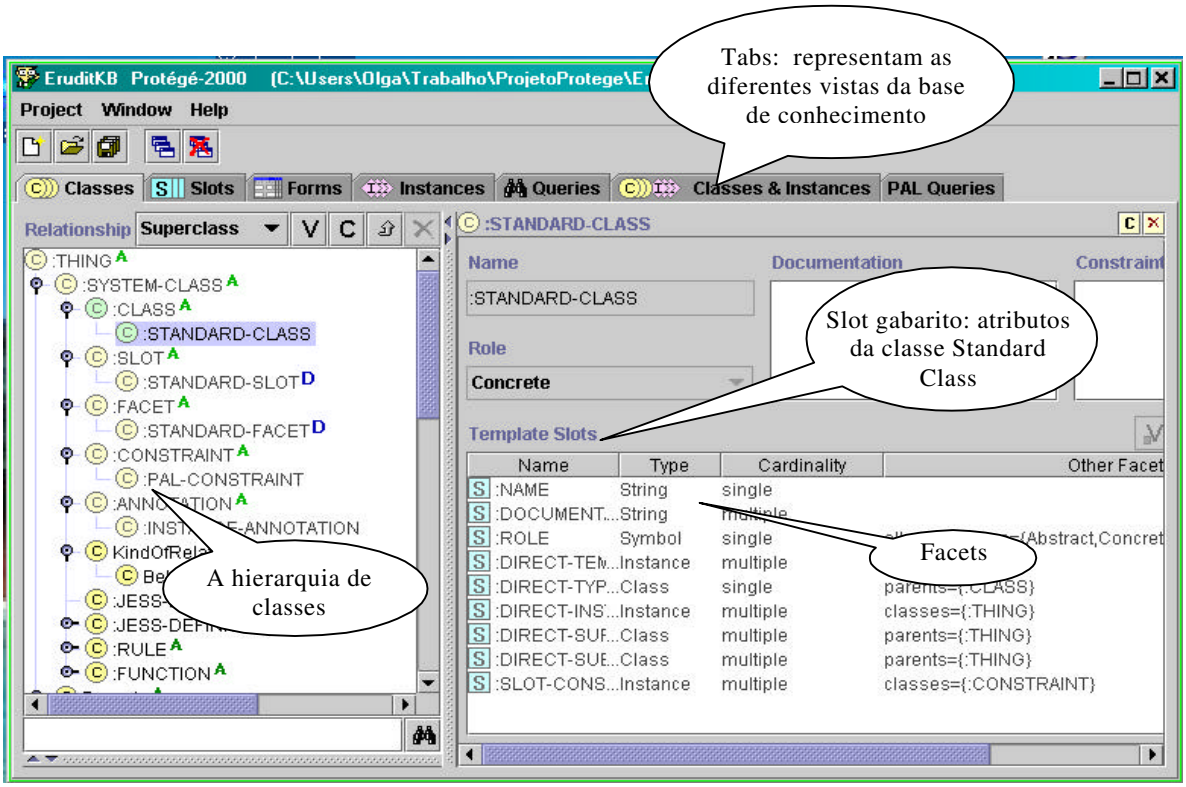


Figura 2.5 Interface com o usuário do ambiente Protégé

– A ontologia no ambiente Protégé consiste de classes, slots, facets e axiomas. Classes são conceitos no domínio de discurso (Noy, 2000). Slots descrevem as propriedades ou atributos das classes. Facets descrevem as propriedades dos slots. Axiomas especificam as restrições adicionais. A base de conhecimento do Protégé inclui a ontologia e as instâncias individuais das

classes com valores específicos para slots. As classes constituem uma hierarquia taxonômica e meta-classe é uma classe cujas instâncias são classes. A Figura 2.5 ilustra na interface de usuário os slots, facets, classes e tabs.

– Slot – descreve as propriedades associadas às classes e instâncias. Um slot é, ele mesmo, um frame. Os slots são definidos independentemente de qualquer classe. Quando um slot é associado a um frame em determinada ontologia ele pode possuir um valor.

– Facetas (facets) – Uma maneira de especificar restrições aos valores de slots é através de facetas. Estas restrições sobre o valor de um slot (por exemplo, inteiro, string, instância de uma classe) provocam um valor máximo ou mínimo sobre um valor numérico. Facetas definem restrições a um slot associado a uma classe.

– Slot – tipo gabarito e próprio – Um slot pode ser associado a um frame de duas maneiras: como gabarito ou próprio. Slot próprio associado a um frame descreve propriedades sobre o objeto representado por este frame (individual ou uma classe). Um slot próprio não é herdado por suas subclasses ou propagado por suas instâncias. Um slot gabarito pode ser associado somente a frames. Um slot deste tipo associado a um classe é herdado por suas subclasses. Um slot gabarito em uma classe se torna um slot próprio nas instâncias dessa classe. Os slots próprios descrevem uma propriedade de um (classe ou individual) frame ao invés de propriedades deste frame. Gabaritos descrevem propriedades de uma instância de uma classe.

b) Ontolingua

– É um ambiente que permite criar, navegar, avaliar e acessar ontologias, usando e mantendo ontologias reutilizáveis. Foi desenvolvido pelo KSL, Knowledge Systems Laboratory, Universidade de Stanford. Seu objetivo é produzir um mecanismo prático e útil, e assim possibilitar reduzir o esforço para produzir uma nova ontologia (Farqhuar, 1995). Sua concepção adotou a seguinte estratégia: fornecer um ambiente de edição rico, que ofereça capacidades de análise automática; disponibilizar este ambiente de edição a uma comunidade ampla; oferecer suporte explícito à construção colaborativa de ontologia; finalmente, permitir que desenvolvedores de ontologia reutilizem ontologias existentes.

– Esse ambiente amplia a linguagem KIF oferecendo um conjunto de axiomas com significância ontológica. Suas facilidades são:

– Ser acessível por qualquer navegador, possibilitar a criação de contas de usuários, remotos ou não, que necessitam desenvolver sua própria ontologia, ou mesmo cadastrar um grupo, permitindo com isso que seus participantes criem juntos uma ontologia.

– Oferecer uma representação semi-formal que permita descrever a linguagem natural e também a representação formal do conhecimento possível de ser interpretada por computador. Ela usa lógica de primeira ordem frame-like plus (definida em KIF) como uma linguagem.

– É apresentada usando o paradigma de Orientação a Objeto em razão da crescente familiaridade do usuário. A apresentação é separada de sua representação interna.

– Usuários podem navegar em ontologias de outros, podem ampliar ou restringir definições existentes na biblioteca.

– Provê as seguintes facilidades de representação em outras linguagens: IDL Corba, Prolog e KIF.

– Uma biblioteca de ontologia que o usuário pode mesclar com sua própria ontologia, dispondo então de um repositório central para ontologias reutilizáveis.

– Suporta sessões interativas, privadas ou de grupos. Gerencia estrutura de dados a fim de administrar estados numa sessão. Pessoas compartilhando uma sessão podem ver modificações que tenham sido feitas por outros membros do grupo e também serem recompensadas pelo tipo de modificação. O servidor da Ontolingua está localizado no endereço http://www.ksl.Stanford.EDU/software/ontolingua/, e pode ser acessado por qualquer navegador de qualquer plataforma.

As funcionalidades da Ontolingua podem ser estendidas por intermédio de outro software, como por exomplo, Chimaera (McGuiness, 2000). Esse aplicativo serve para agrupar ontologias, verificar conflitos de nome, além de permitir navegar e rascunhar ontologias.

c) OilEd

OilEd é o editor para as linguagens *Daml+Oil* e *Oil*; seu propósito principal é suportar a descrição de ontologias e esquemas lógicos.Esse software vem sendo desenvolvido pelo Grupo de Gerenciamento da Informação do Departamento de Ciência da Computação da Universidade de Manchester (http://oiled.man.ac.uk/index.shtml).

O Oil-Ed é similar aos outros editores baseados em *frame* e busca oferecer o suporte ao projeto e manutenção de ontologias. Emprega uma técnica denominada Classificação Rápida de Terminologias, FaCT, do inglês, *Fast Classification of Terminologies* como máquina de inferência, significando que permite ao usuário realizar testes enquanto verifica se a ontologia está consistente; possui mecanismos de integridade e lógica que funcionam nas condições desejadas. Essa ferramenta oferece uma interface que permite utilizar a técnica FaCT, solicitando serviço e resposta a mesma. Isto possibilita ao usuário ver realçadas inconsistências nas classes, além das hierarquias entre as classe poderem aparecer re-arranjadas.

2.2.7 Projetos de ontologia

Muitos projetos têm sido desenvolvidos nos anos recentes visando disponibilizar um quadro de referência que contenha linguagens, editores, analisadores lógicos etc. A seguir apresentam-se alguns esforços, considerados pela autora deste trabalho, como sendo representativos da comunidade de desenvolvedores. Tais esforços são representados por projetos europeus e norte-americanos, incluindo a cooperação do Japão, Singapura e Israel neste assunto.

a) On-To-Knowledge

Atualmente, existe uma grande lacuna entre as necessidades de informação de um usuário e a maneira como a informação a ele chega de forma *on-line.* Neste projeto buscou-se desenvolver ferramentas e métodos de suporte à gestão do conhecimento, tendo como base ontologias reutilizáveis e compartilháveis de conhecimento (Fensel *et al*., 2002). Este é um projeto europeu - IST 1999- 10132–cujo objetivo é auxiliar a criar uma sociedade de informação que seja amigável ao usuário. Ele congrega várias companhias de diferentes portes e centros de pesquisas, e dispõe um orçamento de 2,5 milhões de Euros. Seu enfoque é a Gestão de Conhecimento Orientada pelo Conteúdo, do inglês, *Content-driven Knowledge-Management,* empregando Ontologias Evolutivas (http://www.ontoknowledge.org/). A estrutura tecnológica desse projeto é constituída pelo uso de ontologias para as várias tarefas de integração e mediação da informação. Um dos resultados do projeto On-To-Knowledge é a linguagem Oil.

b) DAML – DARPA *Agent Markup Language*

O DAML(www.daml.org) é um programa da Darpa (*Defense Advanced Research Projects Agency*) com o objetivo de fornecer as bases para a *web* semântica. Esta base compreende

linguagens, ferramentas e técnicas que o conteúdo da *web* deve ser igualmente entendido por pessoas e máquinas (Denker, 2001). O projeto trabalha em consonância com os projetos europeus On-to-knowledge e com os esforços de recomendação da W3C RDF (http://www.w3.org/RDF/).

O esforço é dirigido no sentido do compartilhamento de informações entre mecanismos de software do tipo agentes que, utilizando a representação da informação adequada, podem representar seus usuários em negociações e busca de informações.

A linguagem é baseada em XML e projetada para estende-la em permitir descrever objetos e relacionamentos, expressar semântica e criar um nível mais alto de interoperabilidade entre páginas da Web. Seu desenvolvimento dispõe de um orçamento previsto para três anos de aproximadamente US$ 40,000.00. O projeto já alcançou alguns objetivos como por exemplo gerar uma linguagem de ontologia em primeira versão e a linguagem DAML+OIL é um resultado subseqüente, originado no esforço de trabalho conjunto com o projeto On-To-Knowledge.

O programa possui como estratégia de pesquisa:

i) Criar a linguagem de marcação de agente, DAML, construída sobre XML, que ofereça a usuários pertencentes a comunidades de interesse específico, notações semânticas processáveis em computador;

ii) Criar ferramentas que contenham marcação DAML (DAML *markup*) nas páginas da *web* e outras fontes de informação de maneira a serem transparentes e úteis aos usuários;

iii) Usar estas ferramentas de produção para criar, instanciar, operar e testar conjuntos de programas baseados em agentes que expandam e usem DAML;

iv) Medir, via experimentação empírica, os aprimoramentos de produtividade fornecidos por estas ferramentas;

v) Aplicar estas ferramentas a esforços de desenvolvimento de agente por parte de terceiros e ampliar tecnologias DAML no sentido do uso em larga escala;

vi) Transferir a linguagem DAML para os mercados, comercial e militar, via parcerias.

c) OntoWeb – Ontology-based Information Exchange for Knowledge Management and Electronic Commerce

Este é um projeto europeu – IST 2000-29243 (*Information Societies Technology*)– cujo objetivo é reunir pesquisadores e profissionais da indústria, de modo que todo o potencial das ontologias aumente a troca de informação em áreas tais como: recuperação de informação, gestão de conhecimento, comércio eletrônico e bioinformática. O projeto conta com um orçamento de 1.800,00 Euros e visa desenvolver uma ferramenta para troca de informação baseada em ontologia destinada a suportar a gestão do conhecimento e comércio eletrônico (http://ontoWeb.aifb.uni-karlsruhe.de). Atualmente, os participantes dos projetos OntoWeb, DAML e On-To-Knowledge realizam seus trabalhos em cooperação.

2.3. Agentes

Agentes, genericamente falando, são abstrações para visualizar e estruturar softwares complexos. Agentes neste contexto são uma classe de métodos para solução de problemas. Agentes também são projetados com o propósito de aumentar uma das vistas de percepções de seus usuários, tendo um certo grau de autonomia e raciocínio. Como pôde ser visto na seção anterior, o acesso a informação via Web é uma realidade. O ponto é: como esta informação pode ser acessada e compreensível a pessoas e agentes. E como esta informação pode ser reunida e filtrada, evitando inundar o usuário com dados que não têm utilidade.

Em meados de 1990, as visões de Negroponte e Kay sobre o uso de agentes em funções de interface para realizar algumas tarefas computacionais (Maes, 1997) ,acopladas indiretamente às idéias de Minsky (Middleton, 1999) (aplicação de algoritmos de backpropagation em propostas para inteligência baseada em agentes) conduziram o novo campo de programação baseada em agentes. Experiências com agentes do tipo interface, que aprendem sobre seus usuários e sistemas multi-agentes onde simples agentes interagem para alcançare seus objetivos, dominaram as pesquisas neste campo. Estes sistemas baseados em agentes utilizavam técnicas de inteligência artificial para atingir resultados concretos.

Os modelos de representação do usuário também foram modificados passando a possuir um comportamento dinâmico. Técnicas de máquinas de aprendizado se mostraram particularmente úteis na identificação de padrões no comportamento do usuário.

Ao invés de uma interação iniciada por comandos ou manipulação direta o usuário é engajado em um processo cooperativo no qual pessoas e agentes computacionais podem iniciar a comunicação, monitorar eventos e realizar tarefas.

A despeito do uso extremo do termo agente, existe alguma concordância sobre a definição sobre o que um agente é "um componente de software ou hardware capaz de agir exatamente a fim de realizar tarefas em nome do seu usuário" (Nwana, 1996). Wooldrigdge and Jennings assinalam que para ser um agente, o software ou hardware, ou ambos, necessitam possuir propriedades como autonomia, habilidade social, reatividade e pró-atividade (Wooldrigdge, 1995).

A fim de reunir a informação fundamental sobre teoria de agentes e considerar seu uso neste trabalho, esta seção descreve características e tipos de agentes, agentes inteligentes, multi-agentes e quadros de referência conceitual de agentes.

2.3.1 Características de agentes

Agindo como um representante do seu usuário, um agente possui a obrigação de alertá-lo sobre o que foi incumbido. O trabalho de Charles Petrie (1999) referente a gestão de projeto usando agentes, constitui um exemplo no qual agentes podem ser aplicados ao processo no qual o macro-processo é conhecido, e onde a descrição exata do processo constitui tarefa infinita. O processo do projeto de concepção de um produto não é bem conhecido, apesar do seu macro-processo estar bem claro. O problema está relacionado com o fluxo de trabalho (*workflow*) ou como os processos estão inter-relacionados e a precedência de tarefas entre eles. O objetivo é informar participantes do projeto quando da ocorrência de modificações, no momento em que elas acontecem. O procedimento de coordenação é executado por agentes (sistema Redux), os quais são instruídos como um conjunto lógico de dependências. Um requisito especial estipula que um agente necessita possuir total conhecimento sobre as ações e dependências entre agentes. O agente avalia a propagação das alterações do projeto, acrescentando noções simples de conflitos e lógica de solução, podendo ser derivadas inferências muito úteis. Os agentes rastreiam o aspecto lógico do projeto segundo várias condições e notificam projetistas quando ocorrem mudanças. Mesmo em pequenos projetos, a vista global, às vezes, é perdida, e avisos emitidos por agentes podem revelar conflitos ocultos.

Do processo acima descrito, podemos verificar que agentes agem no sentido de aumentar a percepção das pessoas sobre fatos que elas desejam ser informadas . Principalmente quando estão envolvidos problemas de coordenação, informação dispersa dificultando o trabalho de um coordenador. Este uso de agentes pode evitar re-trabalho, descobertas de conflitos tardiamente e o conseqüente desperdício de tempo.

Em seu trabalho, Jennings (2000) afirma que agentes podem ser articulados com organizações, as quais significam para ele o que hierarquia representa para Booch (uma cadeia de relacionamentos do tipo IS-A SUBCLASS OF). A pesquisa de Jennings e Wooldridge levou-os a concluir que a despeito da similaridade entre agentes e software orientado a objeto, os mesmos possuem diferenças na maneira como reagem. Agentes possuem metas, autonomia e (usualmente) habilidade para cooperar. Também são adequados para representar sistemas complexos de software. Tais sistemas são decompostos hierarquicamente e seus componentes usualmente possuem mais interações internamente do que entre eles. Embora sistemas como esses não sejam descritos plenamente, os componentes maiores o são e podem ser tratados como componentes independentes com uma interação que não é totalmente previsível (aqui pode ser vista a semelhança entre esta descrição e a do sistema Redux de Charlie Petrie).

2.3.2 Tipos de Agentes

De acordo com Maes (1997), os tipos de agentes são: Agentes autônomos, Caracteres Sintéticos, Assistentes Especialistas, Agentes de Sofware, *"Knowbots", and "Softbots"*.

Sycara (1996), definiu e empregou três tipos de agentes: interface, tarefa e informação. Bothorel (1999), propõe três tipos de agentes inteligentes:

– cognitivo (ou social, coarse grained agents); - Sua percepção provém de sua capacidade de analisar o ambiente);

– reativo (ou biológico, fine grain agents); - Reage ao ambiente ou a solicitações; sua percepção resulta da interação coletiva entre agentes não-inteligentes);

– Híbrido – Possui características de ambos.

Este trabalho considera a tipologia definida por Nwana (1996) por incluir todas as características descritas acima. De acordo com ele, tipos de agentes podem ser classificados em

cinco categorias principais que serão discutidas ao longo da seção: possui ou não mobilidade; deliberativo (raciocínio simbólico interno) ou reativo (age / responde numa maneira estímulo-resposta); classificado sob atributos como autonomia, aprendizagem e cooperação; classificado por seus papéis (informação ou agentes de Internet); e agentes híbridos que combinam duas ou mais características. Estas características estão descritas a seguir:

Agentes Móveis – É uma parte de software capaz de percorrer redes, buscar e reunir informação de interesse do seu proprietário. A utilização de agentes móveis pode reduzir custos de comunicação por não exigir a permanência do usuário conectado à rede durante uma transação. Devido aos recursos locais serem limitados, o agente procura outra plataforma onde possa obter informação e buscar recursos específicos que lhe permitam atingir seu objetivo. Este tipo de agente é adequado para computação assíncrona, pois enquanto está sendo executado, o usuário pode até mesmo estar desconectado. É não-convencional e inovativo seu emprego no mercado eletrônico (negociando preços de passagens aéreas) e em telecomunicações (uso em aparelhos móveis), por exemplo.

Agente Reativo – Não possui um modelo simbólico, mas atua/responde ao estado atual do ambiente, num modo estímulo – resposta . É simples e interage com outros agentes em modos básicos. A agência, contudo, pode ser complexa. Um sistema pode ser composto de vários módulos especializados emergindo um comportamento inteligente. Esta pesquisa foi primeiro conduzida por Brooks (Brooks, 1990) não seguiu a abordagem convencional de inteligência artificial. O agente reativo foi usado em componentes *hardwired*, robôs, jogos e indústria de entretenimento.

Agente Colaborativo – Focaliza a cooperação (com outros agentes e pessoas) e autonomia. Esta cooperação possui como característica o uso de uma linguagem de comunicação de agente, constituindo a habilidade social do agente. É adequado para resolver situações que podem ser distribuídas por natureza (ou podem ser decompostas em várias tarefas) onde habilidades sociais são necessárias para negociar melhores condições para o seu proprietário. Necessita negociar para coordenar suas ações com outros agentes e procurar um ponto de concordância mútua em algum assunto. A capacidade de negociar é central para agentes colaborativos e é alcançada com uma biblioteca de protocolos de negociação. O trabalho de Katia Sycara (Projeto Retsina) (Sycara, 1996) é baseado principalmente no emprego desta arquitetura.

Agente para Interface – Enfatiza autonomia e aprendizagem a fim de realizar tarefas de seus proprietários. O fator-chave é um assistente pessoal que colabora com o usuário no mesmo ambiente de trabalho. O trabalho realizado por Pattie Maes (synthetic characters) (Maes,1998) baseia-se, principalmente, em agentes para interface, com o objetivo de liberar usuários das tarefas que podem ser executadas por máquinas automáticas. A pesquisadora também preocupa-se com o aumento da percepção das pessoas, em termos de alertas referentes a novidades e ocorrências que poderiam interessá-las. Assistentes Especialistas enquadram-se bem nesta categoria como Memory Agents , Letizia (Lieberman, 1995) e Expert Finder, pertencem a essa categoria e serão discutidos no capítulo 6 deste trabalho.

Agente de Informação – Auxilia a manipular, gerenciar e coletar informação de muitas fontes (Internet, intranet e data warehouse, por exemplo). Necessita tratar com a quantidade de dados sempre crescente e deles extrair informação proveitosa. *Softbots* da Internet (como Metacrawler projetado pela www.cs.washington.edu (Etzioni,1997)) fazem parte desta classe de agentes. *Knowbots* fazem parte do tipo no qual agentes são usados para extrair informação útil a partir de seus usuários. *Searchbots*, como as máquinas de busca populares para a Web, também são classificados como agentes de informação. O tutorial de Klush (2001) especializou um agente de infomação como:

Cooperativo ou não-Cooperativo – acesso transparente a fontes diferentes e distribuídas, capturando, trazendo e extraindo informação valiosa para seu proprietário ou outros agentes; usa um mediador (*wrapper* ou *broker*) a fim de traduzir as ontologias usadas, decompor e executar consultas complexas, mesclar as respostas fornecidas por fontes diferentes. O colaborativo usa linguagens de comunicação como protocolos Fipa (www.fipa.org) ou KQML (http://www.cs.umbc.edu/kqml). InfoSleuth (http://www.argreenhouse.com/InfoSleuth/index.shtml) e Retsina são exemplos de agentes cooperativos de informação. O agente não-cooperativo adapta-se ao usuário, rede ou ambiente por si próprio. Interage com seu usuário por intermédio de mecanismos como computação afetiva (*affective computing*, Picard, 1998), onde as emoções do usuário são analisadas através do reconhecimento da inflexão da voz ou da expressão facial. O agente pode usar reforço de aprendizagem para formar suas crenças e ampliar sua percepção do usuário a partir dos movimentos deste.

Adaptativo – necessitam tratar com fontes de informação incompletas, incertas e vagas e têm de tomar decisões dinamicamente. Pode aprender, por exemplo, por analogia ou descoberta . Pode empregar processo de aprendizagem reforçada para realimentar sua reação, ou aprendizagem supervisionada ou não. São aplicações atuais: comércio eletrônico, coleta de informação na Web, manufatura, logística e redes de telecomunicações.

Racional – Precisa tratar com comércio mediado automaticamente (teoria de utilidade multi-atributo), formação de alianças entre agentes, protocolos baseados em leilão e esquemas de arbitragem.

O agente informação precisa tratar com diferentes fontes de dados e deles extrair seus significados. Os esforços da "Web semântica" seguem nesta direção, a fim de oferecerem meta-dados para criar uma interpretação comum de informação, permitindo significado intercambiável. Mesmo atuando numa maneira não colaborativa com outros agentes, ele necessita tratar com a semântica da fonte de informação, se quiser ser uma vista representativa do seu usuário.

Tabela 2.2. Características de agentes (Fonte: Recticular Systems, 1999)

Agentes	Autonomia: executam autonomamente
	Habilidade social: comunicam-se com outros agentes ou o usuário
	Reatividade: monitoram o estado do ambiente de execução
Agentes Inteligentes	Capazes de usar símbolos e abstrações
	Capazes de extraírem quantidade significativa do conhecimento do domínio
	Capazes de se adaptarem a um comportamento orientado a meta
Agentes Verdadeiramente Inteligentes	Capazes de aprender a partir do ambiente
	Tolerantes a erro, à entrada inesperada ou contendo erro
	Capazes de operar em tempo real
	Capazes de se comunicarem usando linguagem natural

A tabela 2.2 resume as características dos agentes apresentadas acima destacando o fator inteligência em ordem crescente de capacidade de dedução lógica e reação em tempo real. O agente utiliza outros recursos computacionais para a consecução de suas metas.

2.4 Máquina de Aprendizado – redes *Bayesianas*

Nesta seção será apresentada uma introdução sobre redes *Bayesianas*. Segundo autores da área de inteligência artificial (Sebastiani, 2002; Hugin Expert, 2002; Petrou, 1999, Zhang, 2000; Hong, 1999), é considerado um método válido e eficaz com o qual máquinas de aprendizado decidem sobre um determinado assunto. As redes Bayesianas são utilizadas em diversas aplicações sendo a caracterização de padrões uma delas.

Uma rede *Bayesiana* (rede probabilística causal, rede acreditada *Bayesiana* ou rede acreditada) é um modelo compacto de representação de decisão sob incerteza. Um problema determinado – diagnóstico de falha mecânica, por exemplo – consiste em um número de entidades ou eventos. Estas entidades ou eventos são, em uma rede *Bayesiana*, representados como variáveis aleatórias. Estas variáveis aleatórias estão ligadas por arcos representando a possível relação de dependência entre os eventos ou entidades representadas pelas variáveis aleatórias.

A incerteza do problema é representada através de probabilidade condicional. Uma rede *Bayesiana* consiste de uma parte qualitativa que descreve as relações de dependência do sistema e uma parte quantitativa, que descreve as crenças sobre as forças das relações.

2.4.1 KEA – Algoritmo para extração de frases-chave

O KEA (http://www.nzdl.org/Kea/) (Frank, 1999) é um programa para extração de frases chaves disponível na *Web* escrito em linguagem Java e Perl. Os algoritmos baseados em Redes *Bayesianas* são utilizados para diferentes propósitos, como em sistemas de auxílio à decisão aplicados em Medicina, Processamento de Informações, Economia e outros (Hugin Expert, 2002) e o KEA é um programa especializado que utiliza as redes no seu processo de aprendizado e inferência sobre frases chave contidas em um documento. O KEA foi utilizado nesta tese por ser um programa disponível livremente, ser escrito em Java além das características de máquina de aprendizado ter se mostrado, em testes, mais adequados que outros programas com a mesma finalidade. A adequação para esta tese foi considerada em termos de tempo de processamento do algoritmo e eficiência no processamento de arquivos novos quando o treinamento foi pequeno ou não aconteceu.

Extração de frases chave é uma tarefa de classificação. Cada frase, nesta tarefa de classificação, pode ser ou não uma frase-chave. Utilizando-se a terminologia de máquinas de aprendizado, frases em um documento são consideradas "exemplos" e o problema do aprendizado é encontrar o mapeamento do exemplo em duas classes, as frases chave e as não-frases chave. Quando treinadas com um conjunto exemplo as máquinas de aprendizado podem gerar automaticamente estes exemplos e depois aplicar este mapeamento em novos documentos. A extração de frases chaves utilizando-se o KEA ocorre em duas etapas: treinamento da máquina, onde um conjunto de documentos é escolhido para como conjunto treinamento, e extração da palavra ou frase chaves.

Seleção das frases candidatas

Para efetuar a seleção das frases candidatas o documento do qual as frases serão retiradas é "limpo". Esta limpeza significa a retirada de marcas de pontuação, colchetes, parênteses, números, caracteres alfa-numéricos, etc. Acronismos não são eliminados.

Identificação das frases

As frases candidatas não possuem mais que três palavras, não podem ser nomes próprios e não podem começar ou terminar com *stopwords* (são palavras de nove classes sintáticas como artigos, advérbios, etc). Além disso é utilizado um eliminador de sufixos de modo que as palavras fiquem na sua forma raiz (de modo que palavras como *cut elimination* e *cut elim* são consideradas equivalentes).

São calculadas duas característica para cada frase candidata o vetor TFxIDF (*Term Frequency X Inverse Document Frequency*) que constitui a medida da ocorrência de uma frase em um documento comparado com sua ausência no uso geral, e a primeira ocorrência no documento sob análise, que é a distância da primeira ocorrência da frase em relação ao início do documento. O uso geral é representado pela freqüência do documento, que é o número de documentos contendo a frase no *corpus* (conjunto de documentos sob análise).

Os resultados da aplicação do KEA foram analisados comparando seus resultados no mesmo conjunto de documentos por outros tipos de classificadores e seus resultados foram considerados iguais ou melhores quanto à eficácia do algoritmo e quanto à rapidez em obter resultados (Frank, 1999; Witten et al. 1999).

2.5 Conclusões

As seções anteriores mostraram que a Ontologia (letra maiúscula) provém de aplicações filosóficas e recentemente tornou-se um tema de pesquisa, principalmente na área de Inteligência Artificial. As ontologias ganharam notoriedade pelas seguintes razões: necessidade de compartilhar a mesma percepção e significância do mundo, evitar desperdício de tempo e utilizar a informação disponível de maneira correta, a organização da informação e sobretudo a organização da informação sobre informação (meta-informação). Importantes instituições de ensino e pesquisa somam esforços a empresas; projetos com orçamentos elevados lideram iniciativas voltadas a ontologia, como é o caso do projeto DAML, porém iniciativas da organização W3C (http://www.w3.org/) também têm contribuído para o estabelecimento de recomendações para formatos de informações em compartilhamento de conhecimento.

Agentes representam uma classe de software apropriada para lidar com o problema da distribuição funcional associada à distribuição física. Possui características do tipo cooperação com outros agentes e com pessoas, auxilia pessoas na busca por informação ou representando o seu usuário em situações específicas. Associado com outros programas como ontologias e máquinas de aprendizado pode aprender e sugerir informações, atuações ou mesmo decidir por uma determinada solução.

O próximo capítulo traz os requisitos que um sistema computacional deve possuir para auxiliar no compartilhamento de conhecimento entre pessoas. Com auxílio de agentes e ontologias é possível visualizar uma solução automática para auxiliar na cooperação entre pessoas.

Capítulo 3

Requisitos e Arquitetura do Sistema Proposto

O capítulo anterior discorreu sobre os conceitos teóricos referentes a sistemas de auxílio ao compartilhamento de conhecimento, máquinas para representação de usuário e máquinas de aprendizado. Este capítulo estabelece critérios sobre os requisitos que esses sistemas devem satisfazer de modo a poderem ser considerados habilitadores de trabalho cooperativo, apresenta um estudo sobre tais requisitos e propõe uma arquitetura funcional para a ferramenta de ajuda ao compartilhamento de habilidades que satisfaça os referidos requisitos.

3.1. Introdução

Este trabalho discorre sobre a construção dinâmica de uma rede de habilidades e propõe uma ferramenta de auxílio ao compartilhamento de conhecimento como suporte à criação dessa rede. A ferramenta é idealizada de modo a contemplar as necessidades do usuário, significando que este define os domínios de conhecimento de seu interesse e então ela busca, identifica e informa-o sobre comunidades que possuam interesses iguais ou similares aos seus. Essa operação dirigida pelo usuário, faz com que a ferramenta respeite suas escolhas e preferências, viabilize a criação da referida rede e auxilie o compartilhamento de conhecimento. A descrição a seguir expõe a análise, concepção, projeto e desenvolvimento da referida ferramenta, segundo os seguintes passos:

- estabelecimentos de critérios para definição de requisitos que sistemas de compartilhamento de conhecimento devem obedecer;
- definir o papel da(s) pessoa(s) segundo estes requisitos;
- definir o papel de um sistema computacional que apóie esta(s) pessoa(s).

