OLIMPO: RECOMENDAÇÃO DE CONHECIMENTO PESSOAL ATRAVÉS DE ONTOLOGIAS

Vinícios Batista Pereira

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Aprovada por:

Prof. Geraldo Bonorino Xexéo, D. Sc.

Prof. Jano Moreira de Souza, Ph.D.

Prof. Geraldo Zimbrão da Silva, D. Sc.

Profª. Ana Cristina Bicharra Garcia, Ph.D.

RIO DE JANEIRO, RJ - BRASIL

JULHO DE 2007

PEREIRA, VINÍCIOS BATISTA

Olimpo: Recomendação de Conhecimento Pessoal Através de Ontologias [Rio de Janeiro] 2007

XVII, 169 p. 29,7 cm (COPPE/UFRJ, M.Sc., Engenharia de Sistemas e Computação, 2007)

Dissertação - Universidade Federal do Rio de Janeiro, COPPE

1. Comunidades de Prática
2. Ontologias
3. Web Mining

I. COPPE/UFRJ II. Título (série)

A minha esposa Amanda, que preenche minha vida de várias formas.

Um apólogo

Io era filha do deus-rio Ínaco, que por sua vez era filho de Oceanus e Tétis. Foi sacerdotisa da Hera argiva e pertenceu à raça real de Argos. Outras versões a tratam como uma ninfa, mas Io não é associada a nenhum elemento natural como as ninfas costumam ser.

Sua beleza despertou a paixão de Zeus, que, para cortejá-la, cobriu o mundo com um manto de nuvens escuras, escondendo seus atos da visão de Hera, sua esposa. A estratégia falhou e a deusa, desconfiada, desceu do monte Olimpo para averiguar o que estava acontecendo. Numa vã tentativa de iludir sua esposa ciumenta, o deus transformou sua amante em uma belíssima novilha branca. Intrigada pelo interesse do marido no animal e maravilhada com a beleza do mesmo, Hera exigiu a novilha para si e a pôs sob a guarda do gigante Argos Panoptes. Argos, quando dormia, mantinha abertos cinqüenta de seus cem olhos.

Zeus encarregou Hermes de libertar sua amada. Para tanto, o mensageiro dos deuses, usando a flauta de Pã, pôs para dormir os olhos despertos de Argos, enquanto os outros cinqüenta dormiam um sono natural, e cortou sua cabeça. Hera recolheu os olhos de seu servo e os pôs na cauda do pavão, animal consagrado a ela.

Io estava livre do cativeiro, mas não dos tormentos de Hera. O fantasma de Argos continuava a persegui-la. Para piorar sua situação, a deusa enviou um moscardo para picar a novilha constantemente durante sua fuga.

Io perambulou de Micenas para Eubéia. Atravessou a Ilíria e subiu o monte Hemos, na Trácia. O mar, cujas praias percorreu, recebeu o nome de Mar Iônio. O Estreito de Bósforo, que liga o Mar de Mármara ao Mar Negro, cujo significado é Passagem da Vaca, foi batizado assim após Io tê-lo cruzado a nado. Atravessou a Cítia e, ao chegar ao monte Cáucaso, encontrou Prometeu acorrentado em uma rocha. O titã disse que, ao alcançar o Egito, ela seria restaurada a sua forma humana por Zeus e teria um filho. A criança seria a primeira de uma linhagem que culminaria com Hércules, que acabaria por libertar o próprio Prometeu.

Io finalmente chegou às margens do Nilo. Cansada de tanto sofrimento, implorou a Zeus por um fim. O deus comovido foi falar com Hera e ambos restauraram Io à sua forma humana. Ela teve um filho, Épafo, que foi roubado pelos Curetes sob ordens de Hera. Io recuperou o menino e reinou sobre o Egito, sob nome de Ísis e casada com Telégono. Sua coroa tinha dois pequenos chifres de ouro, lembranças de sua transformação.

Este foi um breve resumo da história contada por Publius Ovidius Naso no primeiro livro de sua obra Metamorfoses (Metamorphoseon, em latim), onde em quinze livros ele descreve a criação e história do mundo segundo o ponto de vista da mitologia greco-romana. Essa história mitológica inspirou a nomeação das entidades criadas neste trabalho.

Agradecimentos

Antes de tudo, agradeço a Deus, por ter me guiado em todos os momentos da minha vida; por me mostrar diversas vezes que fé, perseverança e realizações compõem um maravilhoso ciclo; e por sempre manter minha certeza que tudo tem um propósito positivo em nossas vidas.

A minha esposa Amanda, que desempenha vários papeis fundamentais em minha vida: esposa, amiga, parceira de trabalho, parceira acadêmica, co-orientadora, conselheira, guia espiritual, psicóloga, enfermeira, *personal trainer*, *personal stylist*, agente de turismo, *promoter*, bandinha de música, humorista, guia gastronômica, entre outros. Meu canivete suíço, que, em cada momento ao longo de nossa vida juntos, soube qual ferramenta usar para me apoiar, tranqüilizar e incentivar.

À minha família, que me ensinou que nada que seja perene vem sem esforço e merecimento. Graças a eles aprendi a ser perseverante e confiante para alcançar cada objetivo da minha vida.

Aos dois maiores professores da minha vida profissional: Edílson e professor Blaschek. Responsáveis, cada um a seu tempo, por grande parte do meu crescimento profissional. Grandes incentivadores do meu ingresso no mestrado. Obrigado pela confiança que depositaram em mim em diversas oportunidades.

Aos meus parentes mais próximos, vários tios e primos que sempre me incentivaram e acreditaram no meu potencial. Agradeço a todos, mas como são tantos não me arrisco a citar seus nomes.