A fase de concepção de um processo ou produto será o foco em termos de relacionamento temporal que norteará o de estabelecimento de requisitos. A arquitetura deste sistema será desenhada enquanto a ferramenta implementada, que é considerada uma instância do sistema, estará detalhada nos dois próximos capítulos.

3.2. Definição dos critérios para escolha de requisitos

A formação de rede de habilidades se apoia em um sistema que realiza aquisição de conhecimentos e faça recomendação aos seus usuários. Este sistema deve obedecer a alguns requisitos de modo a ser adaptável, integrável a outros sistemas, requeira poucos recursos de processamento e também possa ser utilizado com confiança pelo usuário.

A fim de estabelecer os critérios que devem nortear o estudo de tais requisitos, foram investigados os critérios organizacionais que contribuem para avaliar o que leva pessoas a compartilharem conhecimento na fase de desenvolvimento de produtos. Os estudos de caso investigados indicam que a cooperação entre parceiros, sejam eles fornecedores ou usuários, desde a fase de concepção do produto, é fundamental para que o mesmo seja manufaturado e tenha êxito.

A cooperação das diversas unidades empresariais envolvidas na concepção de um produto, com as múltiplas contribuições que este processo proporciona, também mostra-se decisiva para que a fase de concepção conheça as fases subseqüentes e o produto seja manufaturado e entregue. Ao mesmo tempo, examina-se detalhadamente como ocorre esta cooperação, tendo-se em consideração as muitas barreiras a serem enfrentadas por esta forma de operação, por mais que seja declarada necessária. Barreiras culturais colocando em cheque a credibilidade da fonte, barreiras pessoais do tipo: “Afinal, se conhecimento é tão precioso porque alguém deveria dá-lo de graça?”.

Quando uma companhia declara que seu bem mais precioso é o conhecimento e pressiona seus funcionários no sentido de compartilhá-lo, os resultados dos estudos são praticamente unânimes em declarar este como o pior caso, onde raramente ocorre compartilhamento de conhecimento; ao contrário, forma-se uma resistência a este tipo de prática. Outro fator é o risco financeiro envolvido na fase de concepção: ele é menor que o risco tecnológico.

Os requisitos de coordenação e participação interfuncional, cooperação eficiente com parceiros externos, proposição de bons conceitos, redução do tempo de desenvolvimento e desenvolvimento de produto com alto grau de inovação são considerados por Harmsen (2000) como competências necessárias quando do desenvolvimento de produtos. McDonough (2000) também investigou como a cooperação interfuncional contribui positivamente ao desenvolvimento de novos produtos, quando coordenado por lideranças reconhecidas pela equipe envolvida.

O estudo realizado por Dröge *et al* (2000) sobre empresas na área automobilística também relata que organizações abertas, isto é, caracterizadas por uma comunicação horizontal entre equipes e pouca interferência das hierarquias gerenciais, são fatores primordiais para o sucesso das mesmas como empresas inovadoras. Quanto a sistemas facilitadores de compartilhamento de conhecimento, o estudo realizado por Jarvenpaa (2000), apresenta hipóteses sobre condições culturais que poderiam levar a um comportamento cooperativo utilizando-se tecnologia de informação. O estudo de campo desenvolvido por Kasanjian (2000) sobre a cooperação entre equipes multifuncionais em projetos inovadores (e criativos) de grande porte, presta esclarecimentos sobre:

- a necessidade de uma rede multifuncional de discussão apoiada não só pelo vis-à-vis mas também pela tecnologia de informação
- a necessidade de forte coordenação para relacionar essas equipes de modo que suas ações não sejam conflitantes em termos de decisões de projeto.

A pesquisa realizada por Moenaert (2000) focaliza exclusivamente os mecanismos de comunicação exigidos por estas equipes multifuncionais (e também internacionais) e conclui sobre os requisitos de comunicação para que haja efetivo envolvimento dessas equipes.

Em termos de tecnologias de informação, buscou-se na engenharia concorrente, onde a noção de afinidade de grupos de trabalho se fundamenta na capacidade de compartilhamento de conhecimento, os requisitos funcionais que apoiem a cooperação e auxiliem a coordenação de equipes multifuncionais geograficamente dispersas, como é o caso dos projetos Cairo (Peña-Mora, 2000), Acácia (Dieng, 2001) e ProcessLink (Petrie, 1997).

O estudo realizado por Drira (1998) traz como contribuição as regras de cooperação segundo um modelo funcional em três níveis – cooperação, coordenação e comunicação –para aplicações distribuídas cooperativas. A pesquisa executada por Cagliano (2000) sobre colaboração tecnológica, investiga os modos de cooperação segundo as formas organizacionais de colaboração técnica. Essa colaboração tem lugar quando uma empresa compra outra e adquire os direitos sobre a produção desta empresa ou na formação de parcerias entre centros de pesquisa, universidades e empresas, entre outras possibilidades. Este trabalho dedica especial atenção a essa pesquisa porque ela mostra os diversos tipos de relacionamentos em uma situação de cooperação tecnológica.

O estudo de Mäntylä (2000) sobre os fatores que conduzem a uma situação de cooperação (coordenação e comunicação) em empresas virtuais, conduziu a uma solução tecnológica de construção de uma ontologia de processo e à coordenação do processo utilizando agentes.

Esta seção examina os requisitos do sistema de apoio ao compartilhamento de habilidades, segundo três critérios: cooperação, comunicação e coordenação. Eles foram escolhidos a partir das pesquisas efetuadas e embora priorizem um ou outro aspecto, estão intrinsecamente relacionados tanto aos três requisitos quanto às fases de concepção e projeto de um produto. Esses critérios são categorizados em três camadas interdependentes como descritas a seguir:

- Cooperação – Compartilhamento de informação é um ato de cooperar. Investiga-se como pessoas e sistemas satisfazem condições de atuação no mesmo sentido;
- Coordenação – A fim de atingir um estado de cooperação torna-se necessária uma ação de coordenação, de modo que esforços não sejam dispersos, ao contrário, sejam convertidos para a mesma direção, gerenciando-se conflitos e direcionando-se o fluxo de informações;
- Comunicação – Para assegurar o fluxo de informação o uso de canais claros e acessíveis de informação é essencial.

A apresentação dos requisitos que satisfazem os critérios foi organizada segundo a visão dos participantes da atividade de compartilhamento de conhecimento e a visão dos sistemas computacionais que proporcionam facilidades para que tal atividade seja realizada com sucesso.

3.3. Requisitos de Cooperação

Estes requisitos foram investigados considerando-se duas visões: a visão do usuário – exprime as característica de um participante do projeto –, e a visão dos requisitos do sistema computacional que o apoia. Esta organização visa clarificar o comportamento de pessoas nas diferentes situações que permeiam a atividade de cooperação.

3.3.1 Visão do Usuário

Cooperação é o processo de apoiar o trabalho conjunto de pessoas de diferentes perfis funcionais, provenientes de áreas distintas intra e interempresa, e que possuem habilidades diversas. Pessoas envolvidas em processo de compartilhamento de informações podem possuir diferentes níveis de conhecimento. O impacto destes múltiplos perfis é detalhado por Dröge *et al*. (2000); estes autores assinalam dois pontos no relacionamento entre múltiplas funcionalidades: a riqueza proveniente da multidisciplinaridade e o fórum de participação e controle na implementação de projetos. Pessoas participantes de um processo de cooperação se comportam como uma equipe quando cada membro atua considerando seu parceiro como uma fonte de conhecimento.

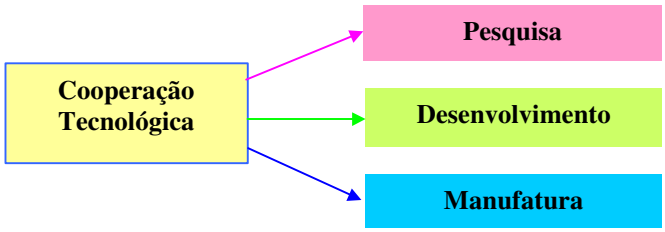


Figura 3.1 Fases sob análise da cooperação tecnológica do ciclo de vida do produto

Segundo o critério de cooperação sob consideração, um estudo elucidativo, embora não possa ser generalizável, foi realizado por Rafaella Cagliano *et al.* (2000). Este estudo investigou as diferenças nos modos organizacionais de cooperação tecnológica presentes nas fases de Pesquisa, Desenvolvimento e Manufatura do ciclo de vida de um produto em um processo de inovação tecnológica ilustrado na Figura 3.1.

A Figura 3.2 mostra que o Alvo da cooperação tecnológica envolve os motivos, o teor e o tipo de parceiros envolvidos na cooperação que serão conceituados a seguir:

– Motivação: pode originar-se da busca por excelência, ou conveniência econômica, maior velocidade em introduzir novos produtos, propensão a pertencer a novos mercados e ter acesso a tecnologias complementares. Uniões do tipo *Joint-ventures* estão ligados a relacionamentos de longo prazo; acordos informais trazem maior flexibilidade enquanto consórcios visam minimizar riscos ao reunir capital e recursos externos.

– Teor da Cooperação: caracterizado pela natureza dos recursos trocados na cooperação e a possibilidade de definir seu escopo claramente. A tangibilidade do objeto de cooperação faz variar o relacionamento de cooperação de um relacionamento horizontal e informal a uma colaboração hierárquica e com contratos formais.

– Tipologia dos parceiros: significa a posição relativa na cadeia de valores e determina a natureza dos relacionamentos entre os colaboradores e o nível de confiança recíproca. Cooperações informais são usuais entre pares enquanto que cooperações entre fornecedores e clientes costumam ser mais formais.

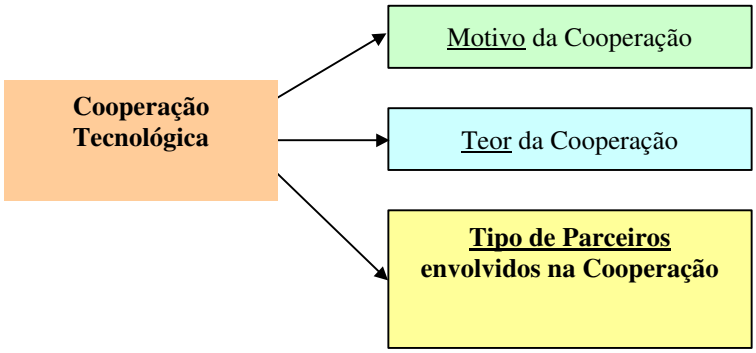


Figura 3.2 Alvo da Cooperação Tecnológica (Cagliano, 2000)

A forma de organização envolve o número de parceiros, o formalismo contratual, a estrutura de controle, o horizonte de tempo e a densidade do relacionamento, como ilustrada na Figura 3.3. Os critérios para análise da forma de cooperação organizacional variam quanto a:

– Formalização do acordo: varia do totalmente informal, baseado na confiança, a um contrato formal baseado em normas e regras escritas.

– Estrutura de controle: é caracterizada pelo poder dos diferentes parceiros para influenciar decisões e ações e o objeto controlado. O controle pode ser centralizado em uma unidade, pode ser compartilhado porém delegado a uma unidade de coordenação ou pode ser compartilhado entre parceiros. As fontes do poder assimétrico podem ser econômicas, tecnológicas ou de

competência ou o nível de confiança entre companhias. Por outro lado o controle pode estar ligado a resultados e atividades, ou seja, o modo como objetivos devem ser alcançados.

– Horizonte de Tempo: o tempo no qual companhias irão cooperar influencia a forma de cooperação. A cooperação entre companhias pode ser ocasional ou recorrente;

– Densidade dos relacionamentos: indica a freqüência e intensidade das trocas. Depende da divisão de tarefas e na facilidade da identificação do conteúdo (teor) da troca.

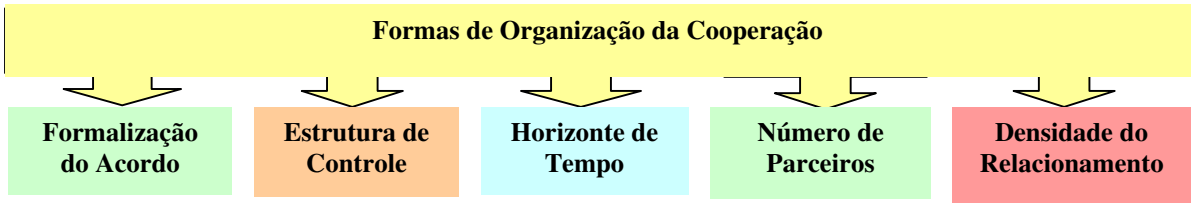


Figura 3.3 Critérios para análise da forma de organização da cooperação (Cagliano, 2000)

i) Cooperação na fase de Pesquisa

Nesta fase o alvo da cooperação é investigado quanto a:

– Conteúdo – Como o resultado não pode ser precisado *a priori*, porque refere-se a atividade exploratória, o teor da cooperação tem um caráter genérico, ocorre principalmente no que diz respeito à técnica (*know-how*) e competência que estão presentes nas pessoas. É uma cooperação que requer transferência de conhecimento entre parceiros. Nesta fase, as atividades são caracterizadas pelo alto risco de falha tecnológico sendo que o risco financeiro ou comercial ainda não é significativo;

– Motivação – Reduzir ou limitar o risco tecnológico e alcançar massa crítica para o esforço em pesquisa. Custos em pesquisas são crescentes e alcançar a massa crítica necessária representa dificuldade, principalmente para pequenas e médias empresas. Acessar e integrar diferentes disciplinas tecnológicas e conhecimento. Kodama (1992) enfatiza que inovação é mais freqüentemente o resultado da combinação de esforços de pesquisa em diferentes disciplinas científicas do que o esforço concentrado em somente um campo específico da tecnologia. Ampliando ou aprofundando o conhecimento da companhia em certas disciplinas tecnológicas, através do acesso a centros de excelência, pode também levar ao comportamento cooperativo. A cooperação é mais intensificada estimulando-se a criatividade

e inovação, conectando pessoas com diferentes culturas, experiências e conhecimento;

– Parceiros envolvidos – Em pesquisa, as parcerias mais comuns são as entre as companhias e universidades e centros de pesquisa, porque estes últimos provêem competência adequada em organizar e gerenciar projetos de pesquisa. Alianças com companhias inovadoras, as que possuem conhecimento especializado em algum tipo de tecnologia, ou com competidores, quando estas companhias, além de possuírem um objetivo comum têm seus papéis e garantias bem asseguradas.

ii) Cooperação na fase de Desenvolvimento

Nesta fase a cooperação é examinada quanto aos seguintes aspectos:

– Conteúdo – usualmente com o escopo definido já que um novo produto ou processo deve ser desenvolvido. O resultado da cooperação pode ser descrito de maneira precisa. Esta cooperação diz respeito a recursos tangíveis tais como Plantas, maquinário ou bens físicos. O conhecimento tecnológico também é trocado porém não é o ponto principal. A colaboração, nesta fase de desenvolvimento do produto, é caracterizada pelo alto risco financeiro ou comercial característica desta fase do processo.

– Motivações – Reduzir o tempo e custo necessários ao desenvolvimento compartilhando o esforço com outros parceiros. Redução de riscos comerciais. Ter acesso aos recursos especializados (que faltam à companhia) para o desenvolvimento de um novo produto. Definição de um padrão tecnológico para novos produtos principalmente os ditos "acoplados", ou seja, produtos compatíveis com outros.

– Parceiros envolvidos – Fornecedores, cuja colaboração é fundamental no desenvolvimento, particularmente quando a qualidade e funcionalidade dos materiais, componentes e maquinário fabricados e utilizados pelos fornecedores são fatores críticos para a garantia da qualidade do produto final. Clientes, que podem auxiliar a companhia a entender as características dos resultados desejados pelo mercado, reduzindo o risco comercial. Firmas, que possuem os recursos especializados para o processo de desenvolvimento. Competidores usualmente se envolvem nesta fase quando o mote da cooperação é definir um padrão a ser imposto ao mercado.

iii) Cooperação na fase de Manufatura

Investiga-se a cooperação segundo os seguintes aspectos:

- Conteúdo – É definido e concerne um escopo preciso e algumas vezes limitado que é a fabricação de um produto ou componente. Os itens trocados são, majoritariamente, tangíveis, concretos. As informações trocadas dizem respeito às interfaces entre as unidades envolvidas, de modo que suas atividades possam ser coordenadas. As atividades conjuntas consistem em tomadas de decisão e solução de problemas. O risco envolvido é geralmente baixo, tanto tecnológico, quanto financeiramente, enquanto o risco de mercado é algumas vezes alto;
- Motivações – Alcançar uma economia de escala, integrar competência complementar, utilizar fornecedor externo (*outsourcing*). Muitas vezes esta utilização é crítica ao sucesso de algumas empresas. Aproveitar uma oportunidade de mercado difícil de satisfazer com a capacidade disponível na própria empresa;
- Parceiros envolvidos – A tipologia dos parceiros depende da natureza da tarefa sendo particionada de modo que, neste caso, a cooperação envolve fornecedores e sub-contratados para alcançar um nível tecnológico específico e de integração operacional. Companhias trabalhando em ramos industriais diferentes podem cooperar de modo a complementar suas tecnologias ou explorar outros mercados. Competidores que necessitam atingir economia de escala ou massa crítica em poucos ou um recurso.

Os trabalhos de pesquisa acima referenciados conduzem a algumas considerações sobre o significado de cooperação por parte de um usuário: cooperação pressupõe troca, mesmo que seja de um recurso intangível, não envolvendo, necessariamente, aspectos comerciais. Em síntese, a fase do ciclo de vida do produto que envolve maior troca de conhecimento é justamente a fase de pesquisa, onde as relações são mais fluidas, amplas e os relacionamentos mais informais. Nesta fase as organizações envolvidas são centros de pesquisas, universidades e companhias se relacionando através de convênios ou parcerias.

A próxima seção analisa sistemas computacionais que possam suportar este tipo de interação entre pessoas e companhias.

3.3.2 Visão do Sistema

Projetos baseados em padrões (*pattern-based*) são adequados para o desenho de sistemas de compartilhamento de conhecimentos (Clark, 2000). É recomendado mesmo quando do reprojeto

de sistemas (Wilhem, 2000). Os modelos advindos destes padrões têm como objetivo auxiliar as equipes relacionadas em criação cooperativa criando canais onde pessoas podem contribuir, criticar e discutir entre si. Além dessa possibilidade, estes modelos auxiliam os usuários finais a descobrir e usar os recursos do sistema. O uso de UML (Felfering, 2000) por diferentes equipes em diversos projetos, também mostrou que o uso de um modelo que seja entendido tanto pela equipe desenvolvedora (em diferentes companhias e diferentes países envolvidas na construção da mesma ferramenta de programação) quanto pelo usuário é determinante para a mútua compreensão elevando a competência técnica e auxiliando a alcançar mais rapidamente o objetivo comum (Molina-Espinoza, 2001).

Falta de envolvimento leva à falta de compromisso (Moenaert, 2000, Nihtila, 1999). Se as pessoas acreditam umas nas outras o processo de cooperação torna-se mais fácil de ser viabilizado. O uso intensivo de recursos eletrônicos e envolvimento da equipe (usuários, pesquisadores e desenvolvedores) desde a fase inicial de concepção e pesquisa é aconselhada por vários estudos de caso (Vandapalli, 2000). Usualmente, o conhecimento encontra-se disperso em várias fontes: arquivos em papel, arquivos eletrônicos em formatos antigos e atuais, bases de dados, a *web* e a memória das pessoas. Compartilhar eletronicamente o conhecimento significa:

- reutilizar antigos sistemas: como incorporar informações em formatos antigos a novas bases de dados;
- construir uma base de conhecimento que contenha melhores práticas e também diagnósticos de falhas;
- ampliar os modos de compartilhamento de informação de forma a possibilitar o uso das especialidades de cada participante.

A presença física dos componentes da equipe para um primeiro contato é importante mesmo quando ferramentas sofisticadas de trabalho em grupo estão disponíveis (Moenaert, 2000). Uma vez a equipe formada, pode ser apoiada por uma variedade de ferramentas que atenda suas necessidades como tele-reuniões, tele-conferências, compartilhamento de diversos tipos de arquivos e mesmo acompanhamento à distância de testes de simulação.

Existe uma certa dificuldade em utilizar-se estas ferramentas devido à dificuldade em integrá-las. Contudo, os portais têm sanado a maior parte dos problemas relativos a acesso a

documentos comuns, além do que, algumas ferramentas permitem o agendamento de encontros sistemáticos de modo que a dinâmica de uma equipe possa ser preservada mesmo à distância. Muitas vezes, como ocorreu no projeto DSE (http://cec.to.alespazio.it/DSE/home.htm), uma única interface homem-máquina deve ser construída de modo a facilitar a tarefa do usuário.

De acordo com estes requisitos, uma ferramenta que auxilie à atividade de cooperação deve demandar pouco processamento, ser não-invasiva, facilmente integrável a outras aplicações e possuir uma interface simples.

Cooperação para este sistema consiste em usar informação disponível levando-se em consideração que a interação entre pessoas dependerá da sua própria vontade, além dos objetivos do empreendimento. Uma ferramenta que aponte a fonte de conhecimento e mostre suas coordenadas, auxiliando a construção de um núcleo de competência proveniente de diferentes entidades, considerando-se não somente as empresas, mas também universidades e centros de pesquisa.

Examinando-se a tecnologia disponível para implementar tal sistema, constata-se que a denominada "orientada a agentes de informação" parece ser apropriada, em razão de permitir lidar com aspectos tais como: barreiras de linguagem, diferenças de aptidão por parte do usuário, diferenças no formalismo em representar conhecimento; e ainda assegurar um caminho entre o usuário e especialistas em potencial. O uso associado de agentes de informação e ontologias podem vir a compor um sistema cujo objetivo seja o compartilhamento de conhecimentos.

Áreas do conhecimento devem ser representadas por um robusto, consistente e flexível meta-modelo que forneça a base para o funcionamento do agente. Este meta-modelo deveria utilizar uma linguagem de projeto poderosa e popular de modo a permitir que seu desenvolvimento possa ser seguido e acompanhado por pessoas com diferentes conhecimentos, como UML (*Unified Modeling Language*).

O agente por sua vez deveria ser desenvolvido utilizando-se metodologia orientada a agentes de modo a clarificar serviços, eventos associados, máquinas de estado e todos os aspectos associados ao comportamento de tal agente.

Começou-se a esboçar o perfil de um sistema que possa auxiliar um usuário no processo de cooperação tecnológica com outra pessoa, porém não exclusivamente, na fase de pesquisa de um

processo de inovação ou início de um projeto. Os itens seguintes exploram os requisitos referentes à coordenação e comunicação que um sistema deve satisfazer, considerando-se a construção de uma ferramenta que auxilie o compartilhamento de conhecimento, de modo a compor-se um quadro de requisitos que norteará o desenho e confecção de tal ferramenta.

3.4. Requisitos de Coordenação

Coordenação significa organizar a informação retirada do ambiente, especificar onde deve ser guardada e conhecer os pontos para sua recuperação (evitando gargalos das fontes principais e também recuperar muitas vezes a mesma informação). O objetivo da coordenação é alcançado quando as pessoas conseguem usufruir o conhecimento do grupo (time, equipe).

3.4.1 Visão do Usuário

No processo de coordenação, os envolvidos devem ter um papel bem definido, estabelecendo de modo claro quem necessita de qual tipo de informação e em qual tempo.

A teoria da coordenação desenvolvida por Malone e sua equipe têm, em um de seus artigos (Malone, 1994), a pergunta primordial: “como o uso generalizado da tecnologia de informação afeta a maneira das pessoas trabalharem juntas?”. O que se quer é auxiliar as pessoas a trabalharem juntas melhorando a qualidade do seu conhecimento sobre o conhecimento dos outros utilizando para isso a tecnologia da informação. Neste artigo (e nos subseqüentes, influenciando toda uma linha de pensamento de diversos pesquisadores) é colocada a multidisciplinaridade da teoria da coordenação. Ao mesmo tempo em que analisar casos e casos onde a interdependência entre atividades aumenta o conhecimento sobre coordenação, como também, fornecer um guia para analisar as necessidades da coordenação em situações específicas e gerar modos alternativos de lidar com elas. Muitos dos trabalhos residem no gerenciamento de recursos compartilhados, que é uma situação crítica para a coordenação. Outros em uma relação produtor-consumidor muito comum em sistemas de manufatura, onde algum tipo de notificação deve existir para indicar a disponibilidade da atividade.

Para uma equipe, a figura do coordenador tem caráter de liderança e pode ser proveniente da organização ou pertencer a uma equipe supra organizacional de modo a ter uma visão do todo e neutra em termos de fontes assimétricas de poder. O coordenador possui características de elemento facilitador indicando a fonte de informação a quem dela necessita. Embora o

coordenador não tenha tarefas específicas a cumprir, ele possui orientação sobre as tarefas que estão sendo realizadas e por quem (Dean, 1997). Ele também possui um conhecimento multifuncional e multidisciplinar de modo a poder estar atento às necessidades de interação entre especialistas para atingirem seus objetivos.

Tabela 3.1 Relação entre as dependências e os processos de gerenciamento

Dependência / **Processo**	Recursos compartilhados Atribuição de tarefas	Transferência, logística	Projetos de manufatura	Restrições simultâneas	Tarefas e sub-tarefas
	Atendimento por ordem de chegada, prioridade, orçamento, mercado, etc	*Just in time* Ordem econômica	Engenharia concorrente	Escalonamento, sincronização	Seleção de objetivos Decomposição de tarefas

Definido por Malone "como o processo para gerenciar dependências entre atividades", essas dependências e seus processos estão resumidos na tabela 3.1. As atividades resumidas na tabela representam os conflitos habitualmente presentes nas relações de projetos de engenharia como por exemplo, projetos distribuídos de produtos, onde a engenharia concorrente traz, com seus métodos e ferramentas, as propostas de solução.

3.4.2 Visão do Sistema

O problema de coordenação entre as organizações participantes de uma cadeia de fornecedores, por exemplo, pode ser direcionado pela organização da cadeia em forma de rede de agentes cooperativos (Barbuceanu, 1997). Vários problemas de coordenação são equacionados utilizando-se soluções baseadas em agentes. Fox (2000), Sycara (1999) , Petrie (2000) e Dieng (2001) decompõem o problema de modo que agentes que possuem um conhecimento do todo verificam a ação dos que possuem tarefas específicas e dirigem a informação aos elementos que dela necessitam, fazendo análise de pertinência e de tempo. A maior questão destes sistemas refere-se à disputa por recursos, ou seja, a concorrência do sistema. Cada um dos sistemas possui uma linguagem de coordenação, contudo, suas arquiteturas são semelhantes. Tem sido uma

abordagem freqüente dispor de um elemento que possui a visão geral do que ocorre no sistema e tem autoridade de reordenação.

Sem coordenação os benefícios advindos da solução de problemas com opção descentralizada desaparecem, e a comunidade de agentes pode rapidamente evoluir para uma situação caótica (Chauhan, 1997). Em sistemas multi-agentes, os agentes perseguem individualmente seus objetivos, sendo que suas ações são restringidas pelas ações uns dos outros. O problema de coordenação surge quando: i) existem ações alternativas ou ii) a ordem e o tempo de execução das ações resultam em diferentes estados de ocorrência (Barbuceanu, 1997).

O problema de coordenação entre as organizações participantes de uma cadeia de fornecedores é endereçado pela organização da cadeia em forma de rede de agentes cooperativos. Para cooperar distinguem-se as camadas de conhecimento sobre interação social dos seus aspectos de conhecimento sobre solução de problemas (Nwana *et al*, 1996).

A proposta de Nwana *et al* (1996) referente às razões pelas quais as ações dos agentes devem ser coordenadas são:

– Existem dependências entre as ações dos agentes que podem ocorrer quando seus objetivos são relacionados;

– Necessidade de satisfazer restrições globais. Essa necessidade existe quando as soluções desenvolvidas pelo sistema devem satisfazer certas condições para serem consideradas sucesso;

– Nenhum agente, individualmente, possui competência, recursos ou informação para solucionar todo o problema.

A preocupação com coordenação origina-se no fato de que uma colônia de agentes é muito mais capaz – eficiente, veloz, considera muitas possibilidades de solução – de ter êxito na solução de problemas que agentes individuais; e no problema de coordenação estar relacionado com competição por recursos.

A fim de tratar a questão de coordenação na cadeia de suprimentos, a solução proposta por Barbuceanu e Fox (2000) segue o mesmo raciocínio de Petrie (2001), ou seja, decompõe o problema em subatividades de modo que uma delas, - a Logística –, conhece as atividades

realizadas pelas outras, possui uma visão global sobre planos e prazos, podendo negociar com o cliente alguma eventual incompatibilidade.

Outro papel que pode ser atribuído a um agente de coordenação é o de rotear as informações entre provedores e consumidores de informação. Esta atuação é semelhante à de um coordenador que conhece o tipo de informação necessária para que cada grupo possa funcionar independentemente. Um agente no papel de coordenador pode ser solicitado concorrentemente, e por isso necessita satisfazer os requisitos de possibilitar desenvolver programas simultaneamente, incluindo consistência de dados, gerenciamento de instâncias, filas e prioridades.

3.5. Requisitos de Comunicação

"A capacidade de inovação de uma empresa é determinada pela coesão dos fluxos de comunicação conectados a competências individuais" (Cohen, 1990). Pode-se dizer que a comunicação é decisiva para a inovação. Podemos colocar também que compartilhamento de conhecimento está baseado também em comunicação. Em termos de relacionamentos tipo fonte / consumo, pode ser dito que a fonte deve possuir vontade de compartilhar e o dreno deve confiar na fonte.

Canais de comunicação devem ser definidos de maneira clara, permitindo que seja implementado o fluxo de informação requerido. Isto se reflete no conhecimento que as pessoas possuem sobre quem é o responsável e a quem dirigir-se.

3.5.1 Visão do Usuário

A pesquisa realizada por Moenaert *et al*. (2000) investigou detalhadamente os requisitos de comunicação em equipes dedicadas à inovação em engenharia, em quatro empresas de porte variado, por dois anos e desenhou um quadro de referência teórico o qual é utilizado intensamente neste trabalho. Segundo esta pesquisa, duas condições tornam a comunicação efetiva, considerando-se uma equipe multifuncional e multicultural. A primeira, refere-se à fonte de informação no tocante a intenção de compartilhar informação. Esta intenção pode não ocorrer porque a fonte: a) não é capaz de transmitir a informação, b) não deseja transmitir a informação, c) pensa que a informação não possui valor suficiente para ser transmitida. A segunda implica que a informação possui efeito sobre o receptor da mesma. Este efeito pode estar localizado em três níveis: a) mudança no conhecimento, que é um componente cognitivo, b) mudança na

atitude, que é um componente emocional, ou c) mudança no comportamento público, que é um componente de disposição ou vontade.

Para a comunicação ser eficiente seus efeitos devem ser obtidos com o menor custo possível. A pesquisa identificou três requisitos de efetividade: transparência da rede de comunicação, a codificação do conhecimento e a credibilidade do conhecimento, e dois requisitos de eficiência: custo da comunicação e o sigilo.

Transparência é definida como o grau de clareza e acessibilidade da rede de comunicação de modo que todos compreendam que informações foram recebidas e quais progressos foram realizados. A importância é devida à necessidade de todos os participantes do processo de inovação saberem de modo inequívoco quais são as pessoas relevantes para se transferir ou receber informação. A dificuldade consiste no fato que projetos internacionais envolvendo muitas pessoas têm um processo inicial caracterizado por um alto grau de ambigüidade.

Codificação do conhecimento é definido como o processo no qual o conhecimento e a experiência podem ser estruturados e tornados explícitos. O problema consiste em codificar o conhecimento tácito que está associado à competência de indivíduos e também ao conhecimento associado a equipes (ou grupos ou companhias ou redes de parceiros) que possuem sua própria e codificada linguagem de comunicação (Brown, 1991).

Credibilidade no conhecimento tende a ser um tópico especialmente sensível em projetos de inovação que envolvem a empresa matriz e suas filiais multinacionais. No contexto da pesquisa em análise, foi argumentado que este tipo de problema não afeta a comunidade de pesquisa e desenvolvimento, devido à própria formação destes indivíduos. No sentido oposto, o argumento é que uma equipe de inovação não é constituída somente de cientistas e engenheiros e mesmo este tipo de profissional não é imune à influência étnico-cultural gerada pelo meio a que pertencem.

O Custo da comunicação diz respeito não só ao custo financeiro mas também ao esforço despendido para que pessoas possam se comunicar umas com as outras. A capacidade de inovação de uma empresa é diretamente ligada à coesão do fluxo de comunicação ligando competências individuais. Ressalta-se então que, tanto em termos de custo, tempo e dinheiro, os

canais de comunicação fazem parte das preocupações envolvidas no estabelecimento de estruturas que suportem a comunicação da equipe no processo de inovação.

O sigilo em uma companhia na qual pretende-se compartilhar conhecimento é fator crítico, seja ela uma rede de companhias ou organizações estendidas porque, enquanto deseja-se envolver elementos considerados chave, não se pode tornar público os primeiros passos de um processo de inovação, sob pena dos competidores serem igual e indesejadamente informados.

A solução seria a equipe de pesquisa e desenvolvimento, que tem a tarefa de inovar, ficar toda concentrada em uma área onde a distância não ultrapasse os trinta metros , com todos seus membros falando o mesmo idioma, além de provirem do mesmo estabelecimento de ensino. Contudo, equipes multifuncionais e multinacionais trabalham juntas. A solução real está em prover ambientes virtuais que sejam eficientes e eficazes em auxiliar a equipe nesta tarefa, além de processos de cooperação e coordenação que promovam a integração da equipe. A solução tecnológica isoladamente não é suficiente, mas é fator *sine qua non* na viabilização da integração de equipes de pesquisa e desenvolvimento cujas atividades são desempenhadas em fusos horários distintos.

No projeto DSE (*Distributed Systems Engineering*) (Drira, 2001), empresas e centros de pesquisa de três países formaram um consórcio no qual seus participantes falam seis idiomas diferentes. Apesar dessa diversidade, constataram-se algumas soluções efetivas, entre elas, a utilização de apenas um idioma, que embora não sendo língua-mãe de nenhum dos participantes, era falado por todos; encontros presenciais inicialmente freqüentes, que foram sendo substituídos por encontros à distância – tele-reuniões – e uso de portal para troca de documentos e listas eletrônicas de discussão. Isto foi levado a cabo à medida que a equipe se entrosava e por decisões de projeto, como por exemplo, a utilização de ferramentas de uso corporativo.

Com o objetivo de promover o fluxo de informação entre as pessoas, unidades, cadeia de suprimentos, etc., deve-se conferir a uma pessoa ou a um comitê, o papel de conselheiro. Este comitê, formado por especialistas, independente da estrutura de tomada de decisão da companhia (ou companhias).

Ferramentas computacionais que podem auxiliar as pessoas no processo de comunicação são concebidas de acordo com uma seqüência de atividades e seu mapeamento na

responsabilidade das pessoas. Formalização dessas seqüências induz as pessoas envolvidas no processo de cooperação a trocar informação regularmente. Estas ferramentas devem ser flexíveis o suficiente para permitir descrições de processo na medida que eles são necessários.

3.5.2 Visão do Sistema

Atores que promovem comunicação, segundo a visão de componente de tecnologia de informação, possuem o papel mais de interface que de conselheiro. Esta interface deve assegurar transparência para o usuário, ultrapassando barreiras do sistema operacional, diferenças nas diversas representações de dados ou em conexões via rede. Embora não seja fácil pode ser factível utilizar representação de dados e interfaces de sistemas padronizados. O desafio está em construir ambientes que sejam customizáveis, interoperáveis e independentes de plataformas computacionais.

A formalização deve refletir-se na representação de domínios da área de conhecimento que pretende-se modelar, utilizar uma taxonomia bem definida, mas ao mesmo tempo possibilitar que novos vocabulários sejam acrescentados, aumentando assim a flexibilidade e aplicabilidade do sistema que deseja-se desenvolver.

Em termos de agentes, a comunicação vincula-se ao aspecto social de cada um deles, o protocolo entre eles, de modo que a semântica seja orientada à cooperação, à coordenação, ou a ambas.

3.6. Quadro de Referência dos requisitos para compartilhamento de conhecimento

Os participantes de uma organização envolvidos em um contrato formal ou não de cooperação tecnológica, têm como requisitos para um processo de compartilhamento de conhecimento, segundo os critérios de cooperação, coordenação e comunicação os elementos resumidos na tabela 3.2.

Estes requisitos indicam que os atores envolvidos neste processo são usualmente especialistas em suas áreas, provenientes de diversas áreas funcionais da organização com objetivo de difundir e formar o conhecimento da equipe cooperante. Utilizam habitualmente ferramentas computacionais de trabalho em grupo, como tele-reuniões, sessões de trabalho

compartilhado, correio eletrônico e troca de documentos chamadas de *groupware* e CSCW (*Computer Supported Cooperative Work*).

Tabela 3.4 Visão x Requisitos para Compartilhamento de Conhecimento

		Cooperação	**Coordenação**	**Comunicação**
Participantes	Atores	Tipo: Engenheiros, Projetistas Papel: Especialista	Tipo: Gerentes Papel: *Chairman* Habilidade: Facilitador	Tipo: Equipe com perfil multifuncional Papel: Conselheiros
	Função	Construir conhecimento do grupo	Assegurar o acesso e distribuição da informação	Assegurar o fluxo de informação dentro e entre os diversos grupos
	Ferramen-tas	*Groupware* CSCW	*Workflow*	Navegadores Ferramentas para espaço de trabalho compartilhado Aplicações em rede.
Sistema	Atores	Tipo: Componentes Papel: mediador, assistente	Tipo: Componentes de Programação orientada a agentes Papel: *Matchmaker*	Tipo: Componentes de Programação tempo real Papel: rede de comunicação
	Função	Assegurar interoperabilidade	Assegurar sincronicidade	Assegurar conectividade
	Ferramen-tas	Agentes inteligentes Programação Orientada a Agentes ontologias	Sistemas *middleware* Espaço de trabalho compartilhado	Protocolos: ACL, HTTP Linguagens: XML, HTML, Java

Para sistemas computacionais os requisitos de cooperação se traduzem em componentes que atuam como mediadores para os recursos a serem compartilhados, servindo muitas vezes como assistentes de acesso a esses recursos. A função destes componentes é assegurar que o recurso possua o mesmo significado semântico para seus usuários. Projetos utilizando agentes de software compartilhando uma mesma ontologia fornecem os componentes que asseguram a satisfação destes requisitos.

Para apoiar a cooperação, a coordenação entre os participantes deve ser efetuada por elementos no papel de gerentes com habilidade de facilitadores. Estes participantes têm a função de assegurar o acesso e distribuição da informação necessária à consecução das atividades envolvidas. Utilizam para tal ferramentas de organização do trabalho, tipo *workflow*.

Para sistemas computacionais os requisitos de coordenação se traduzem em componentes que encontrem o recurso correto para a demanda requerida. Estes componentes devem sincronizar a demanda pelo recurso com a disponibilidade deste, obedecendo prioridades e

principalmente identificando produtores e consumidores. Ferramentas como sistemas operacionais distribuídos oferecem a satisfação deste requisito.

O critério de comunicação, para participantes de uma atividade de compartilhamento de recursos, deve ser realizado por elementos que tenham conhecimento das demandas e assegurem o caminho entre produtores e consumidores do recurso. A informação deve ser clara e possuir o mesmo significado entre os participantes. Ferramentas que utilizem o mesmo navegador para as diversas atividades e portais, evitando que os participantes fiquem envolvidos nos protocolos utilizados são as mais apropriadas para este requisito.

Para sistemas computacionais, protocolos de comunicação que assegurem a conectividade entre os diversos aplicativos são os mais adequados. Aplicativos que utilizam protocolos independentes das plataformas de hardware devem ser os empregados para satisfação deste requisito.

3.7. Sistema de Recomendação: Especificação e Detalhamento da Arquitetura

A construção e operação dinâmica de uma rede de habilidades que viabilize a cooperação entre pessoas é um desafio cuja complexidade requer a investigação criteriosa de aspectos relacionados com o compartilhamento de conhecimento e que envolvem a comunicação, coordenação e cooperação. As seções anteriores detalharam os requisitos que uma ferramenta computacional deve satisfazer para auxiliar a gênese dessa rede, respeitando as preferências dos seus participantes ou usuários.

3.7.1. Especificação da Funcionalidade

Esta seção descreve um Sistema de Recomendação como uma ferramenta que se presta a este fim. Recomendar significa indicar ao usuário desse sistema, os nomes das pessoas cujos interesses são iguais ou semelhantes aos seus. O interesse de cada participante expresso em função de suas áreas de conhecimento. Essa recomendação possui como contexto os problemas explicitados na seção anterior, a dispersão dos recursos – pessoas e conhecimentos – nos seus locais de trabalho. Torna-se assim necessário identificar, localizar e interrelacionar os conhecimentos dispersos. E identificar o relacionamento entre conhecimento informal não-estruturado com o conhecimento estruturado. Levou-se em conta também, a necessidade da

utilização simultânea de vários usuários, orientando assim a concepção de um sistema que leve a uma solução funcional e fisicamente distribuída.

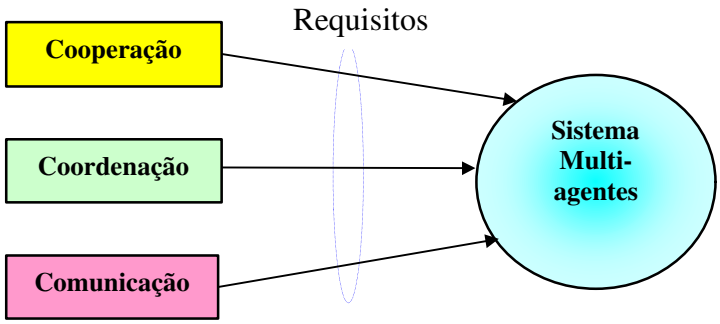


Figura 3.4 Caracterização dos requisitos identificados versus sistema proposto

O sistema descrito a seguir foi idealizado de modo a dispor a seguinte funcionalidade: (a) uma interface homem-máquina que possibilite a cada participante potencial e/ou usuário da rede, descrever seu perfil profissional, consultar sobre outros participantes que possuam interesses semelhantes; (b) realizar dinamicamente a aquisição de conhecimento de cada usuário; (c) sistematicamente atualizar e manter armazenados os conhecimentos capturados dos seus usuários; (d) indicar os nomes de usuários cujos perfis / áreas de interesse assemelhem-se ao do usuário. Esta funcionalidade foi desenhada a partir dos requisitos e considerações acima descritos, o fato de que o compartilhamento de conhecimento requer mecanismos formais de descrição do conhecimento que sejam comuns aos usuários do sistema, tendo-se como fundamento a teoria de sistemas multi-agentes. A Figura 3.4 ilustra graficamente a relação entre os requisitos e a solução proposta.

Um aspecto importante da teoria multi-agentes é seu aspecto de permitir a distribuição, tanto física quanto funcional. A solução multi-agentes possui a funcionalidade necessária, como mecanismos de cooperação entre eles, de modo a satisfazerem um objetivo comum. A "inteligência" do agente está associada a mecanismos de inferência que podem ser utilizadas para solucionar problemas.

Agentes podem ser projetados com mecanismos de coordenação capazes de gerenciar conflitos de recursos e interesses. Os agentes podem atuar em benefício do usuário em tarefas que este julgue apropriadas delegar. Mecanismos de comunicação entre agentes permitem a troca de informação entre eles, eximindo a participação do usuário. Um sistema multi-agentes pode conter

a pró-atividade necessária à realização de tarefas, de tal modo a substituir o usuário em tarefas repetitivas ou exaustivas, como por exemplo, a busca e filtragem de informação na Internet.

A solução empregando sistemas multi-agentes busca permitir conviver com o conhecimento desestruturado e disperso, ao invés de tentar estruturá-lo. Este tipo de solução prima por atender as necessidades do usuário e assim evitar que este tenha de adaptar-se ao sistema instalado.

3.7.2. Arquitetura Funcional do Sistema

A funcionalidade descrita na seção anterior foi organizada tendo em conta que este sistema necessita ser adaptado às diversas áreas de interesse dos usuários. Para isto, empregam-se mecanismos genéricos de representação do conhecimento que possam ser configurados, em função das áreas específicas de interesse dos usuários, como por exemplo, a área de Manufatura. Para facilitar a reutilização de trabalhos anteriores e compartilhamento de conhecimento é interessante incorporar o conceito de ontologias, as quais permitem a formalização dos conhecimentos, facilitando sua manipulação de acordo com os propósitos do sistema.

A arquitetura funcional do sistema é formada por agentes, ontologia e por uma base de conhecimento que se interrelacionam a fim de oferecerem a funcionalidade acima estabelecida. Essa arquitetura é mostrada na figura 3.5; seus elementos estão dispostos em duas camadas funcionais – camada do servidor e camada do usuário – as quais foram assim estratificas, tendo como referência a estrutura de processamento do tipo cliente-servidor. Essa estruturação visa organizar a troca de informações e distribuir as funções dos agentes a partir de suas atribuições, facilitando assim a compreensão e posterior detalhamento da complexidade pelas interações entre esses elementos.

A base de conhecimento deve ser configurada na área específica de interesse do usuário. Dois dos desafios relacionados a especificação dessa base são:

- representar as áreas do conhecimento e ao mesmo tempo possibilitar uma dinâmica associada, ou seja, a base deve ser atualizada de acordo com a mudança ou ampliação dos interesses dos usuários;
- permitir trabalhar com vários idiomas, o que facilitaria a construção de ligações de competências, ultrapassando as barreiras de linguagem.

A representação na Base de Conhecimento deve permitir o crescimento do vocabulário associado às diversas áreas do conhecimento. Assim, as palavras-chave usadas/criadas pelo usuário em seu dia-a-dia no trabalho são elementos que devem estar presentes na base de conhecimento.

Essa base deverá também conter perfis de pessoas com atributos como: área de interesse, área de formação, palavras-chaves da área em que trabalham, projetos em que estão envolvidos, endereços e outros.

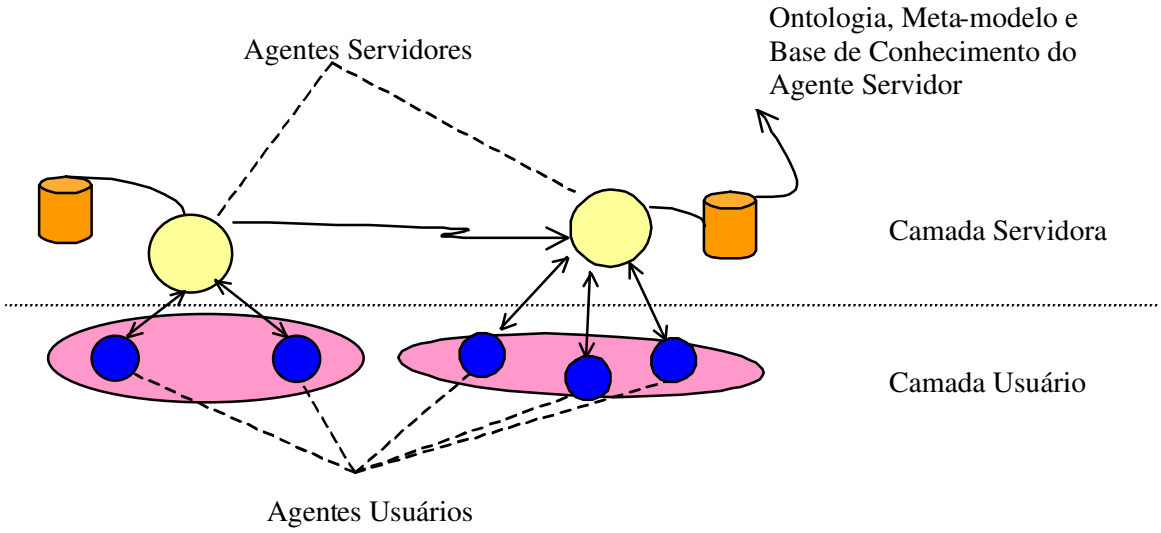


Figura 3.5. Arquitetura funcional do sistema de recomendação proposto

Um sistema multi-agente que esteja parte na camada usuário e parte na camada servidor responde os requisitos de projeto do sistema. O agente da camada de usuário tem características do tipo interface, enquanto o agente da camada servidora tem característica de agente de informação, como ilustrado na Figura 3.5.

No agente da camada de usuário ficará a funcionalidade de aquisição do conhecimento e configuração dos dados do usuário (perfil) e no agente da camada servidora ficará o mecanismo de recomendação, o qual irá interagir com a base de conhecimento, baseado no meta-modelo e ontologia associada, de forma a permitir a identificação de uma rede de outros usuários que preencham os requisitos de compatibilidade para a recomendação de uma rede de habilidades.

3.8. Detalhamento da Arquitetura do Sistema

A construção de grupos deve ser suportada pelo meta-modelo concordando com a noção de construção dinâmica de relacionamentos.

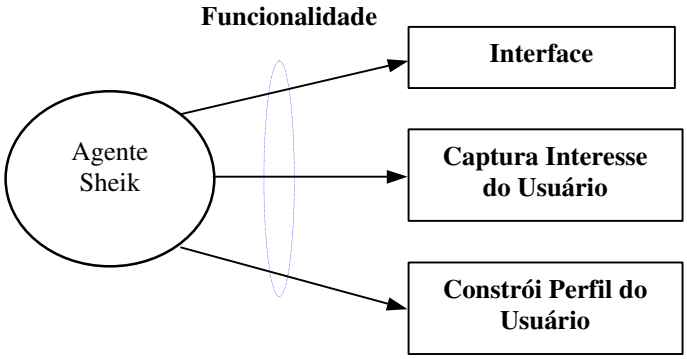


Figura 3.6 Agente *Sheik* e suas funcionalidades

A camada residente no usuário (Figura 3.6) precisa capturar as preferências do usuário, ser leve e não interferir com as tarefas do usuário. Acima de tudo, o usuário não deve ser solicitado a preencher e re-preencher páginas de informação. De acordo com estudos no setor, se não existir um benefício explícito no preenchimento de formulários os usuários não se dispõe a fazê-lo (Stark, 2000). Conseqüentemente, a informação sobre o usuário tem que ser adquirida através do próprio usuário, porém, sem interferir com o seu trabalho. Esta camada deverá adquirir as informações que o usuário necessita. O agente residente no usuário será chamado *Sheik* (*Sharing Engineering Information and Knowledge*) e foi definido primeiramente em (Nabuco-a et al, 2001).

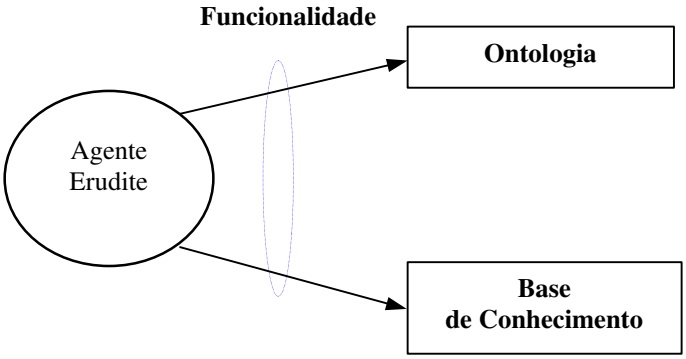


Figura 3.7 Agente *Erudite* e suas funcionalidades

A camada residente no servidor trabalha de acordo com uma demanda comandada pela camada do usuário. É a camada responsável por obter as informações que o usuário necessita e passar para a camada do usuário. Esta camada organiza as informações proveniente da camada usuário atualizando uma base de informação local e também atua como um catálogo telefônico, tanto no sentido informação/pessoa-pessoas quanto pessoa/palavras-chave/área-de-conhecimento, dirigindo a informação a quem dela necessita. A camada residente no servidor, agente *Erudite*, consulta a base de informação local constituída por uma ontologia e suas instâncias, como ilustrado na Figura 3.7. O agente *Erudite* e sua funcionalidade foi proposto primeiramente em (Nabuco-a et al. 2001).

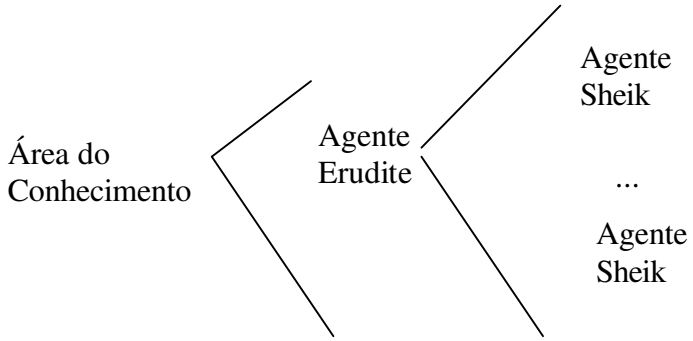


Figura 3.8 Relacionamento entre a área do conhecimento e os agentes

O agente *Sheik* possui várias instâncias, uma para cada usuário, constituindo uma colônia de agentes cujo servidor é um agente *Erudite* que contém seu Domínio de Conhecimento preferencial. O agente *Erudite*, por sua vez, faz parte de uma federação de agentes que possui cada um a ontologia relacionada com um Domínio do Conhecimento, ilustrado na Figura 3.8. Nesta tese, o Domínio do Conhecimento preferencial é ditado pela área de Manufatura, onde os possíveis participantes da rede de habilidades são maior concentração. Porém, a arquitetura não limita o Domínio, podendo existir Domínios de outras áreas do Conhecimento.

O projeto da ontologia, ilustrada na Figura 3.9, contém o meta-modelo de conhecimento constituído pela informação que deve possuir o perfil de um usuário e o Domínio do Conhecimento com o qual o usuário é relacionado. As instâncias deste meta-modelo constituem a base de conhecimento formada pelos usuários, na forma de seus atributos de perfil, contendo área de conhecimento, de atuação, formação e habilidades.

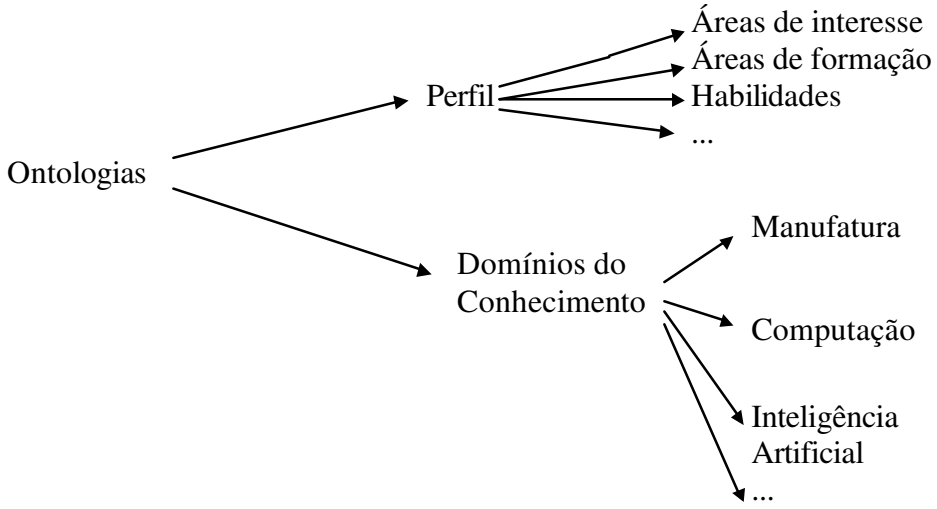


Figura 3.9. Projeto da ontologia residente na camada servidora

A base de conhecimento é capturada através do perfil das pessoas contendo atributos tais como a área de trabalho e os projetos nos quais ela está envolvida. A proposta é que o sistema indique as pessoas que estão na mesma área de conhecimento (Nabuco-b *et al.,* 2001).

A fim de viabilizar essa proposta, emprega-se a programação orientada a agentes que oferece um quadro de referência com os passos a serem seguidos na análise do comportamento dos agentes, como o papel e responsabilidades de cada agente. A definição desta arquitetura foi realizada segundo a metodologia Gaia (Wooldridge *et al.*1999, 2000), resumida no Apêndice 1. Serão então definidos os papéis de cada agente bem como seu comportamento e inter-relacionamento.

3.8.1. Papel do agente *Sheik*

O papel chave deste agente é adquirir e enviar informação. Esse agente implementa a interface homem-máquina especificada na seção 3.6, por intermédio da qual o usuário pode, opcionalmente, preencher seus dados, uma máquina de inferência e um mecanismo que monitora os arquivos acessados pelo usuário – um observador. Estes diferentes componentes atuam de modo independente e inter-relacionam-se tendo como objetivo comum formular perguntas, receber respostas e apresentar esta informação ao usuário. Estas características são modeladas como mostrado na Tabela 3.3.

Protocolo – Engloba as principais funções do agente; a saber:

- *GetProfile*: adquire palavras-chave que trazem a melhor descrição do trabalho do usuário, sua especialidade, ferramentas que utiliza, que compreendem as habilidades do usuário.
- *RegistrationReq:*Após adquirir as preferências do usuário, o agente Sheik envia um pedido de registro ao agente Erudite que ele conhece.
- *SendQueryReq/Rsp*: Constrói e envia uma questão para o agente Erudite, de acordo com a sua sensibilidade para a necessidade do usuário. Para identificar os requisitos do usuário o agente Sheik usa um algoritmo de reconhecimento de palavras-chave (naïve Bayesian network) (Frank, 1999) e observa as atividades do usuário comparando as atualizações dos arquivos textos utilizando o marcador de tempo da máquina do usuário.
- *SendEvent*: O agente se comunica com o usuário enviando uma mensagem para a tela do usuário. Quando o usuário quiser, ele pode acionar a interface que conterá a informação buscada pelo agente.

Tabela 3.3 Esquema para o papel do agente *Sheik*

<table>
<tr><td colspan="3">Papel Esquema: SubmitQuery</td></tr>
<tr><td colspan="3">Descrição: Constrói questões de acordo com o perfil do usuário e envia esta informação ao seu usuário.</td></tr>
<tr><td colspan="2">Protocolos</td><td>RegistrationReq, GetProfile, SendQueryReq, SendEmail, GetEvents.</td></tr>
<tr><td rowspan="3">Permissões</td><td>lê</td><td>Eventos // quando o usuário inicia uma ação
perfil// informação do usuário</td></tr>
<tr><td>adiciona</td><td>Lista de palavras-chave// lista descrevendo área de trabalho do usuário e necessidades.</td></tr>
<tr><td>gera</td><td>Questão (query)</td></tr>
<tr><td colspan="3">Responsabilidades
Vivacidade:
SubmitQuery = (GetProfile.(GetEvents. SendQuery. SendEmail)) $^{\omega}$
Segurança:
Enabled = TRUE; // detecta a disposição do usuário
Events = SELECT {action1, …, action} //começa a trabalhar em eventos específicos</td></tr>
</table>

Permissões: Estão relacionadas com a informação que o agente necessita para realizar sua tarefa.

- *Leitura*: Acesso a eventos (ações realizadas pelo usuário) de modo a realizar suas tarefas e também ler a informação fornecida pelo usuário para construir a informação mais adequada.

– Adiciona: o agente irá atualizar o perfil do usuário adicionando informações do tipo palavras-chave à lista existente.

– Geração: o agente observa o ambiente e no tempo adequado constrói a questão adequada.

Responsabilidade - Descreve as propriedades gerais do agente (Vivacidade) levando em consideração seu protocolo e condições de trabalho (Segurança).

3.8.2. Papel do agente *Erudite*

A função do agente *Erudite* é complementar ao agente da camada de usuário, no sentido que este agente procura a informação que o agente do usuário requisitou. Ele possui o endereço de todos os agentes do tipo *Sheik* sob sua jurisdição e de outros agentes na mesma camada de aplicação e suas áreas principais de atuação. Todos esses agentes possuem um meta-modelo que é comum a todos eles e áreas de conhecimento especializadas.

Tabela 3.4. Esquema para o agente *Erudite*

<table>
<tr><td colspan="3">Esquema Papel: MatchQuery</td></tr>
<tr><td colspan="3">Descrição: Responde as questões ou as redireciona, se não pertence ao seu domínio.</td></tr>
<tr><td>Protocolos</td><td colspan="2">EnrollQuery, SendQueryRsp, RedirectReq, RedirectRsp</td></tr>
<tr><td rowspan="2">Permissões</td><td>Lê</td><td>Lista do Agente</td></tr>
<tr><td>Gera</td><td>Respostas
Redireção</td></tr>
<tr><td colspan="3">Responsabilidades
Vivacidade:
MatchQuery = EnrollQuery. (SendQueryRsp |
RedirectReq | RedirectRsp)$^{\omega}$

Segurança:
TimeToResolve < Slow // Parâmetro dependente dos recursos. Evita iteração sem fim.</td></tr>
</table>

O agente *Erudite* pode receber questões a qualquer momento mesmo as que não estão diretamente relacionadas com sua área principal de atuação. Essa área pode ser expandida requerendo do agente gerenciamento dos recursos de memória para não exceder o limite do sistema. Os limites do agente estão relacionados com recursos disponíveis na plataforma computacional onde reside. Os protocolos relacionados com estes agentes são:

– *EnrollQuery*: insere numa lista as questões que estão aguardando resposta.

– *SendQueryRsp*: esta primitiva contém a informação pedida por um agente *Sheik* determinado.

– *RedirectReq*: esta primitiva consiste em uma questão redirecionada por um agente *Erudite* para outro. Ocorre quando a questão está fora da área de conhecimento do agente. O agente envia a questão para os agentes que ele encontrou similaridade de acordo com as "páginas amarelas" que ele possui disponível. O esquema para o agente *Erudite* está resumido na tabela 3.4.

3.8.3. Modelo de Entendimento

Este modelo define os caminhos de comunicação do agente. Neste trabalho a ligação entre os agentes *Sheik* e o agente *Erudite*, o usuário e seu agente *Sheik*, e as instâncias do agente *Sheik* e instâncias do agente *Erudite*. A distinção entre os agentes servidores *Erudite* é feita pela área do conhecimento. A Figura 3.5 descreve os caminhos possíveis entre os agentes.

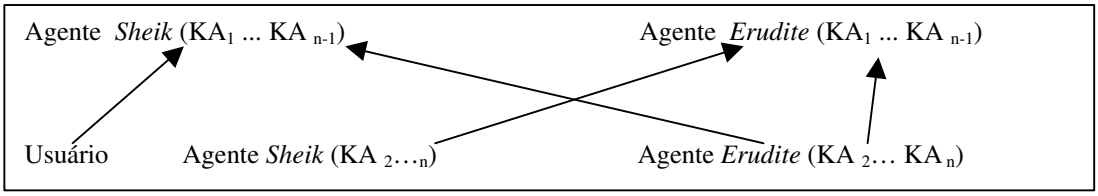


Figura 3.5. Modelo de entendimento para os agentes

3.9. Conclusão

Este capítulo apresentou os requisitos e a arquitetura do sistema. Foram apresentados os requisitos funcionais do sistema segundo o ponto de visão do usuário e do sistema que conduziu no sentido de uma solução multi-agente.

A teoria de multi-agentes foi projetada para:

– compartilhamento de conhecimento e diversos mecanismos de comunicação e coordenação foram desenvolvidos para que eles pudessem cumprir este papel. No caso desta tese, as ontologias cumprem este papel, de fornecerem um vocabulário formal e organizado.

– Aquisição de conhecimento onde o agente pode fazer uso de uma biblioteca de métodos advindos da inteligência artificial ou utilizar uma interface humano-máquina ou utilizar sensores ópticos, elétricos, mecânicos, etc, para poder inferir sobre o ambiente que o cerca. No caso desta tese, uma máquina de inferência baseada em redes Bayesianas foi utilizada para

capturar uma amostra do conhecimento do usuário, conhecimento não-estruturado, através da identificação de palavras chaves contidas nos documentos disponibilizados no computador do usuário.

– Ser um sistema fisicamente distribuído possuindo mecanismos de comunicação como protocolos especialmente definidos para comunicação entre agentes. E funcionalmente distribuído pois cada agente pode ser o executor de uma tarefa específica. São variadas as técnicas de distribuição funcional dos agentes. No caso da tese são dois tipos de agente com funcionalidade de interface e outro de servidor de informações. Os agentes estão residentes nos usuários e nos diversos servidores de informação, usualmente fisicamente dispersos. Os agentes residentes no computador do usuário formam uma colônia para cada agente servidor, segundo a área de conhecimento do usuário. Os agentes servidores formam uma federação segundo os Domínios do Conhecimento selecionados.

Para a comunidade de agentes que está se construindo a cooperação é essencial para garantir o resultado da operação. A capacidade para resolver os problemas propostos está distribuída entre os vários agentes . De modo que sem o agente servidor não se tem acesso às instâncias de conhecimento descritas pela base de conhecimento. Porém ,o agente usuário é que alimenta a base, entregando a informação para o agente servidor.

Para ser efetiva, a base de conhecimento deve incorporar um vocabulário comum. Além do que a fonte de conhecimento deve ter credibilidade junto às pessoas a que ela se destina. No presente momento, é comum e mesmo crescente que uma companhia ou uma cadeia de fornecedores seja formada por pessoas (ou companhias) de diferentes países, falando idiomas diversos. Essas diferenças são alguns dos obstáculos para tornar o compartilhamento de conhecimento possível. No sentido de resolver estes problemas as bases de conhecimento devem ser construídas com o uso de ontologias e dicionários que facilitem a busca de informação (Farquhar, 1996; Gruber, 1993).

A arquitetura do sistema foi desenhado de acordo com a proposta de Wooldridge *et al.* (1999a e 2000b), tendo em vista um sistema distribuído no qual cada usuário dispõe de seu próprio agente bem como as áreas de conhecimento.

Durante o projeto do sistema foram tomadas as seguintes decisões:

- empregar sistemas multi-agentes como solução para cooperação em sistemas distribuídos;
- definir duas camadas de agentes como forma de distribuição funcional;
- definição e reutilização de ontologias para prover o sistema com informações compartilháveis;
- realizar aquisição de conhecimento através de palavras-chave usando redes Bayesianas;
- empregar a linguagem Java para que a codificação fosse independente de plataforma;
- utilizar uma ferramenta de descrição de projeto automatizada;
- ferramentas para desenvolvimento de uso livre e gratuito de modo a garantir a repetibilidade da solução;

O próximo capítulo apresenta o projeto detalhado do sistema multi-agente utilizando a ferramenta *Rational Rose* (www.rational.com).

Capítulo 4

Descrição do projeto do Sistema Multi-agente

Este capítulo apresenta a descrição do projeto do sistema multi-agentes. Como foi visto no capítulo anterior, a arquitetura do sistema multi-agentes prevê um modelo de duas camadas sendo uma camada servidora protagonizada pelo agente *Erudite*, contendo o modelo de informação do sistema, e outra camada, a do usuário, protagonizada pelo agente *Sheik* que faz aquisição das informações referentes ao usuário, no que diz respeito a sua área de trabalho, área de formação e habilidades. Será apresentada uma especificação detalhada do sistema multi-agente que pretende cumprir este papel.

4.1 Introdução

A especificação inicial do sistema de agentes foi realizada utilizando-se a metodologia Gaia de Wooldridge *et al*. (2000). Na seqüência, o detalhamento do sistema se faz necessário. Para realizar esta etapa, utilizou-se a ferramenta *Rational Rose* (www.rational.com), e o UML (Booch, 1994). A ferramenta *Rose* oferece um ambiente onde várias vistas são desenvolvidas e foi utilizada até a geração de protótipos para a programação na linguagem Java (www.java.sun.com), utilizando-se a ferramenta Prototipador de Agentes (Koyama, 2001).

O *Rose* é utilizado para documentar o sistema, o que possibilita também que outros desenvolvedores possam entender a especificação e mesmo verificar problemas como falta de ligação entre módulos. A utilização de uma ferramenta de caráter mais geral possibilita que a implementação possa ser realizada em qualquer linguagem orientada a objetos. Na seqüência, o projeto detalhado do sistema com as classes, o protocolo e a dinâmica dos componentes dos agentes.

4.2. Componentes do Sistema

Um sistema é um conjunto de partes coordenadas realizando determinadas funções (Schoderbek *et. al*, 1980). Olhando a Figura 4.1 podemos ver os componentes principais do sistema. Existem duas camadas de agentes, os pessoais e os servidores. Nessas camadas fica residente, no lado do usuário, um algoritmo de classificação e aprendizado, enquanto no servidor fica localizada a ontologia. A ontologia traz os componentes que modelam o vocabulário e a área do conhecimento além do perfil do usuário.

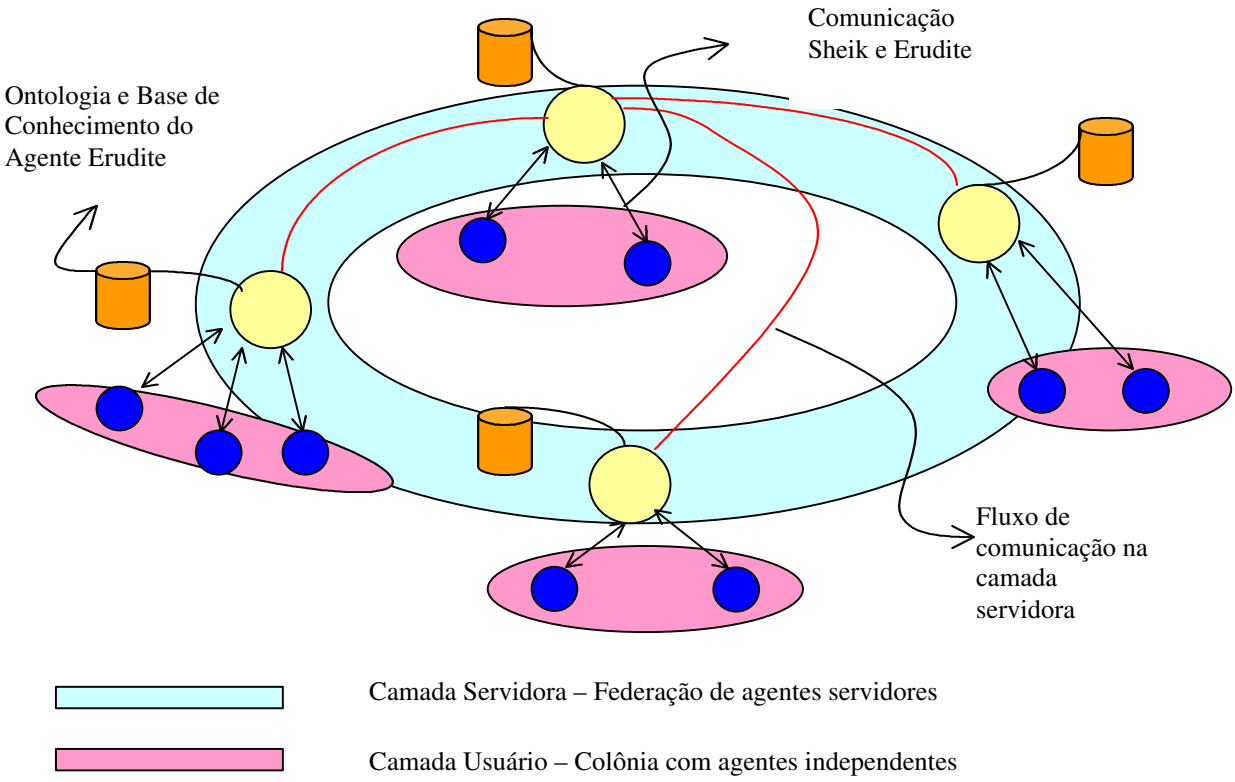


Figura 4.1 Arquitetura do Sistema Multi-agente

A comunicação entre os agentes *Erudite* é assíncrona e esporádica, correspondendo ao sistema de mensagens de redirecionamento (ida e volta) definido no capítulo anterior (*RedirectReq*). O ajuste entre agente *Sheik* e agente *Erudite* ocorre por afinidade e não por localização geográfica. É possível a troca de agente *Erudite* em operação devido à mudança do perfil do usuário. Um usuário pode estar na base de conhecimento de mais de um agente *Erudite*.

A comunicação entre agente *Sheik* e agente *Erudite* acontece por iniciativa do agente *Sheik* e não está prevista comunicação entre agentes *Sheik*. O objetivo da estratificação da comunicação é a coordenação do fluxo de informação, de modo que exista um elemento concentrador da informação, ao invés de uma comunicação ponto a ponto entre todos os agentes Ou usar um sistema de recomendação baseado em recomendação indireta, onde a informação fornecida por elementos acreditados passa a ser a recomendação do sistema. A concentração de conhecimento no *Erudite* é decisão de projeto, baseado na leveza que se quer obter no agente *Sheik*.

A Figura 4.2 traz os componentes do sistema em termos de diagrama UML. O módulo Framework representa a base do sistema e a maneira como os outros módulos se relacionam. O módulo Perfil (Profile) contém as classes que representam o usuário dentro do sistema. E os módulos representantes do agente Sheik e do agente Erudite.

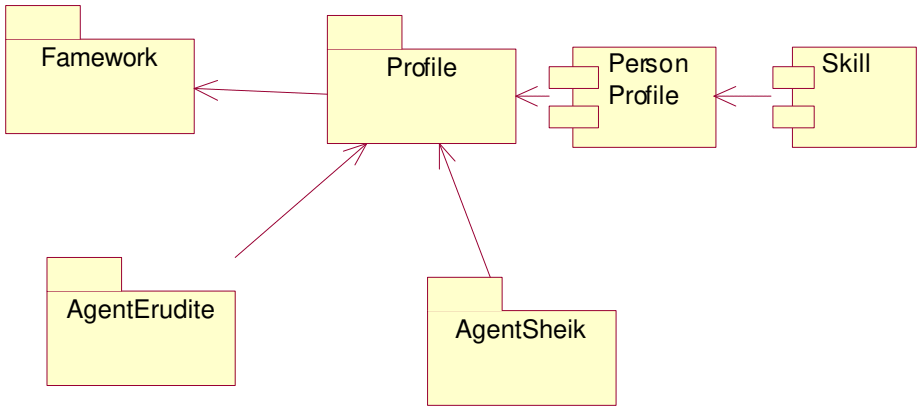


Figura 4.2 Módulos do sistema e seus relacionamentos (Diagrma de *Pakages* UML)

4.2.1. Modelo do Agente Sheik

O módulo *AgentSheik* descreve o comportamento do agente, sua dinâmica de interação com o usuário e com o *Erudite*, além dos atributos, protocolos e máquinas de estado. A Figura 4.3 mostra a interação entre o usuário e o agente *Sheik* e as mensagens trocadas entre os dois.

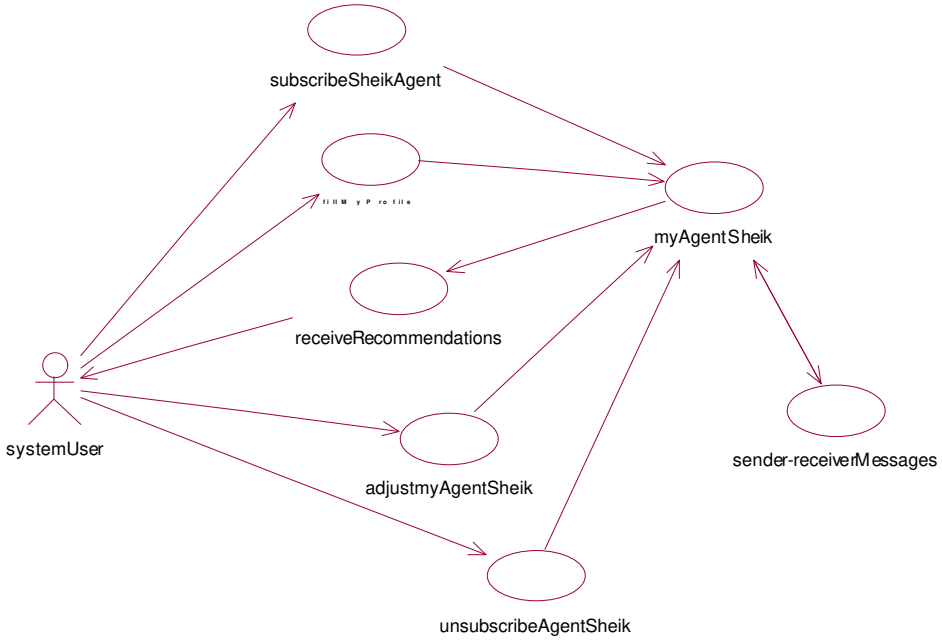


Figura 4.3 Interação entre o usuário e o agente *Sheik* (Diagrama de Utilização UML)

A especificação foi realizada em UML sendo que as figuras são diagramas UML realizados no *Rational Rose*. A seqüência de cima para baixo é intencional e é relacionada com a interação no tempo entre ambos. O usuário pode, a qualquer momento, verificar seu perfil e mesmo alterá-lo através do serviço de ajuste (*adjustmyAgentSheik*). O único fluxo obrigatório é o de assinatura do serviço (*subscribe*) sem o qual o agente não é ativado.

A Figura 4.4 mostra as classes pertencentes ao agente *Sheik*. O agente possui três classes que na prática são tarefas que funcionam independente e concorrentemente. Possui uma classe dita Gerenciador (SheikManager) que recebe e envia as mensagens para o *Erudite* via interface de comunicação (*SheikCommInterface*) e recebe e envia informações do usuário via interface humano-máquina (*Sheik H/M-Interface*).

A tabela 4.1 descreve as classes e seus serviços e conta com um pequeno comentário adicional relatando sua funcionalidade.

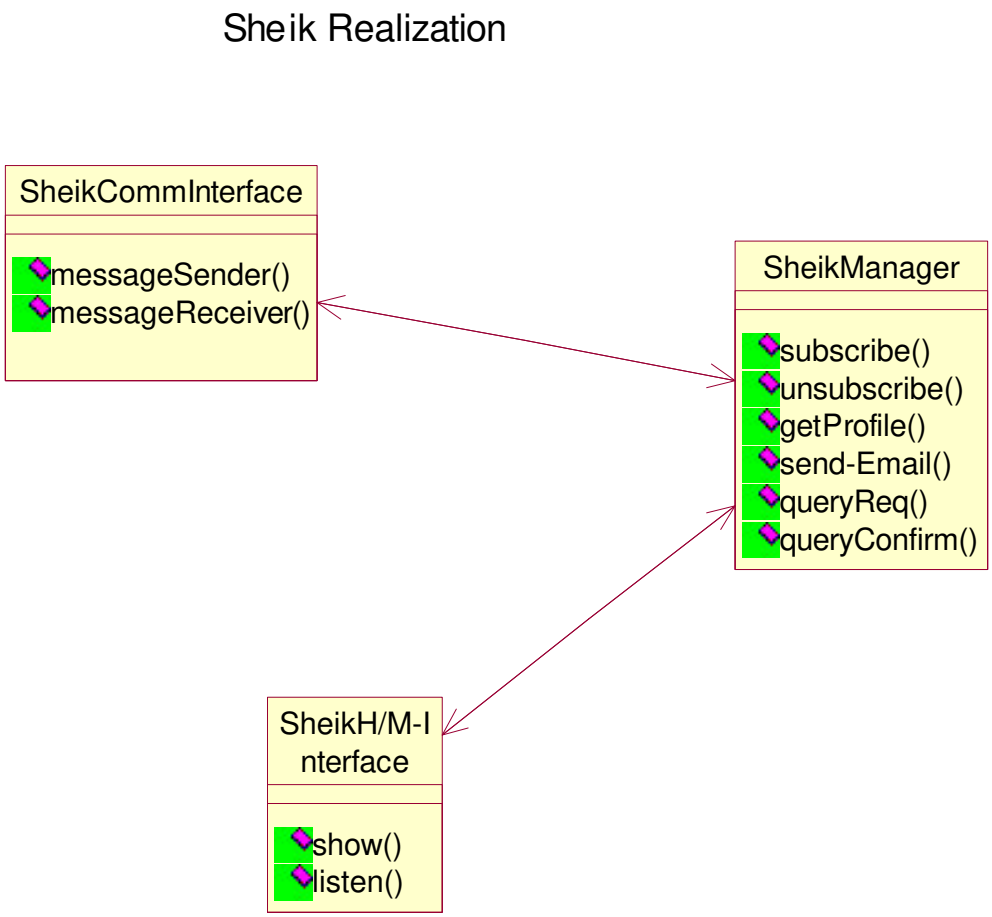


Figura 4.4. Classes do agente Sheik (Diagrama Lógico UML)

Não existe sincronismo entre o envio e o recebimento de mensagens do tipo questão (*query*).

As mensagens são enviadas com um identificador que é devolvido quando da resposta possibilitando a identificação.

O agente pode fazer quantas perguntas queira sendo o limite ditado pelo uso de recursos.

O uso de uma regulagem de tempo máximo faz com que o agente não reserve recursos para questões que 'demoram' a retornar (parâmetro configurável).

Tabela 4.1 Tabela de documentação da classe *AgentSheik*

Classe *SheikH/M-Interface* {*Analysis*} derivada de: *SheikManager*			
Documentação			
Interface com o usuário onde serão indicadas a informação de perfil e também a boa vontade do usuário em participar da Rede de Habilidades.			
Pai	AgentSheik	Abstrato	No

Operações

Nome	Assinatura	Classe	Comentário
show	*show ()*	*SheikH/M-Interface*	Mostra o formulário com o perfil do usuário
listen	*listen ()*	*SheikH/M-Interface*	Aguarda o evento que acorda o módulo
subscribe	*subscribe ()*	*SheikManager*	Realiza a operação de assinatura do serviço
unsubscribe	*unsubscribe ()*	*SheikManager*	Se desliga do sistema e "mata" a tarefa
getProfile	*getProfile ()*	*SheikManager*	Aciona o algoritmo de captura de informação
send-Email	*send-Email ()*	*SheikManager*	Alerta o usuário sobre a informação recebida
queryReq	*queryReq ()*	*SheikManager*	Envia uma questão ao *Erudite*
queryConfirm	*queryConfirm ()*	*SheikManager*	Chegada da resposta à questão

A Figura 4.5 mostra a máquina de estado do agente *Sheik*. A máquina foi utilizada pelo programa Prototipador de Agentes (Koyama, 2001) para configurar os agentes utilizados. A máquina de prototipação utilizada será detalhada no item 4.3.

Quando no estado *Activated* o agente possui uma série de atividades que são realizadas conforme a necessidade ou em seqüência, como por exemplo coletar as frases-chave, construir uma questão (*query*), guardar as informações referentes ao perfil do usuário e estar alerta para os movimentos do usuário. Ainda neste estado o agente aguarda a chegada de mensagens e também realiza a interface com o usuário. Estas atividades são realizadas por tarefas independentes. As

informações são atualizadas em armazenadores (*buffers*) que as tarefas consultam no decorrer de sua execução.

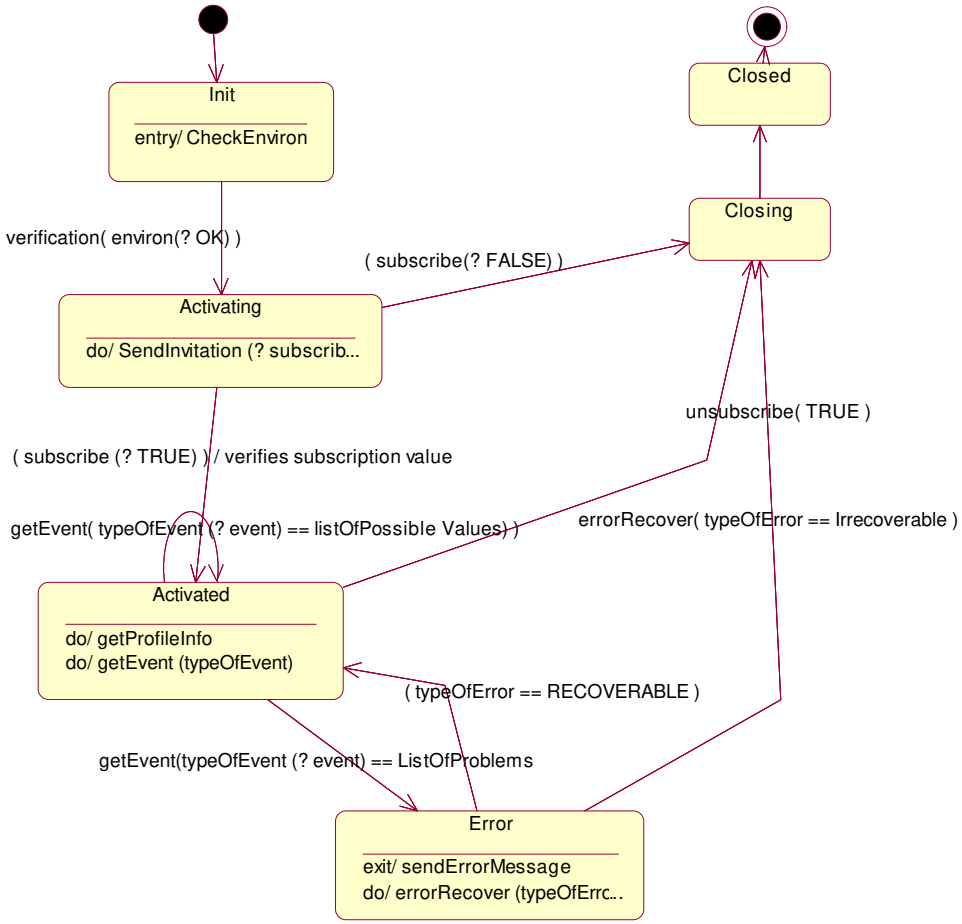


Figura 4.5. Máquina de Estado do agente *Sheik* (Diagrama de estados UML)

Tabela 4.2 Máquina de Estado *State/Activity*

Activities	getProfile	show	messageReceiver
getKeyPhrase	buildQuery	storeProfile	watcherManager
Sender			
Partitions	SheikManager	SheikCommInterface	SheikH/M-Interface

4.2.2. Modelo do agente *Erudite*

O módulo *Erudite* descreve a interação deste agente com outros agentes do mesmo tipo e com o agente *Sheik*. A interação se dá também na busca por informação, quer utilizando a base de conhecimento local, quer buscando a informação em outros servidores ou outros aplicativos. O *Erudite* não possui interface com o usuário. É ele quem gerencia sua fila de atendimento além do acesso à base de conhecimento.

A Figura 4.6 mostra, através de um diagrama de utilização UML do componente, o agente e seu comportamento. O administrador de sistema realiza a configuração inicial, já que como dito anteriormente, o agente não interage com o usuário final. Como a base de conhecimento pode ser acessada sem o uso do agente, a interação entre possíveis usuários ou administradores com a base é indicada no esquema. O agente recebe as mensagens, guarda seus identificadores e tenta resolver a questão. Caso não seja possível, a mensagem é redirecionada para outro agente, o mais próximo em termos de área do conhecimento.

Como a fila é gerenciada, como é o acesso à base de conhecimento e como é a base de conhecimento, serão vistos no próximo capítulo, que é dedicado à base de conhecimento. A federação de agentes pode possuir correlação mas os agentes só conhecem seus pares pelo domínio.

A Figura 4.7 mostra as classes que implementam o agente. Estas classes implementam tarefas que se comunicam trocando mensagens. São três as classes:

i) Interface de Comunicação (*commInterface*): interface de comunicação que lida com o tráfego de mensagens. Ele identifica a mensagem que chega e também constrói a mensagem a ser enviada. Ele escolhe a mensagem que será enviada de acordo com sua fila e o tipo de protocolo que será utilizado. O prototipador sobre o qual o agente foi construído está encarregado de montar a mensagem a ser enviada para o outro agente, sendo responsável pela comunicação entre plataformas.

ii) Interface Aplicativo: Esta classe lida com os outros aplicativos que o agente interage. Ela chama os métodos de leitura/escrita do construtor de ontologias Protégé. Ele também chama, através de interface adequada (uma chamada de função que por sua vez invoca um aplicativo remoto, no caso uma página da internet), os aplicativos WordNet

(http://www.cogsci.princeton.edu/~wn), que é um analisador léxico, em inglês, e o www.dmoz.org que é utilizado para se encontrar o domínio do conhecimento da palavra. O agente possui uma operação (que é uma chamada de função) dita "*callapp*" que procura o recurso adequado na operação de busca de um semelhante para a palavra-chave ou frase-chave recebidas do agente *Sheik*. Outra funcionalidade associada à interface é o cálculo de distância entre a palavra-chave e o Domínio.

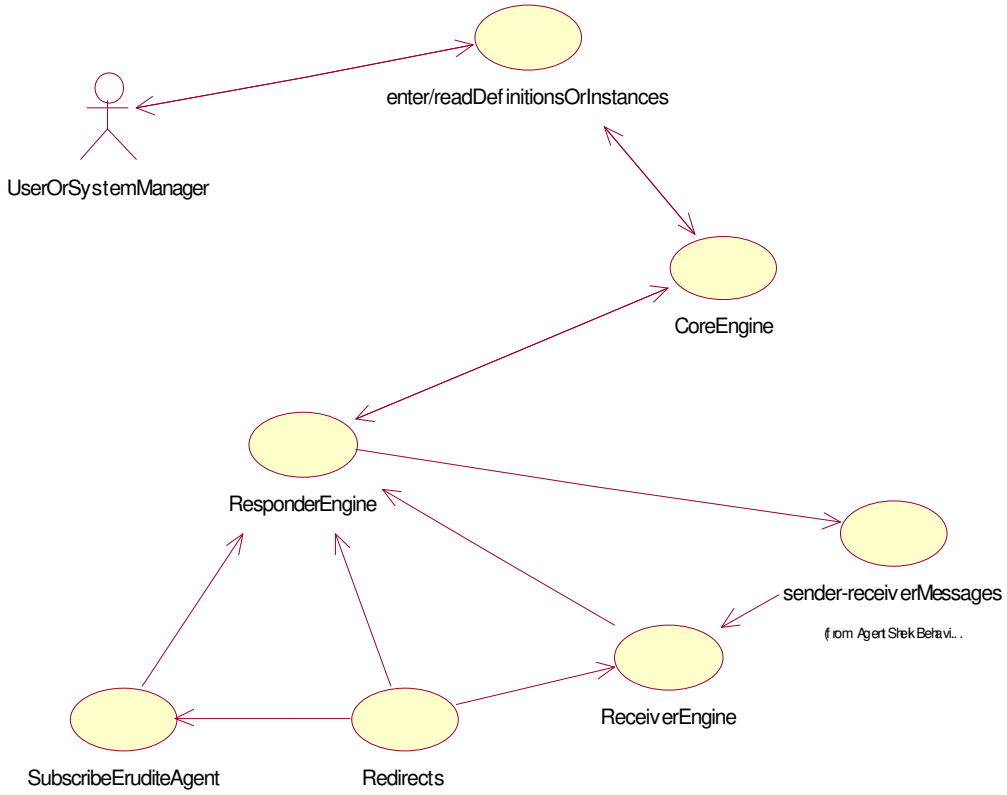


Figura 4.6 Comportamento do agente *Erudite* (Diagrama de Utilização UML)

iii) Gerenciador (*EruditeManager*): responsável por coordenar a chamada das outras tarefas. Lida com o tratamento de erro e envia mensagens para outras tarefas.

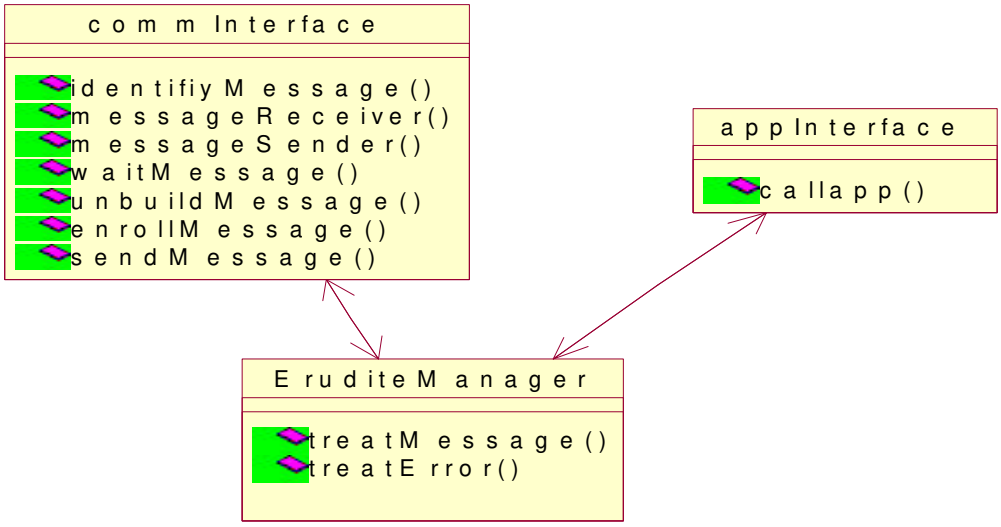


Figura 4.7 As Classes do Agente *Erudite* (Diagrama Lógico UML)

4.3. Módulo Prototipador de Agentes

Os agentes *Sheik* e *Erudite* foram implementados utilizando-se a ferramenta computacional Prototipador de Agentes. Este módulo é fundamental para o sistema pois o implementa em termos de código, além de permitir o teste do sistema, depurando-o. Ele consiste no resultado da tese de doutorado de Koyama (Koyama, 2000) e foi refinado para poder ser utilizado, entre outros, no presente trabalho.

O módulo consiste de uma plataforma de prototipação de componentes, composta de interface, onde o usuário pode configurar seu componente e ferramentas de depuração. Juntamente com o módulo pode ser utilizado um construtor de sistemas em linguagem *Java* como o *Jbuilder* ou *Forte* (www.borland.com ou www.java.sun.com). As versões utilizadas são de acesso livre, podendo o ambiente ser reproduzido em outras situações.

Este módulo permite que se projete os agentes, utilizando-se para isso de máquinas de estado. Ele entrega as tarefas prototipadas em forma de código na linguagem *Java*. A metodologia utilizada consiste da convergência de técnicas de especificação de sistemas distribuídos de tempo real, os diagramas DFD (Diagrama de Fluxo de Dados)(Ward, 1985), com a metodologia e linguagem UML para Sistemas Orientados a Objetos (SOO).

Considera-se que os diagramas DFD, sendo orientados a processos, são adequados para especificar a relação dos agentes com o seu ambiente e entre os próprios agentes, permitindo esboçar as regras de coordenação para o Sistema Multi-Agente.

A linguagem UML é utilizada de maneira complementar. Cabe ressaltar que UML é adequada para a descrição de sistemas orientado a objetos, não contendo diretamente nenhum suporte à distribuição destes objetos no nível de especificação funcional (eventualmente suporta algum tipo de empacotamento ou componentização). As ferramentas UML existentes não geram diretamente código para sistemas distribuídos, validando esta afirmação.

4.3.1. Descrição da Metodologia

- Diagrama de Contexto

Tanto o Projeto Estruturado de Sistemas de Tempo Real, quanto a metodologia UML, partem de uma definição da relação do sistema com seu ambiente, chamado Diagrama de Contexto, a diferença maior entre eles está na simbologia adotada.

- Decomposição / Distribuição

Logo a seguir UML preconiza a utilização de Diagramas de Utilização (Use-Cases) os quais decompõem o sistema em vistas de comportamento. O sistema é testado de modo a responder às situações que ele estaria sujeito. Estes Diagramas de Utilização podem eventualmente estar subordinados a relações semelhantes às encontradas em classes de objetos, como por exemplo "estende" e "usa".

Estes diagramas, quando se pensa em um Sistema Multi-Agentes (SMA) e não em um Sistema Orientado a Objetos (SOO), não mostra um aspecto importante dos SMAs que é a coordenação entre os agentes, de vez que uma vista de comportamento pode significar a interação entre diversos agentes, a qual não é representada no diagrama, tampouco em outro diagrama UML, no nível de especificação (UML mostra a interação entre classes de objetos, mas agentes não são objetos).

- Visão estática

A metodologia proposta utiliza a técnica do PESTR para a decomposição do problema em um conjunto de agentes que se comunicam através de mensagens, sendo o fluxo de mensagens

claramente visualizado no diagrama da Figura 4.8. Definem-se então os agentes e as mensagens trocadas entre eles.

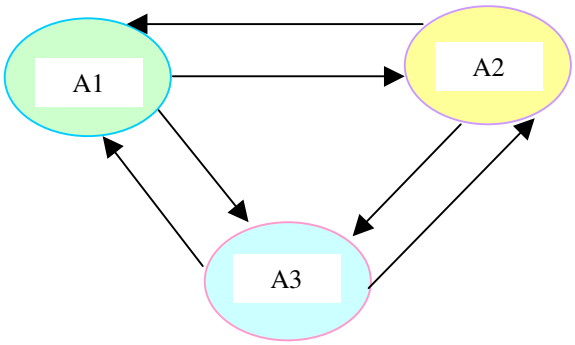


Figura 4.8 Sistema SMA e a comunicação entre os agentes

- Visão Dinâmica

Para explicitar o comportamento do sistema a ser prototipado, é necessário o uso de diagramas de comportamento baseados em máquinas de estado. Como UML utiliza Diagramas de Estado (StateCharts) para a descrição de comportamento, estes diagramas podem ser utilizados, já que são compatíveis e conversíveis em máquinas de estado.

Sendo uma ferramenta para detalhamento do sistema, as máquinas de estado são tantas máquinas quantos sejam os agentes existentes. Estas máquinas interagem entre si para fornecer o comportamento desejado ao sistema, como mostrado na Figura 4.9.

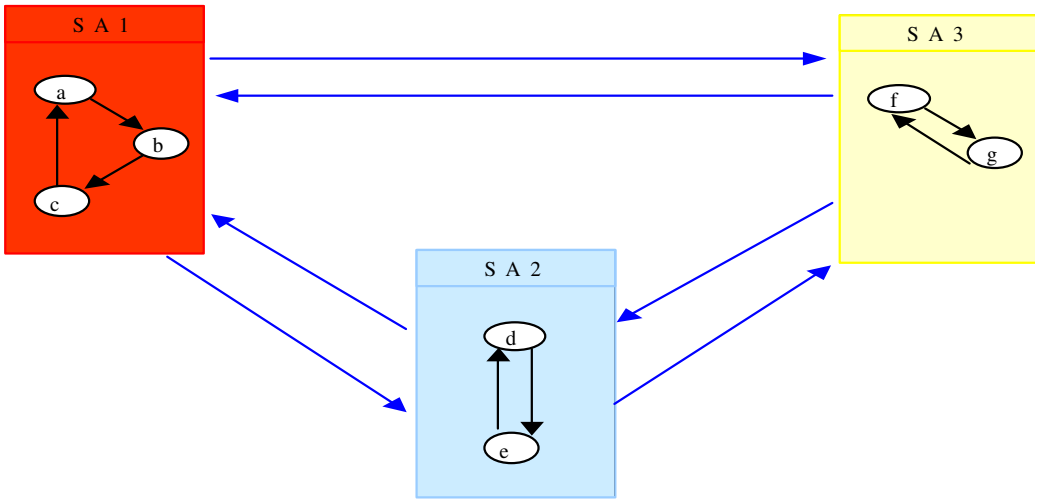


Figura 4.9 Decomposição em SMA

- Determinação iterativa do Comportamento

Normalmente não é possível definir diretamente as máquinas de estado para um SMA, devido à complexidade das relações entre os agentes e de suas funções internas (também ativadas pela máquina de estado).

Deve-se, portanto, partir para uma determinação iterativa deste comportamento através da composição dos vários atos que caracterizam o comportamento global do agente.

Em outras palavras: o comportamento global de um agente pode ser caracterizado através da combinação de seus comportamentos parciais obtidos através do estímulo do agente com cada um dos eventos acessíveis em cada um dos estados possíveis (projetados) do agente.

A maneira de se obter estes comportamentos parciais é através do uso de diagramas de seqüência UML, os quais permitem representar cenários de evolução do sistema. A modificação necessária em relação a UML consiste em nomear as entidades do diagrama de seqüência de acordo com os agentes (e não classes como é feito em UML) e de incluir nos diagramas indicações dos estados pelos quais cada agente transita. Após um certo número de tentativas (e de cenários) pode-se determinar as máquinas por inspeção dos diagramas.

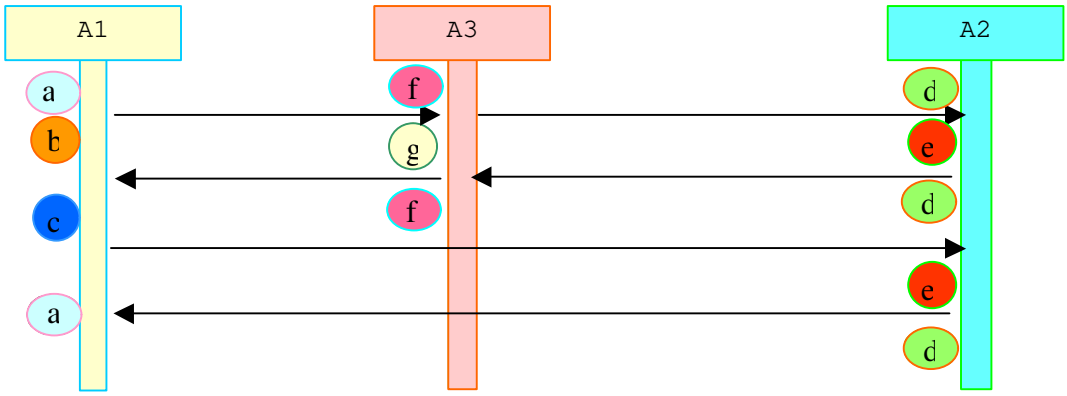


Figura 4.10 Cenário C1

São determinados, como exemplifica a Figura 4.10 n cenários { C1, ... , Cn}, para cada cenário as transições de estado que ocorrem {SC1,... , SCn}. As máquinas de estado de cada agente SA1, SA2 e SA3 são deduzidas por inspeção / sobreposição das máquinas SCx, como ilustrado na Figura 4.11.

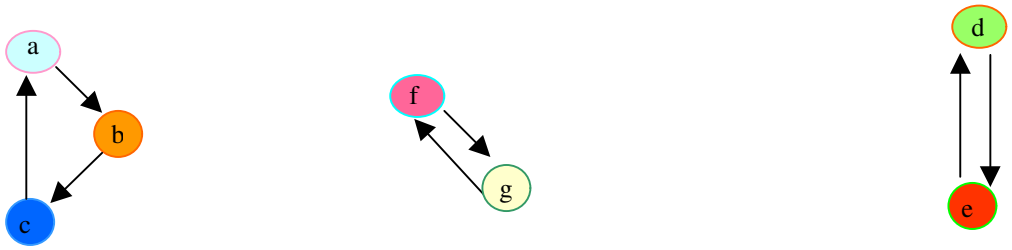


Figura 4.11 Dedução das máquinas de estado

Para cada tarefa que compõe o agente uma máquina de estado é associada. Máquina esta deduzida a partir das mensagens trocadas entre os agentes (componentes deduzidos para formar o núcleo básico de agentes). As máquinas que vão sendo deduzidas vão compondo a máquina de estado do componente, que por sua vez é um "aspirante" a agente.

Estas diversas tarefas são sincronizadas e chamadas por um elemento dito coordenador, que possui esta atribuição de ativar um agente, reiniciar, eliminar, realizar a comunicação entre agentes e tratamento de erro.

Com este conjunto, o agente é configurado no sistema de desenvolvimento que é o Prototipador (o sistema foi utilizado no sistema Syroco (Roy, 1998) de aplicação em manufatura). Um agente núcleo está previamente configurado permitindo que atualizações sejam feitas de modo a obedecer as deduções realizadas sobre o novo sistema a ser implementado.

4.3.2. Vista de Implementação

Uma vez definidas as mensagens e o comportamento associado a cada mensagem fica caracterizado o protocolo. O agente prototipado é constituído pelo núcleo e pelos protocolos. A máquina de estado deve ser descrita segundo Ward e contém:

- Pré-condições – condições definidas à priori para verificação de viabilidade de execução.

- Ações – o que será realizado

- Pós-condições – antes de efetuar a transição para um próximo estado podem ser avaliadas condições que não estão contidas na ação porém contidas no sistema em geral. Condições inesperadas para o estado em se está porém de ocorrência normal para o sistema. Mensagens do tipo "tempo ultrapassado" ou condições de alarme.

Este conjunto é fornecido em forma de código protótipo em linguagem *Java* (no caso presente gerado pelo próprio sistema de auxílio a projeto utilizado, o *Rose*) e o sistema Prototipador gera o agente configurado com os protocolos, já em sua forma executável.

O fato de o sistema gerar um código executável representa um avanço e fator de diferenciação em relação a outros sistemas testado no decorrer do presente trabalho.

A sub-seção 4.4 fornece maiores informações comparativas entre o sistema Prototipador e sistemas como o *Jade* (*Java Agent DEvelopment Framework*, http://jade.cselt.it).

A Figura 4.12 mostra o módulo de configurador de perfis que faz interface com o agente *Sheik*. A janela mostra as palavras-chave encontradas pelo algoritmo de busca para um novo usuário.

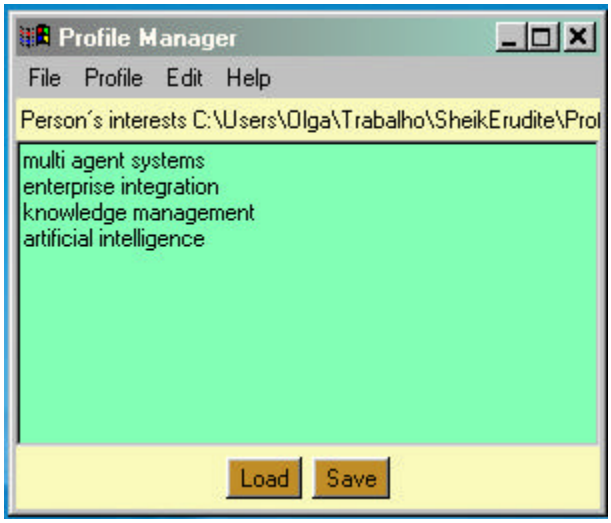


Figura 4.12 Gerenciador de dados do usuário (*Profile Manager*)

O *Profile Manager* apresenta a seguinte opção de menu: *File*, *Profile*, *Edit* e *Help*.

- File: Abre e salva arquivos.

- Profile: Possibilita a criação de Perfis configuração e captura automática de palavras-chave.

  - Contém sub-menus *Config*, *Open* e *Build*

- *Config*: Utilizado na geração de novo Perfil. Pergunta ao usuário por um diretório onde possa criar um arquivo Perfil.
- *Open*: Permite a navegação através dos diretórios. Após a seleção do diretório e o arquivo pré-definido de palavras-chave "*PersonProfile.key*", o sistema irá procurar por arquivo de perfis já existente e abri-lo.
- *Build*: Realiza a extração automática de palavras chave no diretório selecionado e escreve o resultado no arquivo "*PersonProfile.key*"

- Edit: Permite a edição e configuração de Perfis (*Profiles*) e Palavras-chave (*Keywords*). Contém sub-menus *Register* e *Keywords*.

- *Register*: Permite a configuração definindo dados estáticos (nome, e-mail) apresentado na Figura 4.13.
- Keywords: Apresenta o conjunto de palavras-chave para visualização e edição.

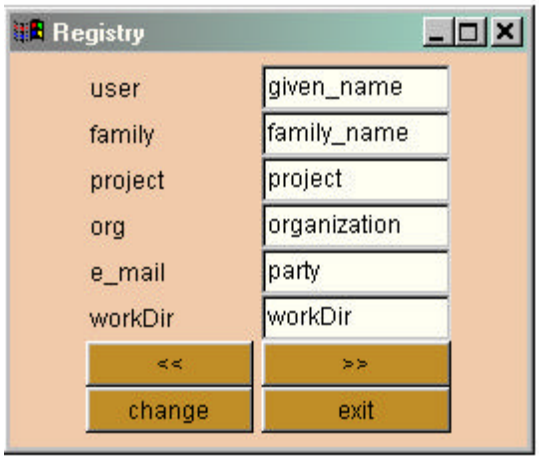


Figura 4.13 Interface para configuração de dados estáticos (*Register*)

4.4. Conclusão

Neste capítulo foi apresentado o projeto detalhado do sistema multi-agente utilizado no sistema. Para implementar o sistema, foi utilizada a arquitetura fornecida pelo capítulo anterior onde foram determinadas as restrições do sistema.

A modelagem utilizando UML tem como objetivo a utilização da "língua franca" do desenvolvimento de programas de modo que se possa exportar o modelo para uma implementação concreta utilizando-se uma variedade de linguagens de programação. O UML tem essa propriedade, poder ser lido por diversos desenvolvedores e mesmo o usuário final, possibilitando uma compreensão geral do sistema e mesmo verificando se foram atendidas as exigências do sistema.

No caso do presente trabalho os diagrams de estado e de seqüência do são utilizados pela ferramenta de desenvolvimento utilizada, o Prototipador. Esta ferramenta possibilita, após o projeto do sistema, sua prototipação em linguagem Java, pronto para testes e resultado final.

Outras plataformas foram testadas, como, por exemplo, o sistema *Jade*. O *Jade* tem um apelo por ter sido testado pela *FIPA* (*Foundation for Intelligent Physical Agents*, www.fipa.org) com vários outros programas e provado sua capacidade de obedecer ao protocolo de comunicação de agentes. Outra vantagem do Jade é que seu *framework* tem várias ferramentas de depuração. Possui também uma equipe que gerencia uma lista de discussão para desenvolvedores, resolvendo muitas dúvidas de implementação e possíveis *bugs* do próprio *Jade*. Para o problema que se pretende resolver o Jade se mostrou uma ferramenta poderosa demais, podendo onerar o usuário com um programa que poderia sacrificar o desempenho da sua estação de trabalho. O Jade, não sendo uma ferramenta de prototipação, exige que todo o desenvolvimento seja realizado pelo próprio desenvolvedor.

Por outro lado, o programa Prototipador entrega uma primeira versão a partir da definição do diagrama de estados em UML. Outra vantagem é a capacidade de integração com os outros aplicativos (*Protégé* e *Kea*) utilizados no trabalho. O problema enfrentado pelo Prototipador é a documentação (comum em trabalhos de desenvolvimento) e a falta de uma equipe como a do Jade, dedicada aos melhoramentos do seu produto.

Capítulo 5

Descrição da ontologia

Este capítulo apresenta, seguindo o modelo de Gruber (1993) para projetos de ontologias, o modelo da ontologia, a ferramenta utilizada para implementá-la e a taxonomia baseada nas atividades que envolvem a área de Manufatura (IMTI, 2000). São detalhados os modelos projetados em UML, sua implementação no ambiente *Protégé*.

5.1 Introdução

Sistemas baseados em conhecimento estão localizados em plataformas computacionais heterogêneas, implementadas em diferentes linguagens e protocolos. Como visto no Capítulo 3, um dos requisitos do sistema é a interoperabilidade. Como alcançar este requisito? "Para comunicação no nível de conhecimento são necessárias convenções em três níveis: formato de representação de linguagem, protocolo de comunicação de agentes e especificação do conteúdo do conhecimento compartilhado" (Gruber, 1993). O formato de representação da linguagem no caso do presente trabalho pode ser de três tipos: html, xml e rdf+xml. Todas essas representações estão popularizadas em banco de dados e informações disponíveis na *web*. O protocolo de comunicação entre os agentes é definido pela plataforma Prototipador de Agentes, descrito na seção 4.3. O conhecimento compartilhado exige alguns acordos para interoperabilidade. Este acordo é conseguido através da construção de uma ontologia, ou várias, que possam traduzir o meta-modelo envolvido na construção do modelo final.

Neste capítulo estão detalhados os projetos do modelo em UML e também a sua implementação no ambiente *Protégé*. A facilidade oferecida pelo ambiente de implementação concorre na facilidade de manutenção e visualização do sistema.

5.2 Critérios para criação de ontologia

Ontologias formais são projetadas. Os critérios apresentados aqui foram buscados nos trabalhos clássicos de Inteligência Artificial aplicada à aquisição de conhecimento, como de Gruber, Finin, Guiness, Ushold e outros. Como Gruber afirmou, muitas vezes não é possível cumprir um critério sem o sacrifício de outro. Torna-se dependente do uso o rigor ao cumprir um ou outro critério.

– Clareza: uma ontologia deve comunicar efetivamente a intenção do significado dos termos definidos. A definição deve ser objetiva. Todas as definições devem ser documentadas em linguagem natural.

– Coerência: deve ter inferências aprovadas que são consistentes com a definição. Ao menos os axiomas definidos devem ser logicamente consistentes. Coerência também deve ser mostrada na documentação em linguagem natural de modo que haja concordância entre as definições formais e a documentação.

– Expansibilidade: sendo projetada para permitir o compartilhamento de vocabulário, uma ontologia deve poder crescer monotonicamente.

– Mínimo comprometimento com codificação: agentes são implementados em linguagens, estilos e sistemas de representação dos mais variados. De modo que a codificação do conhecimento não deve depender de codificação.

– Mínimo comprometimento ontológico: as atividades de compartilhamento de conhecimento devem conter um conjunto de significados já acordados fazendo parte do núcleo possibilitando que especializações sejam feitas na medida da necessidade. Só os elementos essenciais devem ser definidos.

Com base nestes critérios gerais, nas restrições segundo os critérios de cooperação, coordenação e comunicação, foram desenhadas as classes componentes do sistema. Essas classes foram desenhadas no Rose juntamente com o sistema de agentes pois eles interagem. As classes que fazem parte da ontologia compõem a base de conhecimento do agente-servidor. A implementação será descrita mais adiante e foi realizada no ambiente *Protégé*.

Uma consideração a mais a ser feita é que, como no caso dos agentes, utilizando-se uma metodologia orientada a agentes, o sistema foi desenhado em maior detalhe em UML. No caso da

base de conhecimento acontece fato semelhante ao caso dos agentes onde a ferramenta de documentação é insuficiente para especificar totalmente o sistema. O *Protégé* não possui nenhuma maneira (ainda, pelo menos) de implementar nenhum dos métodos desenhados. E nem o Rose (o UML) possui uma maneira de representar as instâncias definidas no *Protégé*. Classes e atributos (classes e slots) porém são desenhadas em um e mapeadas no outro diretamente, e o inverso também é verdade.

5.3. O ambiente *Protégé* para construção de ontologias

O *Protégé* é um ambiente para desenvolvimento de sistemas baseados em conhecimento e continuou evoluindo ao longo da última década (Gennari, 2002). O fato mais importante a respeito do *Protégé* é sua comunidade externa que coopera entre si desenvolvendo extensões ao próprio programa, depurando o código e utilizando-o para os mais diversos fins. Sem essa comunidade provavelmente ele seria mais uma entre tantas ferramentas do mesmo tipo já desenvolvidas, mesmo mais poderosas que ele, mais recentes e com mais recursos como as descritas no Capítulo 2 desta tese, a saber: *CommonKADS*, *Oil* e o *DAML*. O *Protégé* projetou sua própria ontologia usando o formalismo de uma linguagem padrão baseada em *frames,* utilizando conceitos como classes, instancias, *slots* e *facets*. Muitas de suas idéias foram baseadas no sistema OKBC (Chaudri, 1998; Noy, 2000).

5.3.1 Arquitetura do ambiente

O presente trabalho usa a interface programática para informar os agentes e a interface H/M do ambiente para atualizar novas informações ou modelos via administrador de sistema. Também o usuário que desejar informações diretamente pode usar a interface do *Protégé*.

A base de conhecimento do *Protégé* inclui a ontologia e instâncias individuais das classes com valores específicos para *slots*. As classes constituem uma hierarquia taxonômica e meta-classe é uma classe cujas instâncias são classes.

5.3.2 Os componentes do sistema: vista do meta-modelo

Como foi dito anteriormente o modelo do sistema foi desenhado no *Rose* apesar dele não representar totalmente o sistema. De modo que o Esquema e o Quadro de Referência onde se baseia o sistema foram desenhados nele. Foram desenhados também o modelo do Usuário.

Estes componentes juntos e interrelacionados formam a base para a implementação do sistema tanto no *Protégé* quanto no Prototipador de agentes.

Seguindo a recomendação de Noy e McGuiness ( )são sete os passos para se escrever uma ontologia:

– Determinar o Domínio e o Escopo da ontologia. No caso do presente trabalho o domínio é o da Manufatura (como ilustra a Figura5.1. e 5.2.), dentro do escopo de Projeto de Produto, a fase de concepção de um novo produto. A ontologia será utilizada para auxiliar na classificação de novos vocabulários, comuns para uma certa gama de especialistas mas não para a comunidade de usuários e potenciais usuários. A ontologia será utilizada para se construir o modelo formal que incorpora o vocabulário dos inscritos na rede de habilidades.

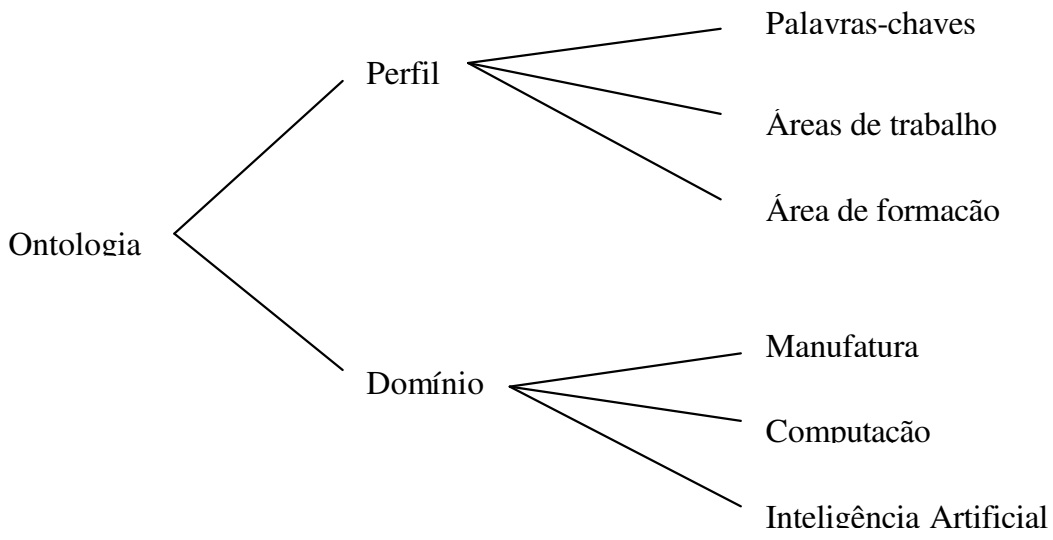


Figura 5.1 Ontologia do sistema de recomendação

– Considerar a reutilização de ontologias já existentes – na biblioteca de ontologias do *Ontoserver* algumas foram pesquisadas e estudadas. A escolha de uma taxonomia desenvolvida por empresas de manufatura e seu relacionamento com uma ontologia que defina um perfil que incorpore o vocabulário dos participantes da rede, tem esse sentido.

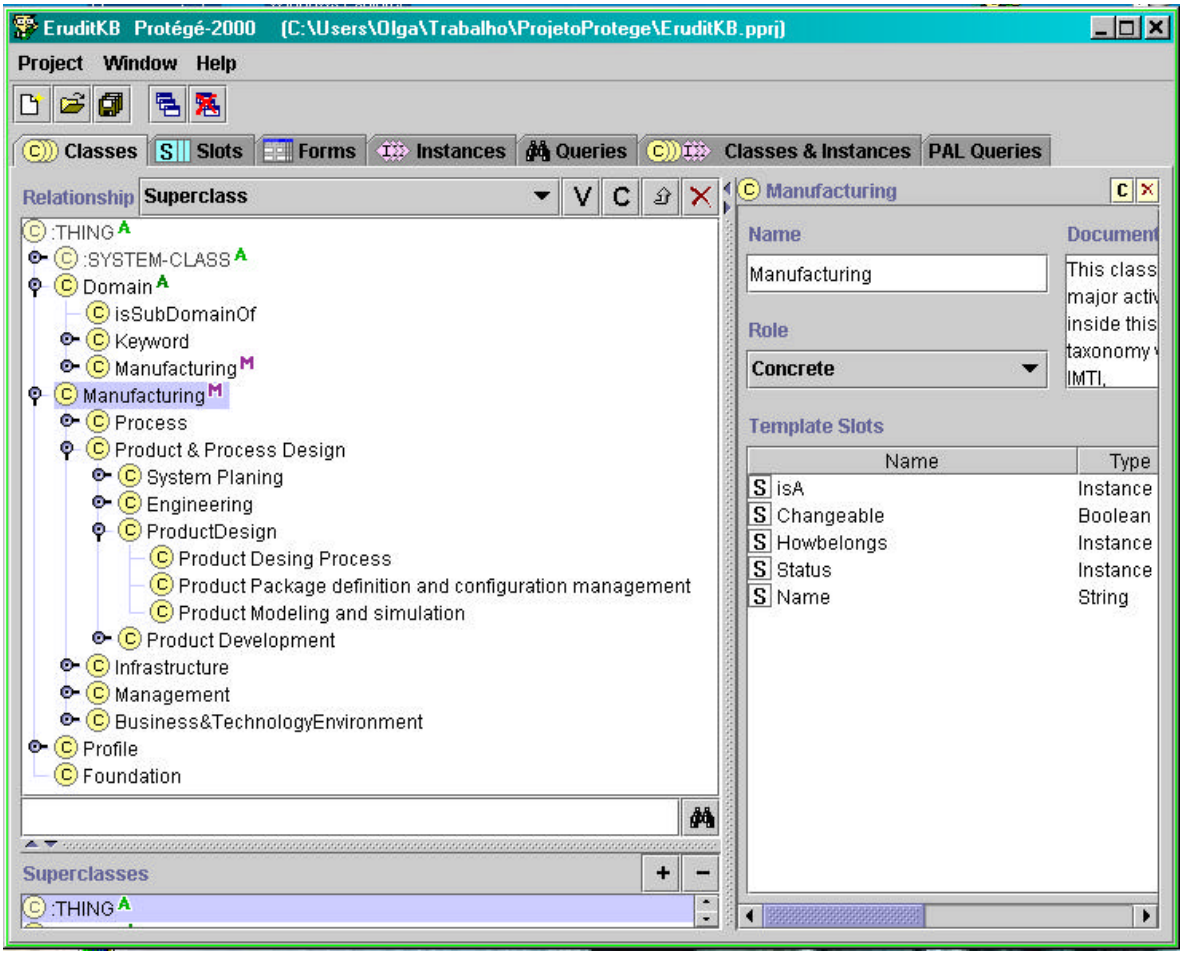


Figura 5.2. A taxonomia de Manufatura definida como classe no *Protégé*

– Enumerar termos importantes na ontologia – alguns termos são realmente diferentes e possuem estranhas interpretações, como é o caso de ontologia, por exemplo. Neste trabalho, alguns termos foram cuidadosa e detalhadamente explicados. Além de ontologia, onde buscou-se as definições de Nicola Guarino, especialista no assunto, buscar as informações; termos como frames, slots, first-order logic, frame logic, foram cuidadosamente explicados. Outros como, *Framework*, *Domain* para se entender a importância do meta-modelo que se quer implementar e finalmente termos da própria taxonomia onde conceitos ligados tanto à área de manufatura quanto à de computação estão citados mas não definidos.

– Definir as classes e as hierarquias de classes - esse trabalho é complexo e deve ser reavaliado continuamente. Alguns conceitos são facilmente identificados como classes outros são definidos como slots. A figura 5.3 mostra como foi implementada a classe perfil da pessoa.

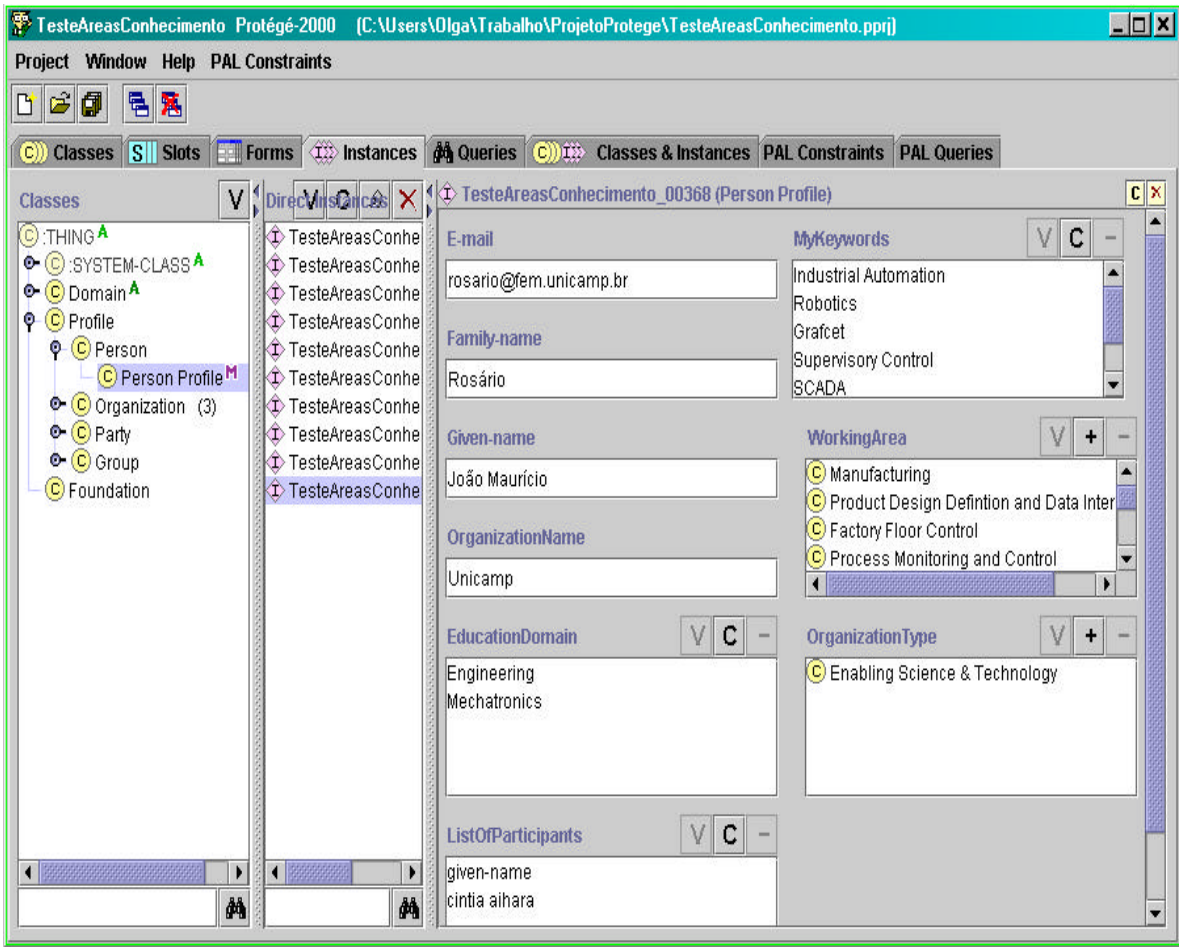


Figura 5.3 Classe PersonProfile implementada no Protégé

– Definir as propriedades das classes – *slots* Uma vez decidida quais as classes, uma definição mais detalhada sobre os seus componentes se faz necessária. No presente caso, a definição em UML foi fundamental para identificar os Pacotes (*Packages*) que no caso do *Protégé* foram transformadas em super classes e metaclasses e as classes, atributos e métodos pertencentes a cada classe. Como já foi dito anteriormente, não são todas as definições do modelo UML que têm mapeamento um para um no ambiente *Protégé* e o inverso também se aplica. O *Package* Profile é implementado tanto no ambiente do usuário e consultado pelo agente Sheik quanto no *Protégé* e consultado e alterado pelo agente *Erudite*.

– Definir as facetas e os *slots* – É continuada e cada vez mais detalhada a especificação onde facetas descrevem tipos, valores possíveis ou permitidos. Descrevem a cardinalidade, valores

máximos e mínimos. As restrições aplicadas sobre possíveis valores para o slot área de trabalho possuem os valores definidos na sub-classe de Domínio que é a área de Manufatura.

– Criar instâncias – este trabalho possui várias instâncias e uma forma dinâmica de construí-las. As instâncias representantes de diferentes perfis vêm associadas às pessoas que se cadastrarem para participar (*subscribe* do agente-usuário). Para não partir "em vazio" algumas pessoas foram cadastradas, os membros e participantes do projeto PIPEFA (Rosário, 1999). As instâncias de palavras chaves também são preenchidas de acordo com o resultado do programa aplicativo KEA (Witten, 1999). São estas as instâncias armazenadas no *Protégé*. O intuito principal é fazer esta base crescer com o uso automaticamente, e sua manutenção é realizada pelo agente Erudite.

Os próximos itens descrevem os componentes do sistema, seu desenho no *Rose* e documentação auxiliar.

5.3.3 Componente Gerenciador de Perfil

A Figura 5.4 mostra um diagrama de classes do componente gerenciador de perfis. Esta classe é implementada de forma duplicada, ou parcialmente duplicada, já que cada agente-usuário possui uma cópia do perfil de seu usuário e o agente servidor possui uma cópia de cada agente usuário como instância da classe *Profile* definida no *Protégé*.

Uma outra característica da classe *Person* é que ela é parcialmente implementada (seus métodos) tanto no agente Sheik quanto no agente Erudite. As classes implementadas no *Protégé* podem ser vistas na Figura 5.5 e referem-se às classes *Person*, *PersonOrganization*, *Organization*, *Party*, *Project*, *Skill* e *Scope*.

- Classe **PerfilDaPessoa** (*PersonProfile*): Esta classe herda os atributos das classes *Person* e os métodos da classe *Profile Manager*. Contém informações sobre um perfil básico de pessoas onde cada pessoa pode possuir uma ou mais formações, participar (ou ter participado) de um ou mais projetos, dados estáticos (ou com mudança menos freqüente) como endereço, e-mail, trabalhar (ou ter trabalhado) para uma ou mais organizações. Cada uma dessas informações é desenhada como classe separadamente, permitindo obtenção destas informações de múltiplas fontes, ou associações de diferentes formas. A criação destas classes fornece flexibilidade ao projeto, já que classes podem ser retiradas ou colocadas mais facilmente uma vez o projeto implementado. O perfil de uma pessoa pode mudar com o passar do tempo. Para guardar esse

tipo de relação, a classe Historian mantém dados e datas relacionados, já que estas informações é que compõem a habilidade de uma pessoa. A intersecção de várias áreas do conhecimento pode trazer um perfil único. A classe *Scope* contém o atributo *Formation* que relaciona o usuário com as classes que definem as áreas de conhecimento de maneira formal, como engenharia mecânica, engenharia elétrica, computação, etc.

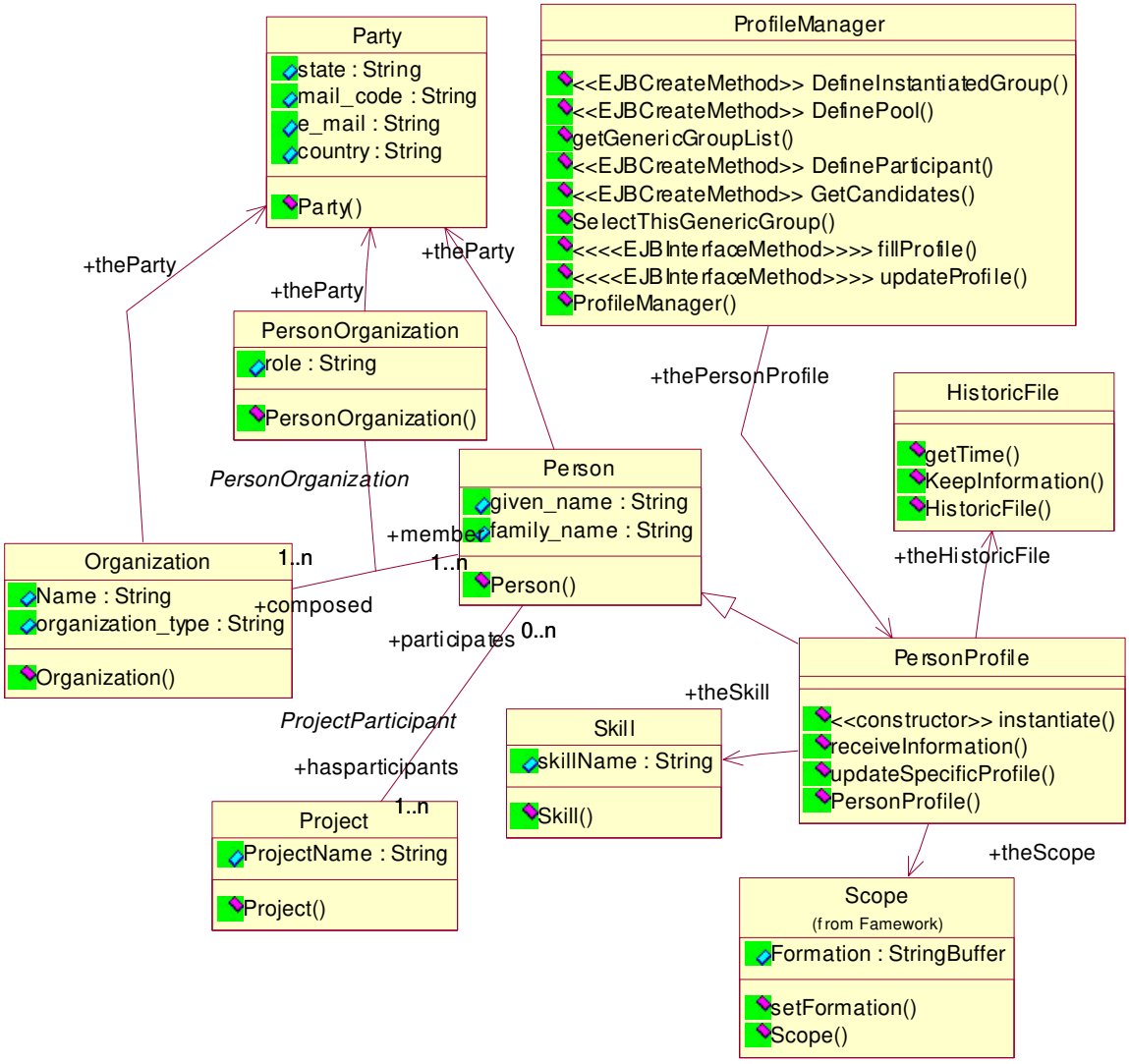


Figura 5.4 Gerenciador de Perfis

– Classe **Gerenciador do Perfil -** Esta classe fornece os métodos de acesso das informações do usuário. Atualizações também são feitas através dela. Seus métodos também são utilizados na implementação do agente-servidor para a construção da lista de participantes que possui perfil semelhante ao do usuário de onde a questão formulada (*query*) é proveniente.

– Histórico – Esta classe guarda os acontecimentos registrados pelo agente na evolução da sua percepção do usuário. Guarda as últimas listas recebidas, as palavras-chave utilizadas ultimamente, os documentos consultados e os dados que foram modificados por novos como projetos anteriores, organizações anteriores, formações anteriores, etc.

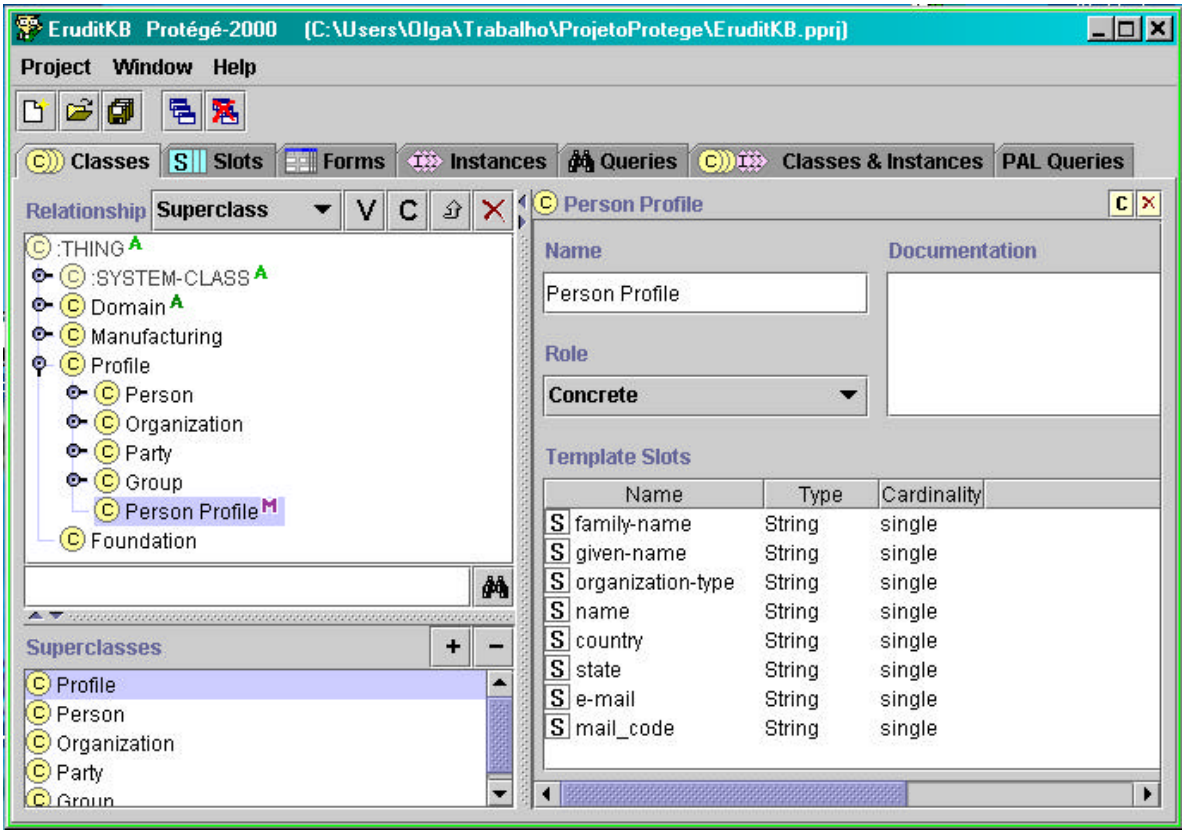


Figura 5.5. Classe Perfil (*Profile*) implementada no *Protégé*

5.3.4. Componentes do Quadro de Referência

O Pacote Quadro de Referência fornece as metaclasses do sistema implementado. As definições dos relacionamentos existentes entre as classes dos Pacotes Domain e Profile são reguladas pelas classes do Quadro de Referência. O Quadro de Referência não possui métodos,

ele define uma série de classes com atributos que são utilizados pelas outras classes. A Figura 5.6. mostra o diagrama UML do *Package* e suas classes são detalhadas a seguir.

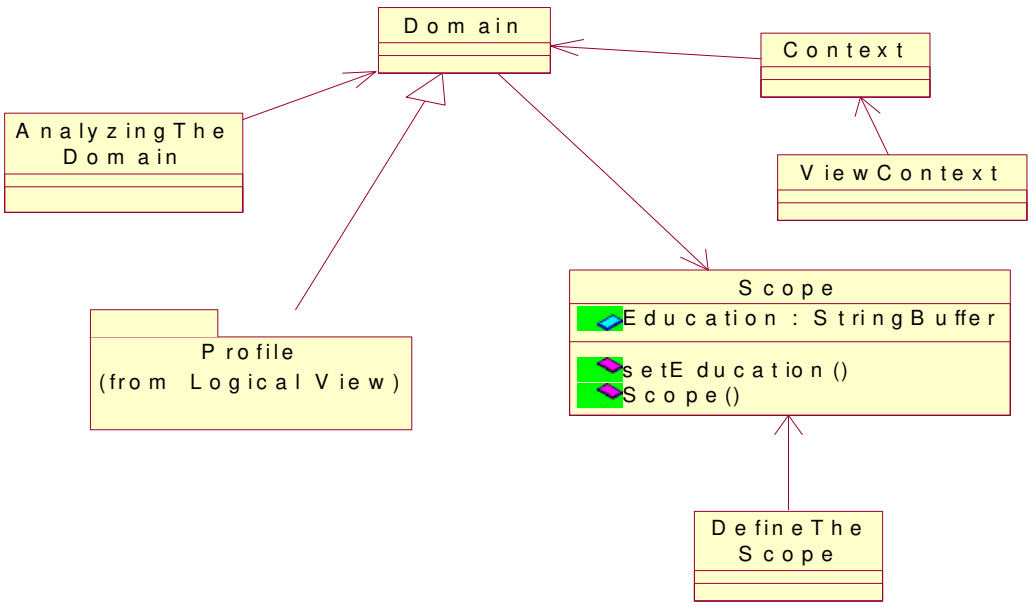


Figura 5.6 Diagrama do Componente *Framework*

Domínio (*Domain*) – esta classe define as possíveis áreas do conhecimento que o agente-servidor (a federação) tem acesso. Define os atributos gerais que deve ter a área. A Figura 5.7 mostra a implementação no *Protégé* da classe Domínio.

– Contexto (*Context*) – usado para especificar uma restrição em relação às áreas de conhecimento do usuário

– Escopo (Scope) – utilizado para determinar o nível e o tipo de educação do usuário e qual relação com o Domínio.

5.3. Conclusão

Neste capítulo foi apresentada a metodologia para desenvolvimento da ontologia. Os passos a serem dados foram descritos e exemplificados com o próprio modelo desenvolvido. À medida que o modelo do sistema foi sendo desenvolvido em UML a possibilidades de implementação que se mostrou mais viável, de início ,foi a criação de uma ontologia partindo-se do rascunho sendo implementada em linguagem Java.

Na medida da evolução do conceito o ambiente *Ontolíngua* passou a ser utilizado. Este ambiente permitia uma interação amigável, com um tutorial para orientar os primeiros passos. A ampliação do entendimento dos conceitos foi alcançada com este ambiente.

O problema seria na integração do ambiente com agentes, que manipulariam esse ambiente. A solução surgiu com o uso do *Protégé*. O ambiente de desenvolvimento tem versões para várias plataformas, é independente, pode ser instalado no computador pessoal e trabalhar com ele não havendo necessidade de recursos especiais tampouco programas auxiliares em servidores remotos, como é o caso tanto do Ontolíngua quanto do OilEd. O ambiente oferece autonomia para se desenvolver não somente as ontologias como a integração com outros aplicativos e mesmo outros ambientes.

Como o modelo inicial foi desenvolvido em UML era de se esperar que fosse implementado em ambiente que comportasse todas as funcionalidades nele descritas. Mas nem o UML suporta algumas qualidades do *Protégé* nem o contrário. Métodos, descritos no UML não são implementáveis no *Protégé*. Nem instâncias, que formam a base de conhecimento, são definidas pelo UML. Como parte do modelo, justamente os métodos foram implementados por agentes, de modo que as classes foram implementadas no *Protégé*.

O próximo capítulo apresenta os resultados da implementação do sistema de recomendação e é feita uma comparação com sistemas semelhante, as soluções propostas e em que este trabalho contribuiu.

Capítulo 6

Avaliação do sistema e trabalhos relacionados

Neste capítulo é realizada a avaliação do sistema a partir da sua eficiência em selecionar pessoas que poderiam fazer parte da mesma rede de habilidades. Ao mesmo tempo o sistema é comparado com outros trabalhos na área de sistemas de recomendação.

Porque um sistema de recomendação é a resposta tecnológica para uma ferramenta que auxilie na construção de uma rede de compartilhamento de habilidades? A concepção do sistema relacionou o perfil dos participantes com suas áreas de conhecimento e relacionou o conhecimento informal com conhecimento formal. O conhecimento formal é representado de forma inequívoca através de uma ontologia. Enquanto o conhecimento informal é obtido através do vocabulário utilizado na linguagem corrente entre as pessoas. Este conhecimento tem uma amostra dele colhido através de palavras ou frases chaves retirados de documentos fornecidos pelo próprio participante.

Nos dois capítulos anteriores foi visto o projeto do sistema em UML, sua implementação utilizando-se o ambiente de desenvolvimento de ontologias *Protégé* e também o ambiente Prototipador de Agentes. Enquanto o UML privilegia uma visão geral, de mais alto nível do sistema enquanto componentes, o sistema de desenvolvimento, tanto em termos de agentes quanto de ontologia, fornecem visões mais aprofundadas da implementação.

6.1 Introdução

O sistema prototipado constitui-se em uma ferramenta de suporte à cooperação, pertencente à classe dos sistemas de recomendação. A motivação na criação de um protótipo está descrita no Capítulo 1, onde é colocada a necessidade de compartilhamento de conhecimento, depois

evidenciada pelo Capítulo 3 onde foi proposta uma arquitetura que satisfizesse os requisitos de cooperação, comunicação e coordenação.

A solução se baseou no relacionamento que conhecimento não-estruturado possui com palavras chave retiradas de um texto e a relação que ontologias possuem com conhecimento estruturado. O relacionamento entre conhecimento não estruturado e conhecimento estruturado fica estabelecido na relação entre palavras-chave e ontologias.

Neste capítulo iremos colocar a contribuição que a solução proposta trouxe aos sistemas de recomendação através de comparação com pesquisas significativas na área.

6.2 Panorama dos sistemas de recomendação

Os trabalhos relacionados se incluem como sendo sistemas de recomendação, ou sistemas de referência, redes sociais, computação afetiva, sistemas de recuperação de informação *just-in-time*, comunidades-de-melhores-práticas (*communities of practice*), sendo estes os nomes mais utilizados, porém não esgotando a criatividade da comunidade de desenvolvedores. É necessário, inclusive, dar ênfase ao fato da comunidade estar sempre criando novos vocabulários, seja para diferenciar um sistema do outro ligeira ou totalmente (acrescentando um “2” ou um “e-” ou “i-” , significando eletrônico ou inteligente) mas criando uma palavra nova em termos de busca direta.

Uma recomendação (um conselho) pode ser vista como uma informação em resposta a uma questão explícita ou implícita. Em outro contexto, pessoas em posição de decisão não estão sempre formulando perguntas mas apreciam que seus colaboradores, conhecedores do contexto da escolha e de algumas alternativas, por sua própria experiência ou observação dos outros, lhes proponham por iniciativa própria, uma solução.

A geração de um conselho se faz por agregação de informação seguida da distribuição aos destinatários interessados em utilizá-la. A aquisição de informação utiliza máquinas de busca e outras técnicas clássicas. Mas o desafio de um sistema de recomendação vai além da busca de informação. O maior valor destes sistemas consiste em recomendar sem ter recebido perguntas explícitas e aumentar os interesses do usuário em domínios nem sempre mencionados em seu perfil. O desafio está em tirar partido do forte potencial informativo do tecido relacional de toda pessoa, à semelhança de uma equipe e seus vários colaboradores.

Os sistemas de recomendação podem se basear em uma especialização crescente no perfil do usuário sem considerar uma mudança temporal ou considerando uma diversidade instantânea coletando todo o tipo de informação afogando o usuário com todo tipo de informação.

O sistema de recomendação pode, ele mesmo, estar baseado em recomendações. Ou seja, se uma determinada fonte de informação é recomendada por alguém de confiança, então o sistema reconhece e também recomenda. O sistema tem um tempo inicial para começar a funcionar (*cold-start effect*), visto que ele começa "em vazio".

Outros sistemas se baseiam em publicações tendo como fonte o *Science Direct* (que também usa um sistema de ranqueamento feito por especialistas e uma máquina de busca por palavras) e outras fontes de divulgação.

Neste sentido o sistema proposto possui duas fases: a concepção do sistema e a manutenção/recomendação. A concepção constitui na construção do meta-modelo e dos núcleos de conhecimento formais, ou domínios. A recomendação constitui, pela agregação por filtros de conteúdo ou filtros colaborativos, do fornecimento da informação a partir de restrições e a manutenção na atualização das instâncias dos domínios, que constitui a base de conhecimento.

Estes tipos de sistema são utilizados pela *Microsoft* e *National Secutrity Agency* e ou o profissional ou o administrador do sistema realizam a atualização dos dados de perfil existentes na sua base. Seu uso é normalmente destinado a achar os profissionais com determinadas características, já que são organizações com mais de 50.000 pessoas. Esses dados se encontram em banco de dados e para construir a base a maioria das empresas emprega uma taxonomia própria, quando emprega. Os sistemas aqui apresentados possuem uma diferença de abordagem em relação a estes sistemas apesar de se destinarem basicamente ao mesmo fim. Os sistemas que são aqui descritos têm como objetivo o usuário e não a corporação. A corporação, que pode ser uma única ou virtual ou mesmo, como no caso do sistema implementado, empresa cooperativa (formada por várias empresas que cooperam), pode vir a ser beneficiada mais pelo caráter de compartilhamento de conhecimento da ferramenta do que por um objetivo organizacional.

6.2.1 Filtragem colaborativa

A filtragem colaborativa associa sintática ou estatisticamente um objeto a um contexto. Uma requisição sobre um conteúdo referenciado e susceptível de ser aconselhado segundo a

opinião coletiva. Se baseia no princípio da confiança seja em jornais, críticos, especialistas, nos quais se tem confiança.

Na filtragem colaborativa o usuário é, além de cliente, participante. Ou seja, ele compartilha as suas recomendações sobre determinado assunto, técnica, documento, site, etc. O sistema verifica o ranking desta recomendação pelo número de seguidores da recomendação. De modo que o sistema pode recomendar tendo o apoio prévio de outros utilizadores. Não existe gerência de perfil mas de associações de contexto a uma determinada fonte (seja ela uma URL, artigo, técnica, etc). Como exemplo da utilização desta técnica temos os sistemas *Yenta* (Foner, 1999), *Siteseer* (Rucker, 1997), *GroupLens* (Konstan), *ReferralWeb* (Kautz, 1997), *Ricochet* (Bothorel, 1999), entre outros.

O *Siteseer* é baseado em uma ramificação de *bookmarks*, classificados hierarquicamente em diretórios onde o título é explícito o suficiente. O sistema faz a comparação entre ramificações de diferentes pessoas e se há pontos em comum suficientes, propõe a extensão das listas de *bookmark*. De modo que o *bookmark* de uns se torna recomendação para outros.

O *ReferralWeb* combina redes sociais, que são redes de colaboração informal entre colegas e/ou amigos ou participantes de uma lista de discussão, e estendem o conceito para grupos que não necessariamente se conheçam. Baseia-se no fato de que pessoas que detém um determinado conhecimento não necessariamente o divulgam. A divulgação deste profissional ocorre através de uma recomendação pessoal. A rede ajuda a alcançar o profissional que terá maior boa-vontade (porque recomendado) em responder e a pessoa que procura terá maior confiança na resposta. A filtragem se realiza em co-ocorrência de nomes em documentos disponíveis na *Web* em assuntos próximos. As fontes destes documentos podem ser *links* em páginas pessoais, lista de co-autores em artigos técnicos e citação em artigos, trocas entre indivíduos gravadas em arquivos de novidades em rede, mapas organizacionais (como de departamentos de universidades). A rede é construída de forma incremental e utilizam-se grafos para mostrar a relação entre uma pessoa e as outras da rede. As pessoas são citadas nominalmente, o sistema utiliza-se do *AltaVista* para fazer sua pesquisa na *Web* e agentes para realizá-lo. O sistema tem um servidor onde o usuário insere o assunto sobre o qual quer verificar quem são os especialistas.

O *Yenta* apresenta pessoas com o mesmo interesse e que não necessariamente fazem parte das pessoas que publicam ou participam escrevendo em uma lista de interesse (podem ser apenas

observadores). O sistema é baseado em agentes organizados de um modo descentralizado e é ponto a ponto na maneira de estabelecer o contato. O sistema é residente no computador do usuário e está sempre operacional observando a proximidade que ele pode alcançar em termos de interesse do seu usuário. Utiliza um sistema de agentes que recomenda o outro caso esteja no mesmo conjunto de assunto, de uma forma que o autor intitula de *hill-clibing*, ou seja aos poucos se alcança o objetivo. Existe uma pré-formação do conjunto 'bom' através da coleta de pequenos textos de e-mails, grupos de novidades, etc, para achar o conjunto. Um vetor vai desenhando esta pré-formação até desenhar o conjunto baseando-se no conhecimento de outros agentes que ele vai contatando durante este período. Este processo é longo mas só ocorre na fase inicial.

GroupLens é baseado na recomendação de pessoas e pontuações que os membros do grupo vão fornecendo ao sistema. Utiliza o *Usenet News* como fonte para suas comendações.

O *Ricochet* forma grupos com a integração progressiva de modificação dos dados, com algoritmos adaptativos, com um sistema de meta-aprendizado exterior ao sistema de multi-agentes que forma o sistema. Não trabalha com a noção de perfil do usuário, mas sim com o interesse que o usuário vai demonstrando ao longo do tempo e se adaptando a isso com a ajuda de um agente mediador.

Todos os sistemas utilizam a *Web* como forma principal de adquirir as informações necessárias para a realização do seu objetivo último que é servir ao usuário, como agente pessoal.

6.2.2 Recomendação sem análise de conteúdo

O conteúdo tem atributos que não fazem parte do sistema de recomendação. De modo que podem ser incluídas informações de vários tipos como música, pintura, informações multimídia, que sem a utilização do sistema colaborativo seriam muito complexas para fazerem parte do sistema de recomendação. Ou seja alguém pode recomendar mas o sistema não faz análise de conteúdo e sim da meta-informação.

Como exemplo temos a associação de peritos científicos por encadeamento de referências bibliográficas sem compreensão do artigo científico. O sistema *Phoaks- People Helping One Another Know Stuff* (Terveen, 1997), onde as anotações de recomendações por parte dos usuários são partilhadas e permitem a construção estatística de correspondência entre URLs recomendadas e contexto da recomendação.

Contrapondo a recomendação por filtragem colaborativa de conteúdo, onde a especialização do perfil do cliente é o objetivo, a filtragem colaborativa diversifica as recomendações classificando-as segundo o contexto pelo qual o usuário está passando naquele momento.

Quanto mais os usuários participam mais o sistema explora os retornos da experiência. Um retorno de experiência pode ser anotações de recomendação ou o simples fato de ter consultado. Um exemplo de filtragem colaborativa é o www.Amazon.com .

6.2.3 Método de classificação

Técnicas fatoriais: operam a redução da dimensão de certas representações multidimensionais. Apóia-se sobre a álgebra linear, em particular sobre as propriedades matemáticas das matrizes retangulares. A classificação reagrupa os indivíduos em um número restrito de classes homogêneas.

Três famílias de métodos: métodos diretos, *single-pass* e algoritimo iterativo.

- Métodos diretos (*graph theoretical algorithm*): baseados em matrizes de distâncias inter-indivíduos grupados dois a dois. Como o interesse é sobre um indivíduo em particular a matriz pode ser explorada procurando-se os n indivíduos mais próximos como primeiro refinamento para outros métodos.
- *Single-pass*: Definem-se os grupos de partida, um nó é definido para cada grupo e os dados são agregados no grupo onde o nó é o mais atraente. Neste método a rapidez de convergência vai depender do nó inicial escolhido. Se o tipo de perfil particular é dado, definem-se os nós de grupos por este tipo de perfil e o algoritmo classifica os indivíduos nas suas categorias.
- *Algoritmo* iterativo: a classificação é realizada em passos sucessivos. O primeiro pode ser utilizando-se um dos métodos anteriores. Esta família de algoritmos permite partir de nós de grupos (especificados por especialistas ou encontrados automaticamente) e evoluir estes nós de acordo com uma repartição ótima: o algoritmo para quando o critério de convergência fixado é satisfeito. O ótimo é local porque depende da configuração inicial.

6.2.4 Sistemas de compartilhamento de conhecimento

Estes sistemas têm como objetivo construir uma base de conhecimento e para isto utilizam-se de ontologias e de recomendações, como os sistemas vistos anteriormente. O objetivo continua sendo semelhante ao aproveitar o conhecimento já existente, como a ajuda de um especialista ou mesmo a construção de uma memória de melhores práticas.

O sistema *OntoShare* (Davies, 2002)é uma ferramenta projetada para auxiliar os usuários a compartilharem seus comentários, pontos de vista, com outros fazendo uma anotação ligada a uma informação. As interfaces da ferramenta são desenhadas para suportar este tipo de interação. Para classificar a informação e o comentário foram desenhadas classes ontológicas. As classes ontológicas são caracterizadas por um conjunto de termos (palavras-chaves ou frases) e a avaliação de pertinência entre informações e classes é feita através de um algoritmo vetor-co-seno. O sistema faz sugestões para o usuário que por sua vez pode aceitar ou modificar acrescentando alternativas. O domínio do conhecimento implícito na ontologia pode ser tomado como uma estrutura unificadora para fornecer à informação uma representação e semântica comuns. O perfil do usuário é levado em consideração para o compartilhamento das informações mas o sistema pode acrescentar informações a este perfil com o uso. A intenção é espalhar o conceito de comunidades-de-práticas. Cada usuário possui um agente que é seu representante.

O *QuickStep* (Middleton, 2002) tem como objetivo recomendar páginas da *Web* a partir do aprendizado adquirido através do próprio usuário tendo como base exemplos fornecidos pelo próprio usuário. Foi concebido como um sistema híbrido: é tanto baseado em recomendações feitas por um usuário (que então pode ser recomendado para outro usuário com perfil semelhante), quanto baseado em conteúdo: páginas bem classificadas, páginas mal classificadas e uma máquina de aprendizado. É utilizada uma representação vetorial dos termos para representar um dado artigo. Esta representação é baseada na freqüência que a palavra aparece no artigo. As recomendações são feitas de acordo com a correlação entre perfil e artigos devidamente ordenados e graduados. Além disso só é recomendado se ainda não consta da lista (URLs) usada (ou visitada) pelo usuário (assegurando pelo menos uma probabilidade de novidade). Artigos são graduados antes de sua apresentação ao usuário.

A ferramenta *OntoCoPI* se baseia em compartilhamento de interesses para identificar comunidades-de-prática. A informação é capturada através dos co-autores de artigos, participação

nos mesmos eventos, membros do mesmo projeto ou organizações. Utiliza ontologias para classificar as áreas de conhecimento. Um novo usuário é identificado através de pessoas relativas a ele e publicações relativas aos seus interesses.

O *PeopleFinder* (Becerra-Fernandez, 2000) utiliza agentes e técnicas de inteligência artificial para auxiliar no gerenciamento de sistema de conhecimento. O sistema é um repositório de conhecimentos que indica os profissionais que possuem um determinado tipo de conhecimento dentro de uma organização. A proposta prevê a automação do sistema com a utilização de sistemas de ferramentas de inteligência artificial, de modo a eximir o usuário da tarefa de atualizar o seu perfil. O sistema utiliza mineradores de dados para verificar junto aos documentos publicados os autores e co-autores, verificar as palavras-chave, e criar um dicionário sobre o domínio.

Outro sistema que utiliza agentes para satisfação do cliente (Pierre, 2000) satisfaz as *queries* pesquisando os vários *sites* onde a informação pode estar distribuída e trazendo-os para o usuário. O agente integra a parte cliente e servidora e o usuário interage com o agente através de várias janelas de interface. Dados como Domínio de interesse, fontes de interesse (editores, universidades, autores, companhias) e mesmo a formação do usuário, são inseridos através de uma interface com o agente. Cada sessão é gravada permitindo o retorno a uma situação prévia e mesmo filtragem sucessiva dos resultados obtidos.

6.3 Quadro comparativo entre os sistemas

A tabela 6.1 que se segue apresenta um resumo das principais características observadas nos sistemas de recomendação aqui resumidos. Estes sistemas procuram, em sua maioria, relacionar seus usuários com especialistas em uma determinada área. Este especialistas usualmente publicam em revistas que disponibilizam seus artigos na *web*.

As ferramentas que foram projetadas recentemente, no ano de 2002, utilizam ontologias para ajustar seus modelos. A maioria utiliza sistemas multi-agentes como forma de rastreamento dos seus usuários, seja concentrados em um servidor ou mesmo hospedadas na náquina do usuário como é o caso do Sheik e do Yenta. Algoritmos de pertinência são os mais utilizados enquanto máquinas de inferência só pelo Sheik e PeopleFinder. A maioria consiste de sistemas exploratórios, ou seja, são resultados de pesquisa, ainda não consistindo de produtos. Porém,

pelo que pôde ser observado no decorrer deste trabalho, a demanda por trabalhos deste tipo tem crescido, principalmente em aplicações para web semântica.

Tabela 6.1 Comparação entre os sistemas de recomendação

	Multi-agente	Recomendação	Ontolo-gia	máquina	Pertinên-cia
Yenta	S	Artigos científicos			S
Siteseer		*bookmark*			
GroupLens		*Usenet News*			
ReferralWeb	S	cientistas			S
Ricochet	S	diversos			S
Phoaks		cientistas	S		
OntoShare		opiniões	S		S
QuickStep	S	*Web*	S		S
OntoCoPI		*CoP*	S		S
PeopleFinder	S	especialistas		S	
Sheik	S	Perfis semelhantes	S	S	

6.4 O sistema implementado e seus resultados

A ontologia traz um formalismo ao sistema que permite que sejam reconhecidas as comunidades e que possa ser estendido para outras possibilidades. O relacionamento informal pode ser reconhecido pelo sistema formal e este então aponta outras fontes pertencentes à mesma fonte de informação. O conjunto informal dito de palavras ou textos-chave, é obtido com ajuda de um algoritmo de máquina de aprendizado e classificado com a ajuda da ontologia. A ontologia, como vimos no capítulo anterior, possui dados sobre a área de atuação dos profissionais ligados à Manufatura. A Manufatura tem suas áreas desenhadas como taxonomia, vocabulário estruturado que contém as sub-áreas pertencentes à área principal. Juntamente com o perfil do usuário e seu relacionamento com as sub-áreas em termos de pesos.

Este sistema pode ter algumas utilidades diferentes, uma vez que a taxonomia foi definida, como por exemplo, relacionamentos com documentos classificados para as diferentes áreas, casos informais contendo as "melhores-práticas", inter-relacionar as áreas com um produto, com pessoas que estão naquelas áreas e estão de alguma forma relacionadas com a área e com o produto.

A implementação realizada relaciona o vocabulário informal dos especialistas com uma área formal da manufatura e com outros especialistas dentro da mesma área ou com o mesmo vocabulário. Essa implementação é feita através de restrições feitos entre os relacionamentos lógicos que ligam as classes e por conseqüência as instâncias. O agente-servidor faz uso destas restrições aplicadas à base de conhecimento diretamente na interface programática do *Protégé*. A Figura 6.1 mostra em que módulo do Protégé é realizada a interação com o agente Erudite.

Para adquirir o vocabulário a ferramenta de máquina de aprendizado KEA (Witten, 2000), foi utilizada como ferramenta auxiliar do agente usuário. Palavras-chave fornecem meta-dados semânticos que resumem e caracterizam documentos. A ferramenta KEA identifica candidatos a palavras e textos-chave usando métodos léxicos, calcula valores característicos para cada candidato e usa um algoritmo baseado em redes *Bayesianas* para predizer quais são os candidatos a palavras-chave. O esquema da máquina de aprendizado primeiramente constrói um modelo de predição usando documentos para treinamento com palavras-chave conhecidas, usa então o modelo para achar as palavras-chave em novos documentos. Foi utilizado um teste amplo para avaliar a eficiência de KEA em termos de como as palavras-chave são corretamente identificadas, a partir da indicação do próprio autor.

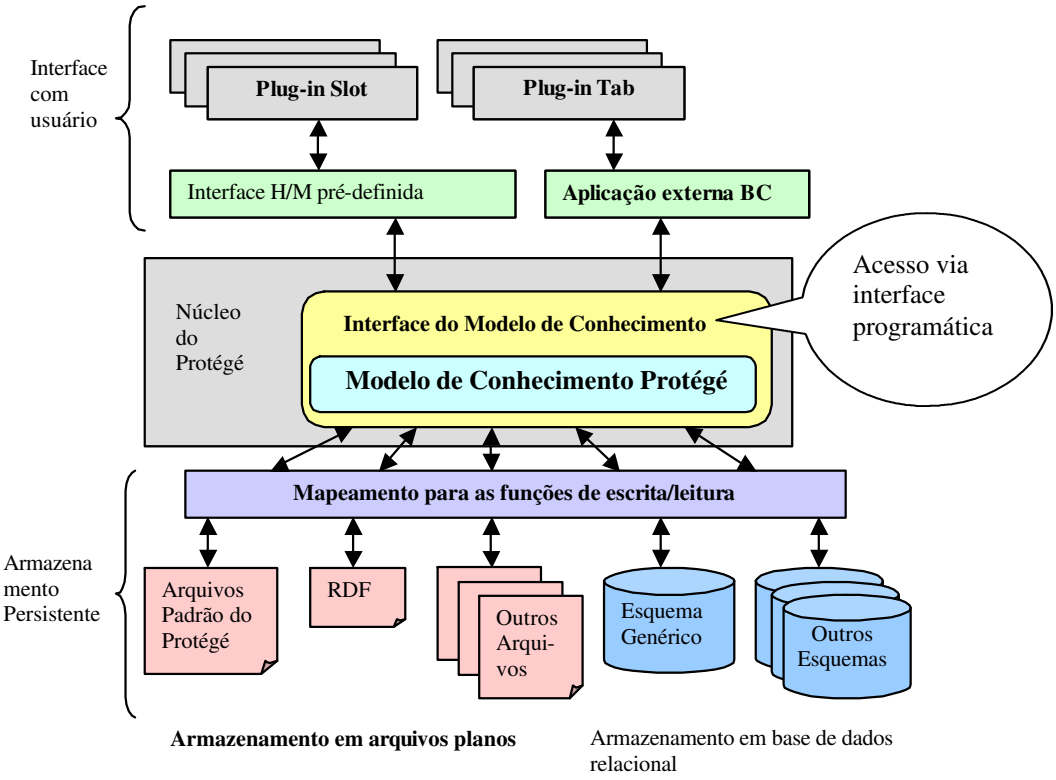


Figura 6.1 Acesso ao núcleo do Protégé via interface programática

A interface programática fornece acesso direto às classes do Protégé. Utilizando a classe edu.stanford.smi.protege.model.Project diretamente no arquivo protege.jar o agente Erudite tem acesso às instâncias da ontologia. Os elementos que formam a base de conhecimento podem ser lidos e comparados com os elementos enviados pelo agente *Sheik*.

As Figuras 6.1 e 6.2 exemplificam a interação do agente *Sheik* com o agente *Erudite* portando as palavras chaves que o algoritmo. Ao invés de usar um vocabulário controlado a ferramenta utiliza um conjunto selecionado de palavras-chave para sintonizar o algoritmo. Após a seleção de um primeiro conjunto de palavras-chave o algoritmo melhora seu desempenho, sintonizando o descobrimento das palavras-chave relacionando com o primeiro conjunto novas palavras.

Uma vez inquirido pelo agente *Sheik* sobre o possuidor dessas palavras-chave o agente*Erudite* responde sobre quais instâncias satisfazem a condição recebida.

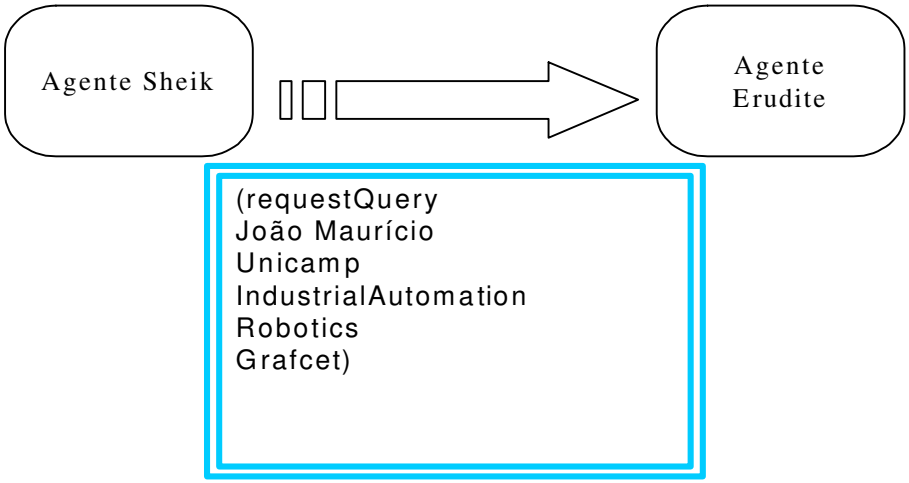


Figura 6.2 Agente *Sheik* demanda informação do agente *Erudite*

A mensagem contém os dados da classe *PersonProfile* que para o Protégé significa as instâncias desta classe. Estes dados correspodem aos slots da classe, como nome, local que trabalha e as palavras-chave capturadas pelo *KEA*. Na resposta o agente Erudite envia o mesmo conjunto de dados, ou seja, pessoa ou pessoas por palavra-chave encontrada.

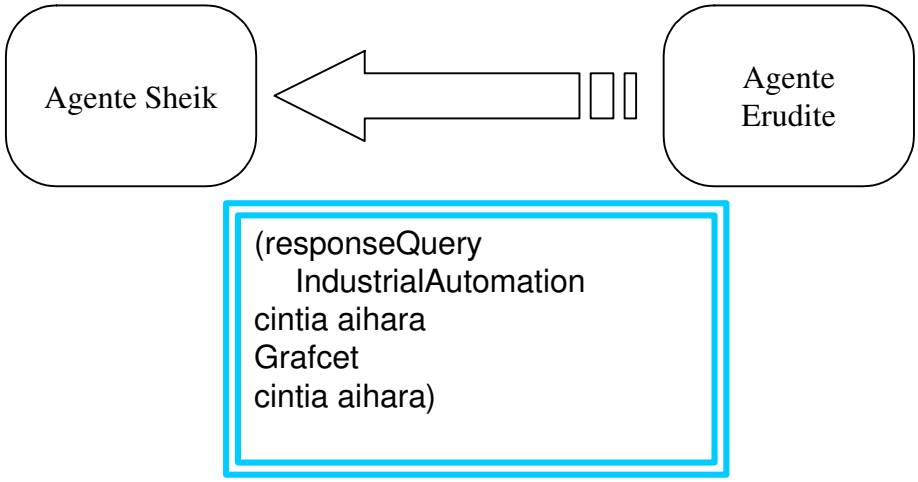


Figura 6.2 O agente *Erudite* responde à demanda

A vantagem do sistema se baseia na restrição que diz que usuários não atualizam suas informações nem quando necessário. Principalmente se o sistema não foi feito por ele. Para vencer a barreira o sistema tem que se mostrar eficiente e não invasivo e só então o usuário médio (usuários de sistemas complexos do tipo CAD, CAM, ERP, Planejamento da Produção, etc).

Durante os testes de comunicação do sistema os agentes *Sheik* e *Erudite* situavam-se em máquinas distintas e separadas geograficamente. Foi utilizado o método de serialização da linguagem java para a transmissão das mensagens.

A fase atual de desenvolvimento do sistema prototipado é a de entrada de dados para evitar partida a frio, o que não permitirá portanto avaliações de desempenho quantitativo, do tipo número de requisições corretamente respondidas por unidade de tempo, mas não obstante pode-se fazer uma série de observações qualitativas que permitirão avaliar o potencial da ferramenta implementada.

Para a definição de quais são os aspectos de interesse a observar, utiliza-se uma técnica de rastrear o grau de atendimento dos requisitos do sistema (descritos no cap. 3) na implementação do protótipo.

Lista-se a seguir os requisitos que eram antevistos para o sistema, extraídos do cap. 3 e organizados segundo a decomposição conceitual em requisitos de cooperação, coordenação e comunicação:

Requisitos do Sistema

Cooperação:

g) a ferramenta deve ser leve,

h) deve ter baixa demanda de recursos,

i) ser não-invasiva,

j) integrável com outras aplicações

k) possuir interface simples.

l) deve apontar a fonte de conhecimentos,

m) auxiliando a construção de um núcleo de competências.

Coordenação:

n) deve permitir decompor o problema em unidades menores

o) estabelecer uma visão geral que permitirá coordenar as partes

Comunicação:

p) deve considerar a dispersão de recursos

q) e o desconhecimento das atividades do outro.

Antes de prosseguir convém fazer uma rápida descrição do protótipo, para facilitar a argumentação que se seguirá.

A ferramenta foi prototipada na forma de um sistema multi agentes, programado na linguagem Java e que consiste de dois tipos:

Agentes-usuário, *Sheik*, os quais desempenham um papel de interface e permitem a aquisição automática de dados do ambiente de trabalho do usuário. A configuração destes agentes é efetuada através de uma interface visual apresentada no cap. 4, nas figuras 4.12 e 4.13. Os agentes *Sheik* não se comunicam entre si mas apenas com um agente *Erudite*.

Agentes-servidores, *Erudite*, responsáveis pela interação com bases locais de conhecimentos, ordenadas de acordo com uma ontologia específica para o domínio de conhecimento dos usuários e que também obedecem a um esquema de relacionamento (meta-modelo de recomendação) entre os conhecimentos e perfis de usuários. Estes agentes recebem mensagens dos agentes *Sheik* (na sua comunidade local) e podem comunicar-se com outros *Erudite* para a resolução do problema de recomendação. A configuração, da parte de tratamento de conhecimento, dos agentes Erudite, pode ser feita através da interface visual do sistema Protégé, como visto na fig. 5.3.

Verifica-se que no sistema prototipado os requisitos a), b), d) e e) são plenamente atingidos no lado dos agentes-usuários, já que o uso da linguagem Java permite os aspectos de leveza, baixo consumo de recursos, integrabilidade e simplicidade de interface. O aspecto c), ser não-invasiva deriva do caráter automático do agente e da prévia autorização para pesquisar apenas no diretório de trabalho (fig. 4.13, item *WorkDir*). A mensuração da quantidade de recursos utilizados pelo cliente-usuário faz-se pela quantidade de memória máxima (RAM) ocupada em tempo de execução, que foi de 7,5 Mbytes, sendo que aí está incluída a memória necessária para o sistema KEA.

Do lado do agente servidor os requisitos a), b), c) e e) não se aplicam, já que não há interação direta com o usuário (que faz as consultas via agente-usuário)e normalmente o servidor pode ser maior consumidor de recursos por decisão de projeto. A ocupação de memória foi de 19 Mbytes, incluindo a ativação do sistema Protégé. A questão da integração com outras aplicações d), é plenamente atendida através do uso da linguagem Java que permite a interface com sistemas pré-existentes, como o Protege e o Prototipador de Sistema Multi Agentes e o KEA que foram utilizados.

Os requisitos h), i) e j) são atendidos pelo fato da ferramenta ser configurada na forma de um sistema multi agentes, o qual tem por característica básica a capacidade de distribuição e pelo fato da arquitetura em duas camadas proposta ser propícia à coordenação necessária.

Os requisitos f), g), e k) derivam da natureza do problema, que é a cooperação de especialistas em certa área do conhecimento. Eles são satisfeitos através do uso de uma base de conhecimento compatível com estruturas formais de conhecimento, que estão contidas na ontologia utilizada e no meta-modelo proposto.

Finalmente, cabe citar que os pesquisadores que participaram nesta fase inicial do sistema deram opiniões que, mesmo sendo subjetivas, apontam caminhos para melhoria do protótipo. Em especial foi apontado que o fato do sistema de aquisição de conhecimento, implementado no agente-usuário através do *software* KEA, ser restrito à leitura de arquivos no formato texto (*.txt*) é uma grande limitação, já que os artigos e documentos de referência normalmente estão nos formatos *doc* e *pdf* . Os artigos em postscript podem ser transformados utilizando-se uma primitiva do ghost script, pstotext, disponível gratuitamente e os textos em Látex estão em formato .txt. Quanto às interfaces visuais e facilidade de uso a opinião é que elas são adequadas e suficientes.

Assim, neste primeiro instante de avaliação do protótipo, o parecer é de que a solução implementada mostrou-se adequada. A eficácia em se reunir pessoas segundo palavras-chave e inferir que essas palavras-chave são representativas de uma área do conhecimento deriva de uma das premissas que é a utilização de um Domínio específico, se bem que amplo, do conhecimento, no caso a área de Manufatura. Esta restrição aumenta a eficiência do algoritmo de máquina de aprendizado, fazendo com que se restrinja o relacionamento entre palavras-chave com área ou subáreas e com pessoas.

6.4 Contribuições

As seguintes contribuições foram feitas com este trabalho:

– Sistema de recomendação com uso de ontologias para classificar conhecimento;

– Os sistemas multi-agentes mais máquina de aprendizado como forma de adquirir palavras-chave relacionadas com uma mesma área do conhecimento,

– Relacionar conhecimento não-estruturado ao conhecimento estruturado.

– Desenhar um sistema de recomendação utilizando agentes, ontologias e máquina de aprendizado.

– Desenhar uma arquitetura baseada em restrições de coordenação, cooperação e comunicação.

6.5 Conclusão

Neste capítulo foram mostrados diversos sistemas diferentes, com diferentes graus de complexidade na sua implementação e mesmo tamanho do público usuário e participante. Como pode ser visto, os sistemas vieram adquirindo mais flexibilidade principalmente no respeito à questão da manutenção. O quesito manutenção do sistema é crítico e muitas vezes deixados para o administrador do sistema ou para o usuário. O que ocorre, comprovado por inúmeros trabalhos, é que o sistema não é atualizado devido ao fato que o usuário, quando não contumaz do sistema, não atualiza seus dados.

A resistência do usuário se deve ao fato de que este tipo de sistema, apesar de projetado para ele, não é o que ele utiliza no seu dia a dia como instrumento de trabalho. Para que ele seja incorporado ou ele teria que ter sido desenvolvido por ele usuário ou ele, o sistema, tem que mostrar sua eficiência sem dar trabalho extra ao usuário.

O sistema proposto tem esta finalidade. Além de se propor a ser um instrumento de compartilhamento de conhecimento usa um método que tira a preocupação do usuário em realizar a manutenção do sistema. A maneira como realiza este trabalho, com a distribuição de classes de programas, os agentes, que tem como característica principal a pró-atividade, para cumprirem um papel de representantes do usuário.

O ambiente de desenvolvimento *Protégé* tem uma característica à parte. Este sistema concede interoperabilidade ao sistema, na medida que gera e lê arquivos de diferentes formatos e está integrado aos mais diversos aplicativos da área de aquisição de conhecimento.

O algoritmo KEA, por sua vez, traz a característica de aprendizado necessária ao sistema para que ele evolua e não se perca. De modo que as informações geradas pelo KEA são mantidas pelo agente e utilizadas para sintonizar as novas palavras-chave em novos documentos do usuário. Esta dinâmica faz com que o sistema se atualize e cumpra o seu papel junto ao usuário, sendo uma ferramenta de tamanho pequeno, robusta e eficaz.

O próximo capítulo traz as conclusões gerais do trabalho e também sugere outros trabalhos nessa área de aquisição de conhecimento.

Capítulo 7

Conclusão e Propostas Futuras

Este capítulo conduz ao fechamento deste trabalho ressaltando os resultados alcançados e possíveis benefícios trazidos pelo uso conveniente da tecnologia adequada. Outro aspecto diz respeito aos trabalhos podem ser originados a partir deste, ou usando os métodos e metodologias aqui desenvolvidas e utilizadas.

Este trabalho procurou criar conceitos e vocabulários tendo sido desenvolvido no sentido de fornecer uma ferramenta, porque considerou-se que o uso de ferramentas é mais efetivo para a introdução de uma metodologia que a própria metodologia. O objetivo foi trazer para um público alvo uma nova metodologia e novas técnicas, empregando-se uma ferramenta que auxiliasse neste propósito.

Cada fase no ciclo de vida de um produto possui, ela própria, sua cadeia de suprimentos. Cadeia de suprimentos aqui é vista de uma maneira ampla, considerando parceiros comerciais mas principalmente tecnológicos, os fornecedores de conhecimento. Na fase de concepção do produto, a cadeia de suprimentos é formada principalmente por centros fornecedores de tecnologia e pesquisa. Da própria empresa ou mesmo de fora. Do próprio país ou não. Essa cadeia varia conforme o produto e não obrigatoriamente é formada à priori por empresas de longo contato. Mas deve ser de confiança.

O mesmo acontece com as pessoas que participam desta cadeia. Distribuídas geograficamente por empresas ou centros de pesquisa diversos, não supõem um longo contato mas pressupõem confiança.

A confiança é um dos habilitadores da cooperação. É nesse fator que se baseia uma grande parte dos sistemas de recomendação. As pessoas que fazem as recomendações são reconhecidas

na área de conhecimento. Confiança também na ferramenta. A ferramenta deve corresponder à expectativa que se tem dela, em termos de confidencialidade e eficácia em cumprir seus objetivos.

Neste trabalho procurou-se acompanhar diversos paradigmas: projeto em fase de concepção de produto, aquisição de conhecimento, identificação de comunidades-de-prática, identificação de vocabulários. Estes paradigmas estão intimamente relacionados entre si e elegeu-se a área de Manufatura para ser o conjunto chave onde seria trabalhado o exemplo do quadro de referência a ser desenvolvido.

Concluindo este trabalho, os próximos itens descrevem como foi alcançado o objetivo acima em cada um dos capítulos anteriores. Recomendações para o desenvolvimento de trabalhos futuros são também apresentados neste item.

7.1 Metodologia para formação de rede de habilidades

As pessoas formam rede de habilidades quando seus interesses coincidem tendo ou não perfis semelhantes. O que foi caracterizado aqui é uma metodologia de aquisição de conhecimento com a conseqüente visualização do vocabulário usado por determinado grupo e que grupo é esse. A metodologia envolve o uso de técnicas para adquirir os vocabulários que são próprios de comunidades específicas e relacioná-los com áreas do conhecimento descritas formalmente. Regras de relacionamento são estabelecidas de maneira a ser capturado os grupos mais diversos e também o grau de relacionamento entre as pessoas e as palavras. A contribuição que este trabalho traz é, além de estabelecer a seqüência a ser obedecida por um modelo que quer descrever de maneira formal o conhecimento informal de pessoas, o caráter exploratório para a engenharia mecânica, em particular para o projeto mecânico, que outros domínios do conhecimento podem trazer de contribuição para seus próprios objetivos.

O Capítulo 1 descreveu os objetivos do trabalho e o que era observado como benefícios. Identificando possíveis áreas de uso e também as dificuldades que o assunto trazia. A maior parte deste capítulo se baseia em fontes que tratam sobre inovação, compartilhamento de conhecimento, dificuldades vividas por equipes, empresas de diversos portes e em vários países. Essa foi a fonte de inspiração para o trabalho, onde a solução do problema proposto foi desenvolvida a partir deste contexto.

Apesar da seqüência dos capítulos o capítulo 3 é cronologicamente anterior ao 2. Isto porque para realizar o levantamento sobre ontologias, sistemas multi-agentes e máquinas de aprendizado utilizados no sistema, foi feito um levantamento dos requisitos. Os requisitos, eles também foram selecionados utilizando-se critérios essenciais para sistemas computacionais para trabalho cooperativo: cooperação, coordenação e comunicação. Foi utilizando-se as qualidades que um sistema deste deveria possuir que foi feito o levantamento sobre o estado da arte do setor, obedecendo as restrições que tal sistema deveria satisfazer. Outro passo importante foi selecionar entre as ferramentas computacionais que implementavam as metodologias e métodos quais as mais adequadas para serem utilizadas e quais deveriam ser implementadas.

No Capítulo 2 procurou-se estabelecer o Estado da Arte em metodologias e métodos a serem utilizados e em quais teorias estavam baseados tais métodos. Procurou-se esclarecer a terminologia e a teoria, de uma maneira resumida, porém que trouxesse os porquês básicos da escolha dos sistemas. Por exemplo, porque se utilizam *frames*, *slots*, e não atributos, parâmetros, já que classes eram comuns a ambos os sistema. Existem razões históricas para que tal ocorra e uma comunidade inteira, a de inteligência artificial utiliza estes termos como vocabulário corrente. Outro ponto importante foi a necessidade de trabalhar com formatos de armazenamento padrão e não só isso, mas formatos passíveis de serem recuperados por programas automáticos e integradas as suas informações.

RDF e XML ocupam cada vez mais este espaço uma vez que sua utilização, nas páginas da internet, está se fortalecendo. Entidades que congregam empresas interessadas, entidades de pesquisa ou financiadoras de pesquisa (universidades, centros governamentais da comunidade européia, governo americano e outros), a comunidade geradora de inovação para internet, o W3C, tem ocupado tempo, dinheiro e cérebros no desenvolvimento de um padrão. O tempo maior é sempre gasto nos acordos mais que nas variações metodológicas ou técnicas.

O Capítulo 2 trouxe também a tecnologia de agentes como vista pelos especialistas da área e suas aplicações. A busca por aplicações diversificadas tem intenção de demonstrar ao leitor as possibilidades da tecnologia e porque ela foi escolhida, já que outras poderiam ser igualmente adequadas, teoricamente. Assim como também fica visível o que significa em termos de negócios os investimentos em ontologias e seus formatos.

O domínio Inteligência Artificial trouxe para este trabalho a técnica necessária para serem cumpridas as metas de aprendizado e aquisição de informação. As redes Bayesianas se revelaram adequadas ao problema trabalhando com probabilidade de ocorrência e semelhança. O algoritmo se revelou leve, significando que a implementação é econômica em termos de memória, requisito essencial já que o algoritmo é executado na máquina do usuário. Agentes e ontologias também são ditas como pertencentes ao domínio da IA porém estão ligados a diversas áreas já tendo espalhado seus domínios. A Ontologia tem sua origem na Filosofia porém, com algumas modificações, está sendo utilizada por diversas outras áreas.

Com esses três domínios do conhecimento foi desenhado o sistema. Foi utilizado o ambiente Rational Rose para um detalhamento do sistema, mesmo se trabalhando com as restrições que o sistema impõe. Na verdade o que ocorre é que não existe um mapeamento direto entre o Rational e a Orientação a Objetos e a metodologia Orientada a Agentes e a ontologia. Mas insistiu-se neste uso devido em primeiro lugar à linguagem e o seu conhecimento tanto por implementadores de sistemas quanto por usuários. É a língua franca da engenharia de *software*, linguagem muito divulgada e utilizada, o que facilita quando da implementação do sistema por equipes diversas, com diferentes tipos de formação e mesmo quando o sistema vai ser explicado para outros. A documentação do sistema foi realizada utilizando-se este sistema.

Os Capítulos 4 e 5 trouxeram a implementação dos sistemas utilizando-se os ambientes de desenvolvimento específicos para cada um enquanto o capítulo 6 oferecia uma visão integrada do trabalho. O ambiente Protégé foi uma descoberta, já que foram utilizados outros sistemas antes dele. Facilidade de uso, tutorial para auxiliar na compreensão da ferramenta, e todas os recursos extras proporcionados por uma legião de usuários/implementadores além do suporte oferecido por seus implementadores. Todos os sistemas utilizados neste trabalho são disponíveis publicamente, código inclusive, possibilitando que o sistema seja reproduzido e mesmo modificado, personalizado, como queira um desenvolvedor ou usuário.

Cada capítulo traz sua contribuição para o quadro de referências para sistemas de aquisição de conhecimento através de vocabulário. O Capítulo 6 faz a avaliação do sistema em termos da sua interação com o usuário e com os sistemas integrados como o Protégé e o KEA. O Capítulo traz também sistemas que têm objetivos semelhantes ou usam o mesmo sistema. Foi realizada

uma comparação entre os diversos sistemas com o objetivo de facilitar o reconhecimento das qualidades que são mais adequadas para uma determinada aplicação.

No trabalho aqui desenvolvido privilegiou-se a pró-atividade do sistema, considerando-se que o usuário não tem boa-vontade para responder questionários, atualizar dados na medida que se faz necessário, tampouco compartilhar informações se o ambiente não é o de colegas da mesma área. Enfim, se o sistema não é da sua área direta de atuação nem foi desenvolvido por ele, significa que o usuário, de uma maneira geral o olhará com desconfiança. E isto é completamente normal mesmo para pessoas da área de computação que tem esse *métier* de experimentação de ferramentas computacionais como atribuição.

Plataformas para a aquisição de conhecimento tocam em pontos particularmente sensíveis como o conhecimento da pessoa e mencionando o fato que este conhecimento é o bem mais valioso de uma organização. E a pessoa não é a organização por mais que "vista-a-camisa" da empresa. De modo que o sistema implementado, apesar de beneficiar a empresa indiretamente, não tem esse primeiro objetivo. O alvo primário é a pessoa e a troca de conhecimento entre pessoas.

Este trabalho apresentou uma arquitetura baseada em agentes para a solução de problemas de aquisição de conhecimento. O sistema é baseado em classe de agentes diferentes, tendo sido projetado um modelo em camadas de modo a alcançar o nível de exigência conhecido dos usuários. O sistema é automático, ou seja, sua manutenção acontece na medida que o usuário recebe, escreve, carrega novos documentos para sua máquina. O sistema é pessoal e tem como aprimoramento do conhecimento do usuário. O sistema é pró-ativo, na medida em que ele próprio adquire as palavras-chave do usuário e realiza uma questão (*querie*) sem necessidade de intervenção do usuário. O sistema utiliza uma ontologia baseada no sistema Protégé como base de conhecimento do agente-servidor e, por extensão, dos usuários.

A solução, como foi visto no Capítulo 6 os e mesmo nos itens anteriores, pode ser ampliada, melhorada, modificada, por que é um experimento passível de repetição e seu código é aberto.

A ferramenta encoraja o conhecimento que as pessoas podem ter uns sobre os outros no domínio da Manufatura, da qual o Projeto e concepção Mecânica faz parte. A Manufatura reúne

muitas especialidades de modo que as classificações em termos de Manufatura abrem opções para a confecção de um produto. Encontrar pessoas com interesse e vocabulário semelhante pode auxiliar na hora de escolher parceiros, pedir conselhos, melhorar o conhecimento sobre determinado assunto. Respeitar o usuário no seu desejo de não ser importunado foi uma restrição imposta ao sistema que se procurou atender adequadamente.

A relevância deste tipo de pesquisa aparece principalmente quando estão sendo procurados especialistas para compor um novo projeto, ou melhorar um processo. Mas tanto o processo quanto as ferramentas podem ter seu escopo de aplicação ampliado. Não é difícil não saber o escopo do conhecimento do colega. Também não implica que mesmo sendo conhecido o parceiro, o usuário se sinta impelido a procurar o colega. Ou que a questão (*querie*) feita pelo agente seja exatamente o que queria o usuário.

De qualquer maneira o uso de uma ferramenta como a apresentada indica a boa vontade em se conhecer possíveis parceiros, em compartilhar conhecimento (mesmo sendo de palavras-chave, que são muito importantes em um sistema de classificação). Indica também a vontade de conhecer os possíveis participantes de uma comunidade-de-prática.

Finalmente, este tipo de ferramenta pode fornecer informação valiosa, que poderia ser difícil de ser obtida por meios convencionais e mesmo no tempo em que ela retorna.

7.2. Trabalhos Futuros

Várias direções podem ser indicadas para a continuação deste trabalho ou mesmo para trabalhos relacionados a este. Algumas áreas se destacam nesta ampliação, como será visto a seguir.

r) o sistema poderia ser ampliado sem grande esforço para contemplar a inclusão de artigos ou *sites* da internet. Para ampliar este escopo, sem utilizar recomendações de outros usuários, as pesquisas podem ser feitas a partir da seleção das palavras em *sites* científicos, como *Citeseer*, ou na seção de publicações de universidades.

s) Outra característica seria adicionar mobilidade ao usuário, ou seja, o agente do usuário seria um agente cuja mobilidade é determinada pela do usuário. Para reconhecer um servidor o agente deveria ter o endereço dos agentes da federação atualizado. O agente-servidor encaminharia para o agente adequado.

t) A extensão da ontologia para contemplar endereços da internet como fontes de informação para identificação de novos vocábulos ou como extensão de Domínio. Ao invés de colocar a taxonomia como parte integrante da ontologia, a referência é realizada como uma URI. Isto torna a ontologia mais enxuta e com mais possibilidades de integração.

u) O uso de RDF para formato do arquivo da ontologia. Ou OWL, Ou DAML-OIL. O problema seria escolher ou preparar possibilidade de vários formatos conforme a integração dos dados exigisse.

v) Ampliar os eventos relacionados à atualização de informação como por exemplo os tipos de aplicativos que o usuário usa (ambiente de desenvolvimento Java, Rose, simuladores, etc). O agente poderia ter uma classe que seria adaptativa ao sistema operacional, onde este tipo de informação é armazenada (arquivo *Recent* do Windows 2000, por exemplo). Hoje a atualização de informação é feita com base nas datas dos arquivos tipo doc, pdf, os e txt.

w) Acesso ao arquivo do *History*. O sistema deve evitar trazer informações repetidas. Mas como saber se o usuário quer a informação antiga senão possibilitando que este atributo seja configurado pelo usuário? Este arquivo pode ser acessível via interface. Uma chamada ao agente-usuário e o *History* poderia ser uma das possibilidades de visualização.

x) Possibilidade de inclusão do sistema em um outro, de Gerenciamento de Conhecimento. Devido ao uso de padrões algumas facilidades são previstas quando da inclusão deste sistema em outros com o mesmo propósito geral.

Apêndice 1

Metodologia de Programação Orientada a Agentes

A metodologia definida por Wooldridge, Jennings e Kinny em (Wooldridge-a, 2000, e Wooldridge-b, 1999) é a utilizada aqui para a fase de especificação do sistema de agentes onde foi desenhada a arquitetura implementada.

A metodologia é fundamentada na concepção de organização computacional onde um sistema multi-agente consiste de vários papéis interagindo. Segundo os autores a metodologia é adequada para sistemas que:

a) fazem uso significativo de recursos operacionais

b) têm como proposta maximizar a qualidade global (que pode significar que algum dos sistemas trabalhe em condições sub-ótimas)

c) Agentes heterogêneos, ou seja, diferentes agentes sendo implementados em diferentes linguagens de programação, arquiteturas e técnicas. A plataforma de implementação não é especificada.

d) Onde a estrutura de organização do sistema é estática, ou seja o inter-relacionamento dos agentes não muda com o tempo

e) A habilidade dos agentes e os serviços fornecidos são pré-determinadas

f) O sistema completo não possui vários diferentes tipos agentes

A metodologia tem a configuração esquematizada pela Figura A.1, onde os conceitos são divididos em duas fases: abstrata, usada durante a fase de análise e concreta, usada na fase de projeto.

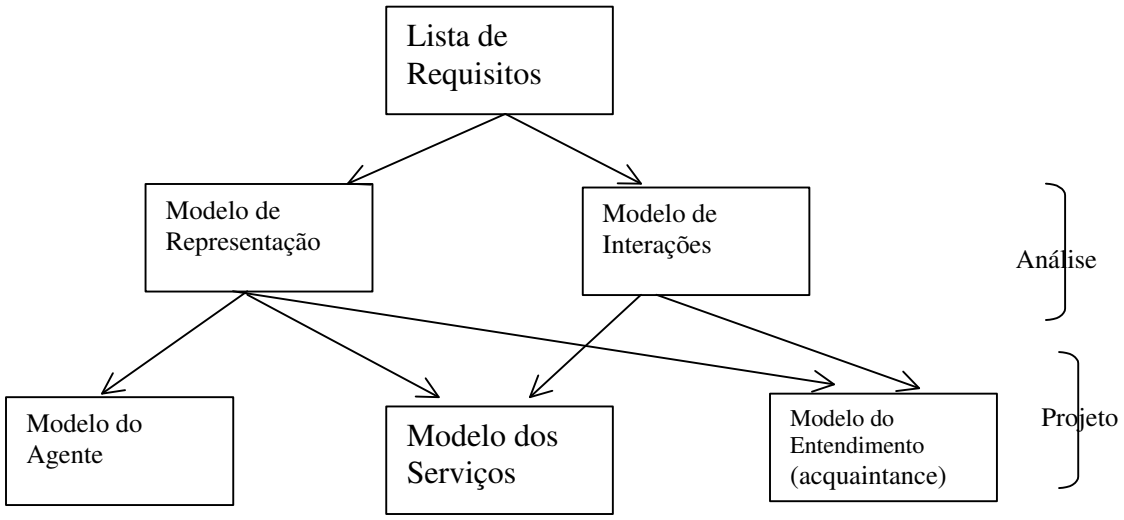


Figura A.1 Relacionamento entre os modelos

Fase de Análise - A fase de Análise tem como objetivo desenvolver a compreensão do sistema e sua estrutura. Proporciona também a compreensão de como o sistema é organizado. A entidade mais abstrata é o sistema que no caso dos agentes pode significar a "sociedade" ou "organização". O próximo nível na hierarquia é o papel definido por sua vez por quatro atributos: responsabilidades, permissões, atividades, e protocolos.

Tabela A.1 Conceitos Abstratos no modelo

Sistema – utilizado aqui com idéia de sociedade. O nível mais alto na abstração hierárquica.
Papel – Significa a representatividade na sociedade.
Responsabilidades – Atributo chave. Determina a funcionalidade
Comportamento – informações sobre o estado do sistema
Segurança – informações sobre condições de trabalho adequadas
Permissões – identifica a disponibilidade de recursos relacionada a um papel
Atividades - são ações relacionadas ao comportamento interno do agente
Protocolos – define o modo de interação com outros papéis

Adicionalmente, o atributo de Comportamento possui seu próprio conjunto de operadores para expressar o comportamento do agente. São eles:

Tabela A.2 Operadores para Comportamento

x.y	x seguido de y
x*	x ocorre 0 ou mais vezes
x^{ω}	x ocorre infinitamente
x ‖ y	x e y intercalados
x | y	x ou y ocorrem
x+	x ocorre 1 ou mais vezes
[x]	x é opcional

O modelo de interação representa o relacionamento existente entre os diversos papéis e é identificado por definições de protocolos. Os protocolos consistem dos seguintes atributos:

- Objetivo: texto descrevendo a natureza da interação;
- Iniciador: o papel responsável por iniciar a interação;
- Respondedor: o papel segundo o qual o iniciador interage
- Entradas: informação usada pelo papel iniciador enquanto desempenhando o protocolo;
- Saídas: informação fornecida pelo respondedor do protocolo;
- Processamento: descrição textual breve de qualquer processamento que o iniciador realize durante a realização da interação.

Fase de Projeto – A fase Projeto tem como objetivo transformar os modelos abstratos obtidos durante a fase de análise em modelos implementáveis. O modelo diz respeito a uma sociedade de agentes que cooperam para atingir um determinado objetivo. O modelo identifica os tipos de agentes as instâncias de agentes que podem ser inferidos destes tipos e o modelo de entendimento (acquaintance), que documenta as linhas de comunicação entre diferentes agentes.

A fase Projeto contém três modelos:

– o modelo Agente, que identifica os tipos de agentes que permeiam o sistema e as instancias de agente que concretizam estes tipos

– o modelo Serviços que identifica os serviços principais necessários para realizar o papel do Agente. O serviço corresponde a um método em termos de Orientação a Objetos. Entradas, saídas, pré-condições e pós-condições devem ser identificados.

– O modelo Entendimento que documenta as linhas de comunicação entre diferentes agentes. Indica se existe um caminho de comunicação, identificando potenciais gargalos. O modelo é representado por um grafo onde os agentes são representados pelos nós e os arcos indicam o caminho de comunicação.

Referências Bibliográficas Capítulo 1

Bernes-Lee, Tim, Handler, James, Lassila, Ora. The semantic web. Scientific American. Vol 284. Nº 5. Pp34-43. Maio 2001.

Booch, Grady. Object-Oriented Analysis and Design with Applications. 2ª Ed. Redwood City, CA. Benjamin/Cummings. 1994.

Deloitte Research. Collaborative Knowledge Networks. Deloitte Consulting and Deloitte & Touche LLP. All rights reserved. ISBN 1-892383-97-7 .2001. Disponível www.dc.com (link revisitado em dezembro 2002).

DSE – Distributed System Engineering http://cec.to.alespazio.it/DSE/home.htm

Figueiroa, Edgar, Conceição, Pedro, "Rethinking the innovation process in large organizations: a case study of 3M". In Journal of Engineering and Technology Management 17 (2000) pp 93-109. Elsevier Science 2000.

MakeIt- Management of knowledge using integrated tools for SMEs ESPRIT Project-No. 25734 http://www.pr-steyr.ac.at/makeit/. 2000

Nihitilä, Juka. R & D – Production integration in the early phases of new product development projects. Journal of Engineering and Tecnology Management JEY-M. Vol 16. Pp 55-81. 1999.

Radeke, Elke, Korzonnek, Jorg. Distributed Information Management in Virtual Engineering Enterprises by GEN RIDE-VE99 International Conference on Research Issues on Data Engineering - Virtual Enterprises, Sydney 23.-24. Mar. 1999 http://www.c-lab.de/genial/Papers.html ESPRIT Project-No. 25734

Shith, Reid, Farquhar, Adam. The road ahead for knowledge management. AI Magazine. Vol. 21. Nº4. 17 páginas. 2001.

Smith, Reid G., Farquhar, Adam. The Road Ahead for Knowledge Management AI Magazine. V2, I4, P17, Winter 2000.

Wasko, M. McLure, Faraj S.. “Is is what one does: why people participate and help others in electronic communities of practice. Strategic Information Systems. Vol. 9. Pp 155-173. 2000.

Referências Capítulo 2

Bernes-Lee, Tim, e Lassila, Ora. The Semantic Web. Scientific American. Maio 2001.

Bothorel, Cecile. Systeme multi-agents pour la organization de communautés d'interets dynamiques et distribuees. These de doctorat presentee au LAAS-CNRS. Rapport LAAS nº 99468. 1999.

Champin, Pierre-Antoine. RDF Tutorial. http://www710.univ-lyon1.fr/~champin/rdf-tutorial/. 2001

Chandrasekaran, B., Josephson J. R. and Benjamins V. R., Ontology of Tasks and Methods, Banff Knowledge Acquisition Workshop. 1998

Chaudri, V. K.et al. OKBC: A programmatic foundation for knowledge base interoperability. Anais do AAAI'98. Madison. USA. Pp 600-607. 1998.

Etizioni, O. e Selberg, E. The MetaCrawler Architecture for Resource Aggregation on the Web IEEE Expert. Vol. 12(1):8-14, 1997. http://www.cs.washington.edu/homes/etzioni/papers/ieee-metacrawler.html

Farqhuar, Adam. Ontololingua Tutorial. http://ksl-Web.stanford.edu/people/axf/tutorial.pdf , 1997.

Farquhar, Adam, Fikes, Richard, Pratt, Wanda, Rice, James. Collaborative Ontology Construction for Information Integration. http://ksl.stanford.edu/KSL_Abstracts/KSL-95-63.html 1995.

Fensel et al. On-To-knowledge Final report. Outubro de 2002. Disponível http://www.ontoknowledge.org/downl/del43-new.pdf (link revisitado em dezembro 2002).

Fensel, Dieter et al. OIL in a nutshell. Knowledge acquisition, modeling and management. Anais

do European Knowledge Acquisition Conference (EKAW 2000). Lecture Notes in Artificial Intelligence. Outubro 2000.

Fensel, Dieter et al. Ontology Languages. http://www.ontolnowledge.org/downl/del1.pdf. 2000.

Flett, Alan e Brown, Mike. Enterprise-standard ontology environment for intelligent e-business. Anais do IJCAI workshop em Ontologies and Information sharing. Seattle, USA. 4 e 5 de Agosto de 2001.

G. Denker, et al. Accessing Information and Services on the DAML-Enabled Web. Apresentado no Second International Workshop Semantic Web. 2001 Disponível: http://www.daml.org/services/SemWeb01-SRI.pdf (revisitado em dezembro de 2002).

Genesereth, Michael, Fikes, Richard. Knowledge Interchange Format version 3.0 Reference Manual. Technical Report, Logic-92-1, Computer Science Dept., Stanford University. http://www.cs.umbc.edu/kse/. 1992.

Gilbert Don, Aparicio, Manny, Atkinson, Betty, Brady, Steve, Ciccarino, Joe, Benjamin Grosof, Pat O'Connor, Damian Osisek, Steve Pritko, Rick Spagna, and Les Wilson, White paper on intelligent agents. IBM Corporation, Research Triangle Park, NC.

Guarino, Nicola, Giaretta, Pierdaniele. Ontologies and Knowledge Bases Towards a terminological clarification. In N. Mars (ed.) Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing. IOS Press, Amsterdam: 25-32. 1995. http://www.ladseb.pd.cnr.it/infor/Ontology/Papers/OntologyPapers.html

Hendler, James, McGuinness, Deborah. The DARPA Agent Markup Language. IEEE Intelligent Systems Trends and Controversies. 2000.

Hong, Se June, Weiss, Sholom. Advances in Predictive Data Mining Methods. MLDM'99. Lecture Notes in Artificial Intelligence. Pp 13-20. 1999.

http://lide.uhk.cz/home/fim/ucitel/fumikup1/www/dIZS/Temata/Chandra.pdf

HTTP://www.dstc.edu.au/Research/Projects/rdf/RDF-Idiot.html

http://www.xml.com/

Hugin Expert. White Paper. www.hugin.com. (link revisitado em dezembro de 2002).

Izumi, Noriaki e Yamaguchi, Takahira. Building business application by integrating heterogeneous repositories based on ontologies. Anais do IJCAI workshop em Ontologies and Information sharing. Seattle, USA. 4 e 5 de Agosto de 2001.

Jennings N. R. "On Agent-Based Software Engineering" Artificial Intelligence, 117 (2) 277-296. 2000.

Karp, Peter D., The design space of frame knowledge representation systems. May 1993. http://www.ai.sri.com/pubs/

Klush, Mathias. Information Agent Technology for the Internet. Data and knowledge engineering. Vol. 36. Pp337-372. 2001.

Maes, Pattie. General tutorial on software agents. 1997. Disponível http://pattie.www.media.mit.edu/people/pattie/CHI97/ (link revisitado em dezembro de 2002).

Maes, Pattie. Reflections of Agents that reduce work and information overload. Reading in Intelligent User Interfaces. Morgan Kaufman Press. ISBN 1-55860-444-8. 1998.

McGuinness Deborah L., Fikes Richard, Rice James, e Wilder Steve, ``The Chimaera Ontology Environment," In the Proceedings of The Seventeenth National Conference on Artificial Intelligence (AAAI 2000), Austin, Texas. 30 de julho a 3 de agosto . 2000.

McGuinness, Deborah. Ontologies Come of Age. Dieter Fensel, Jim Hendler, Henry Lieberman, e Wolfgang Wahlster, editores. The Semantic Web: Why, What, and How. MIT Press. 2001.

Middleton, Stuart. Interface agents: a review of the field. Technical Report Nº: ECSTR-IAM01-001. ISBN: 0854327320. University of Southampton. 1999. Disponível http://www.ecs.soton.ac.uk/~sem99r

Nardi, Daniele, Brachman, Ronald. An Introduction to Description Logics. Retirado do The Description Logic Handbook Theory, Implementation and Applications. Edited by F. Baader, D. McGuinness, D. Nardi, P. P. Schneider. Cambridge University Press. http://www.cs.man.ac.uk/~franconi/dl/course/dlhb/dlhb-01.pdf. 2002. (link revisitado em dezembro de 2002)

Nwana, Hyacinth S.. Software agents: an overview. Knowledge Engineering Review. http://citeseer.nj.nec.com/nwana96software.html. 1996.

OilEd http://oiled.man.ac.uk/

Ouellet, Roxane. Introduction to DAML. http://www.xml.com/daml. 2002.

Perez, Assunción Gomez. A survey on ontology tools. Project IST 2000-29243. May 2002. http://ontoWeb.aifb.uni-karlsruhe.de/About/Deliverables/D13_v1-0.zip

Petrie, Charles, Goldmann, Sigrid, Raquet, Andreas. Agent Based Project Management. Lecture Notes in AI - 1600, Springer-Verlag, 1999.

Petrou, M. Leaning in pattern recognition. MLDM'99. Lecture Notes in Artificial Intelligence. Nº 1715. Pp1-12. 1999.

Picard, Rosalind. Toward agents that recognize emotion. Actes Proceedings IMAGINA. Pp153-165. March 1998. http://www.media.mit.edu/~picard. (link revisitado em dezembro de 2002).

RDF - Resource Description Framework Model and Syntax Specification. W3C Recommendation 22 February 1999. http://www.w3.org/TR/1999/REC-rdf-syntax-19990222.

Recticular Systems. Agent Builder. http://www.agentbuilder.com. 1999.

Schreiber, G. et al. CommonKADS: Comprehensive methodology for KBS Development. IEEE Expert. Vol. 9. Nº 6. 1994.

Sebastiani, Fabrizio. Machine Learning in Automated Text Categorization. ACM Computing

Surveys. Vol 34. Nº 1. Pp 1-47. Março 2002.

Sowa, John F.. Ontology http://www.jfsowa.com/ontology/index.htm. Agosto 2001.

Staab, Steffen et al. Engineering ontologies using semantic patterns. Anais do IJCAI workshop em Ontologies and Information sharing. Seattle, USA. 4 e 5 de Agosto de 2001.

Sycara, Katia et al. Distributed Intelligent Agents. IEEE Expert. Dezembro 1996.

Ushold, Mike, King, Martin, Moralee, Stuart, Zorgios, Yannis. The Enterprise Ontology. Enterprise Project Deliverable: MID 3.1, Versão 1.1. 1995.

W3C Feature Synopsis for OWL Lite and OWL. http://www.w3.org/TR/2002/WD-owl-features-20020729/. 2002.

Wielinga, B., Schreiber, A. T., and Breuker, J. KADS: A modeling approach to knowledge engineering.Knowledge Acquisition. Vol. 4. Nº 1. Pp 5-53. 1992.

Witten, I. H. et al. KEA: Practical automatic keyphrase extraction. Anais do DL'99. Pp 254-256.1999.

Zaychik, Vera. DAML: The DARPA Agent Markup Languages. http://edge.mcs.drexel.edu/GICL/howto/DAML/DAML_files/v3_document.htm. 2001.

Zhang, Huajie et al. The learnability of Naïve Bayes. Canadian AI 2000. Lecture Notes in Artifical Intelligence. Nº 1822. Pp 432-441. 2000.

Referências Bibliográficas Capítulo 3

Barbuceanu, Mihai, Fox, Mark. Team formation and management in an agent structured supply chain. Anais do WetIce'97. 1997

Brown, John, Duguid, Paul. Organizational learning and communities-of-practice: Toward a unified view of working, learning, and innovation. 1991. http://www2.parc.com/ops/members/brown/papers/orglearning.html (link revisitado em dezembro de 2002).

Cagliano, Rafaella, Chiesa, Vitório, Manzini, Rafaela. Differences and similarities in managing technological collaborations in research, development and manufacturing: a case study. Journal of Engineering and Technological Management Jet-M. Vol. 17.193-224. 2000.

Clark P., Thompson J., and Porter B.. Knowledge Patterns. In KR'2000 (Anais da 7 Int Conf). 591-600, Eds: A. Cohn, F. Giunchiglia, B. Selman, CA:Kaufmann, 2000. http://www.cs.utexas.edu/users/pclark/papers/kr00.ps.Z

Cohen, W.M. and Levinthal, D.A. "Absorptive capacity: a new perspective on learning and innovation", Administrative Science Quarterly, vol. 35:128–152 (1990).

Dean, Douglas , Lee, James et al. Enabling the effective involvement of multiple users: methods and tools for collaborative software engineering. Journal of Management Information Systems. Vol. 14, n° 3, pp 179-222. Winter 1997-98

Deepika Chauhan, JAFMAS: A Java-based Agent Framework for Multi-agent Systems Development and Implementation, ECECS Department Thesis, University of Cincinnati, Cincinnati, OH, 1997. http://www.ececs.uc.edu/~abaker/JAFMAS/

Dieng, Rose. Acquisition des Connaissances pour l'Assistance à la Conception par Interaction entre Agents. http://www.inria.fr/rapportsactivite/RA2001/acacia/acacia_tf.html (link revisitado em dezembro 2002).

Drira, K., Martelli, A., Villemur, T. (Eds). Cooperative Environments for Distributed Systems Engineering. Lecture Notes in Computer Science 2236. Springer Verlag. 2001. http://link.springer.de/link/service/series/0558/tocs/t2236.htm (link revisitado em dezembro 2002).

Drira, Khalil, Gouezec, F., Diaz, Michel. A cooperation service for distributed cooperative applications. Anais da 2ª Conferência em "Parallel and Distributed Computer Networks" (PDNC'98). 14-16 Dezembro em Brisbane, Austrália.1998.

Dröge, Cornelia, Jayaram, Jayanth, Vickery, Shawnee K., "The Ability to Minimize the Timing of New Product Development and Introduction:". In Journal of Product Innovation Management vol. 17 (2000) pp. 24-40.

Farquhar, Adam, Fikes, Richard, Rice, James, "The Ontolingua Server: a tool for collaborative ontology construction. http://ontolingua.stanford.edu . 1996.

Felfering, Alexander. UML as domain specific language for the construction of knowledge based systems. International Journal of Software Engineering and Knowledge Engineering. Vol. 10 n° 4 pp 449-469. 2000.

Fox, Mark, Barbuceanu, Mihai, Teigen, Rune. Agent-Oriented Supply-Chain Management. International Journal of Flexible Manufacturing Systems. Vol. 12. pp 165-188. 2000.

Frank E., Paynter G.W., Witten I.H., Gutwin C., Nevill-Manning C.G. Domain-specific keyphrase extraction. Anais do Sixteenth International Joint Conference on Artificial Intelligence, Morgan Kaufmann Editores, San Francisco, CA. pp. 668-673. 1999.

Gruber, Thomas. Toward principles for the design of ontologies used for knowledge sharing. Formal ontology in conceptual analysis and knowledge representation. Ed. Nicola Guarino, Kluwer Academic Publishers. 1993

Harmsen, Hanne, Grunert, Klaus, Bove, Karten. Company competencies as a network: the role of product development. Journal of Product Innovation Management. Vol. 17.pp 194-207. 2000. www.elsevier.com

Kazanjian, Robert K., Drazin, Robert, Glynn, Mary Ann. Creativity and technological learning: the roles of organization architecture and crisis in large-scale projects. Journal of Engineering and Technology Management Jet-M. Vol. 17. pp 273-298. 2000.

Kodama, F. Technology fusion and the new R&D. Harvard Business Review. Vol. 70 (4). Pp 70-78. 1992.

Malone, Thomas, Crowston, Kevin. The interdisciplinary study of Coordination. ACM Computing Surveys. Vol. 26(1). Pp 87-119. 1994.

Mäntylä, M., and Ranta, M. Engineering Process Ontologies for Communication, Co-operation, and Co-ordination in a Virtual Enterprise. CD-Rom do Prolamat '98, Trento, Itália. Setembro 1998.

McDonough III, Edward. Investigation of factors contributing to the success of cross-functional teams. Journal of Product Innovation Management. Vol. 17. pp 221-235. 2000

Moenaert, Rudy, Caeldries Filip, Lievens Annouk, Wauters, Elke. Communication Flows in international product innovation teams. Journal of Product Innovation Management. Vol.17. 360-377. 2000

Molina-Espinosa J.M., Nabuco O., Drira K.. A UML model for session management in collaborative design for space activities. 8th European Concurrent Engineering Conference (ECEC' 2001) Valencia, Espanha. Pp170-174. 2001.

Nabuco-a O., Drira K., e Rosário J.M.. Sharing engineering information and knowledge. contributions to Pipefa's platform. In International NAISO Congress on Information Science Innovations (ISI'2001), Dubai (EAU), 17-21 Março 2001, pp.431-437.

Nabuco-b, O., Drira, K., Dantas, E.. A layered design model for knowledge and information sharing cooperative systems. IEEE 10th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'2001), Cambridge. EUA. Pp.305-310.2001.

Nihtila, Jukka. R&D–Production integration in the early phases of new product development

projects. Journal of Product Innovation Management. Vol.16. pp. 55-81. 1999.

Nwana, H. S., Lee, L., Jennings, N. R. Coordination in software agent systems. Journal of BT Technologies. Vol. 14. Nº 4. Outubro 1996.

Petrie, Charles. Agent based software engineering. Lecture Notes in Artificial Intelligence. Springer-Verlag. Nº1957. 2001

Petrie, Charles. ProcessLink Coordination of Distributed Engineering. http://www-cdr.stanford.edu/ProcessLink/papers/osaka/J-PLink.ps (link revisitado em dezembro 2002)

Stark, John. Principles of good product development. 2000.http://www.johnstark.com/prn08.html (link revisitado em dezembro de 2002).

Sycara, Karia, Lu, Jianguo, Klush Matthias, Widoff, Seth. Matchmaking among heterogeneous agents on the Internet. Anais do 99'AAAI Spring Symposium on Intelligent Agents in Cyberspace. Março 1999.

Vadapalli, Anand, Mone, Mark. Information technology project outcomes: user participation structures and the impact of organization behavior and human resource. Journal of Engineering Technology Management Jet-M. Vol 17. pp 127-151. 2000.

Wilhem, Georg, Peter Woyzesche, Erni Karin. Lessons learned: pattern-based redesign of an object-oriented knowledge engineering system. Journal of Object Oriented program. 3-7. Dezembro. 2000.

Wooldridge-a, Michael, Jennings, Nicholas, Kinny, David. The Gaia Methodology for agent-oriented analysis and design. In Journal of Autonomous Agents and Multi-Agent Systems. Vol 3(3):285-312. 2000.

Wooldridge-b, Michael, Jennings, Nicholas, Kinny David. A methodology for agent-oriented analysis and design. Anais da 3 International Conference on Autonomous Agents (Agents-99) Seattle. WA 69-76. 1999.

Referências Bibliográficas Capítulo 4

Booch, G. Object-oriented Analysis and Design. (Second Edition). Redwood City: The Benjamin / Cummings Publishing Company, Inc., 1994, 589p.

Koyama, M. F., Roy, D., Anciaux D., Dantas, E., Nabuco, O., Haddad, R. A Reactive Architecture Designed for Small And Medium Enterprise. 15th ISPE/IEE International Conference on CAD/CAM Robotics, and Factories of Future, Águas de Lindóia, SP, Brazil, 18-20, Aug. 1999 p. MT1-1 - MT1-6

Koyama, Mauro. Arquitetura de supervisão e controle de chão de fábrica baseada em componentes genéricos. Campinas: Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, 2001. 140p. Tese de doutorado.

Schoderbeck, C.G., Schoderbeck, P.P., Kefalas, A.G. Management systems: conceptual considerations. Dallas: Business Publications, 1980, 355p.

Ward, P., Mellor, S.. Structures Development for Real Time Systems. Englewoods Cliffs, Nj. Prentice Hall. 1985. 4 volumes.

Referências Bibliográficas Capítulo 5

Gruber, Thomas. Towards principles for the design of ontologies used for knowledge sharing. Formal ontology and conceptual analysis and knowledge representation. Nicola Guarino e Roberto Poli Editores. Kluwer Academic Publishers. 23p. 1993.

IMTI Integrated Manufacturing Technology Initiative. 21st Century Manufacturing Taxonomy A Framework for Manufacturing Technology Knowledge Management. V1.0. 12 http://www.IMTI21.org Dezembro 2000 (link revisitado em dezembro de 2002).

J.H. Gennari, M.A. Musen, R. Fergerson, & ali. The Evolution of Protégé: An Environment for Knowledge-Based Systems Development. International Journal of Human-Computer Studies, in press. http://faculty.washington.edu/gennari/#pubs

Noy, Natalya Fridman, Fergerson, Ray W., Musen, Mark A. The knowledge model of Protégé-2000: combining interoperability and flexibility. 2th International Conference on Knowledge Engineering and Knowledge Management (EKAW'2000), Juan-les-Pins, France. 2000.

Noy, N. e McGuinness, D. Ontology developement 101: A guide to create your first ontology.http://protege.stanford.edu/publications/ontology_development/ontology101-noy-mcguinness.html. 2001.

Noy, N. F., M. Sintek, S. Decker, M. Crubezy, R. W. Fergerson, & M. A. Musen. Creating Semantic Web Contents with Protege-2000. IEEE Intelligent Systems 16(2):60-71, 2001.

Witten, I.H., Paynter, G.W., Frank, E., Gutwin, C. and Nevill-Manning, C.G."KEA: practical automatic keyphrase extraction" Proc Digital Libraries '99. 254-265, Berkeley, CA, August. 1999.

Referências Bibliográficas Capítulo 6

Amaral, Daniel Capaldo, Rozenfeld, Henrique. Gerenciamento de conhecimentos explícitos sobre o processo de desenvolvimento de produto. 3º Congresso Brasileiro de Gestão de Desenvolvimento de Produto. Florianópolis, SC. Setembro 2001.

Brown, John Seely, Duguid, Paul. Organization learning and communities-of-practice: Toward a unified view of working, learning, and innovation.

Chauhan, Deepika. JAFMAS: A Java-based Agent Framework for Multiagent Systems Development and Implementation, ECECS Department Thesis, University of Cincinnati, Cincinnati, OH, 1997. 170p. http://www.ececs.uc.edu/~abaker/JAFMAS Tese (Doutorado)

Davies, John, Dukes, Alistair e Stonkus, Andrius. OntoShare: Using ontologies for knowledge sharing.

Dellarocas, Chrysanthos. The design of reliable trust management system for electronic trading communities. Working paper.

Foner, Leonard Newton. Political Artifacts and Personal Privacy: The Yenta Multi-Agent Distributed Matchmaking System. Boston: Media Arts and Sciences, Massachusetts Institute of Technology, 1999. 129p. Tese (Doutorado).

Heberman, Bernardo A., Hogg Tad. Communities of Parctice: Performance and Evolution.

Kalfoglou, Yannis et al. MyPlanet: an ontology-driven Web-based personalized news service. Anais do IJCAI-01 workshop on Ontologies and Information Sharing. Seattle, USA. Agosto 2001.

Kautz, Henry, Selman, Bart, Shah, Mehul. The hidden web. American association for artificial

intelligence. Summer 1997. p 27-36.

Konstan, Joseph et al., Applying collaborative filtering to Usenet News. Communications of ACM. Vol 40. Nº3. Março 1997.

Middleton, Stuart, Roure David De, Shadbolt, Nigel. Capturing knowledge of user preferences: ontologies in recommender systems.

Mladenic, Dunja. Text-learning and related intelligent agents: A survey. IEEE Intelligent Systems. p 44-54. 1999.

Petersen, Sobah Abbas, Divitini, Mônica. Using agents to support the selection of virtual enterprise teams. Department of Computer and Information Science, Norwegian University of Science and Technology, N-7491 Trondheim, Norway

Picard, Rosalind. Toward agents that recognize emotion. Actes proceedings IMAGINA. Março 1998. p 153-165. www.media.mit.edu/~picard

Rhodes, Bradley James. Just-In-Time Information Retrieval. Boston: Media Arts and Sciences, School of Architecture and Planning, Massachusetts Institute of Technology. 154p. 2000 Tese (Doutorado).

Schultze, U., Boland Jr, R.J., Knowledge management technology and the reproduction of knowledge work practices. Journal of Strategic Information Systems 9 (2000) 193-212.

Trigg, Randall, Bodker, Susanne. From implementation to desing: Tailoring and the emergence of systematization in CSCW. CSCW'94, Chapel Hill, NC, Outubro 1994.

Wasko, M. McLure, Faraj S. "It is what one does": why people participate and help others in electronic communities of practice. Journal of Strategic Information Systems 9 (2000) 155-173.

Zhang, Tong, Iyengar, Vijay S.. Recommender systems using linear classifiers. Journal of Machine Learning Research 2. p 313-334. Fevereiro 2002.