Aos incontáveis amigos que fiz ao longo dos anos. Amigos de colégio, graduação, mestrado... Amigos do Rio (da capital até Bom Jardim, passando por Valença e Duque de Caxias, claro!), de Minas (de Juiz de Fora até Santos Dumont), de Salvador, do Mato Grosso, do Espírito Santo, entre outros estados... A cada um que me apoiou com longas conversas em momentos difíceis, ou aqueles que simplesmente me ofereceram vários momentos de diversão em *happy hours* ou festinhas.

Ao meu orientador Xexéo, que acreditou e defendeu a minha proposta e me deu liberdade para executá-la ao longo deste trabalho; que se mostrou um grande defensor da qualidade do trabalho aqui executado todas as vezes que foi necessário.

Ao professor Geraldo Zimbrão, que como professor e orientador do meu trabalho de projeto final de curso, mostrou-se um grande incentivador ao meu ingresso no mestrado.

A todos que contribuíram ao longo deste tempo para meu trabalho ou para a manutenção da minha sanidade mental meus mais sinceros agradecimentos.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

OLIMPO: RECOMENDAÇÃO DE CONHECIMENTO PESSOAL ATRAVÉS DE ONTOLOGIAS

Vinícios Batista Pereira

Julho/2007

Orientadores: Geraldo Bonorino Xexéo
Jano Moreira de Souza

Programa: Engenharia de Sistemas e Computação

Existem várias iniciativas na comunidade científica para produzir sistemas de gestão do conhecimento e CSCW para fins educacionais, contudo, ainda persistem os problemas quanto à troca de informações entre os aprendizes. Nessa dissertação nós apresentamos Olimpo, um sistema para auxiliar aprendizes a não apenas compartilhar conteúdos e detalhar as fontes da informação a eles necessárias, mas também para auxiliá-los a usar o conhecimento disponível. Olimpo utiliza ontologia e mineração de dados para criar cadeias de conhecimento de uma forma semi-automática, que é um trabalho que normalmente exige bastante esforço. Um módulo monitora a navegação do aprendiz pela Web. Em seguida, outro módulo classifica seu conteúdo usando uma ontologia, cria e recomenda uma cadeia de conhecimento ao aprendiz. Como um subproduto deste trabalho nós temos uma base de conhecimento com páginas da Web classificadas.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

OLIMPO: PERSONAL KNOWLEDGE RECOMMENDATION USING ONTOLOGIES

Vinícios Batista Pereira

July/2007

Advisors: Geraldo Bonorino Xexéo
Jano Moreira de Souza

Department: Systems and Computer Engineering

There are many initiatives in the scientific community to produce knowledge management and CSCW systems. However, it is difficult to promote the easy information share among learners. In this dissertation we present Olimpo, a system to help learners to share not only what the information content is, where the information is, and those who have the information needed by the learner, but also how to use the available knowledge. Olimpo uses ontologies and data mining to create knowledge chains in a semi-automatic way, which is a job that usually would take a lot of effort. A module monitors the learner's web navigation. From there, another module classifies its content using an ontology, creates and recommends a knowledge chain to the learner. As a sub-product of this work we have a knowledge base with classified web pages contents.

Índice

Capítulo 1 - Introdução 1
1.1 Motivação 1
1.2 Contexto 2
1.3 Objetivos 4
1.4 Organização dos Capítulos 5
Capítulo 2 - Recomendação em Comunidades de Aprendizado 6
2.1 Conhecimento 7
2.1.1 Aquisição do conhecimento 9
2.2 Comunidades de prática 10
2.2.1 Fatores de sucesso para criação e manutenção de comunidades de práticas 11
2.3 Comunidades de aprendizado 17
2.4 Colaboração oportunística 18
2.5 Conclusão 20
Capítulo 3 - Mineração de Dados e Estratégias de Classificação de Conhecimento .. 22
3.1 Mineração da Web 22
3.2 Mineração de Conteúdo 23
3.2.1 Pré-processamento do texto de entrada 25
3.2.2 Delimitação de segmentos textuais 26
3.2.3 Atribuição de pesos a cada palavra do segmento textual 26
3.2.4 Cálculo do peso TF-ISF de cada palavra da sentença 26
3.2.5 Cálculo da média TF-ISF para cada uma das sentenças 26
3.2.6 Cálculo da MAX média 27
3.2.7 Cálculo do limitante inferior para seleção de sentenças 27
3.2.8 Produção do extrato 27
3.3 Mineração de Utilização 28
3.3.1 Preparação de Dados 30
3.3.1.1 Filtragem dos dados 31
3.3.1.2 Identificação de usuários 32
3.3.1.3 Identificação das sessões 35
3.3.1.4 Identificação de transações 36

3.3.2 Descoberta de padrões 37
3.3.3 Análise dos padrões 39
3.4 Folksonomia 41
3.5 Conclusão 42
Capítulo 4 - O Sistema Olimpo 44
4.1 Arquitetura do Olimpo 45
4.1.1 Argus-MMN (Módulo Monitor de Navegação) 46
4.1.2 Hera-MMD (Módulo Manipulador de Dados) 47
4.1.3 Hermes-MAD (Módulo Analisador de Dados) 48
4.2 Sistema de recomendação de IO 52
4.3 Implementação do Sistema Olimpo 53
4.3.1 Desenvolvimento do Argus 53
4.3.2 Desenvolvimento de Hera 55
4.3.3 Desenvolvimento do Hermes 57
4.4 Exemplo 59
4.5 Conclusão 61
Capítulo 5 - Ontologia de Linguagem de Programação 63
5.1 Ontologia 63
5.1.1 Ontologias de Aprendizado 65
5.2 Estudo de Caso da Ontologia 66
5.3 Construção da Ontologia 68
5.4 Conclusão 68
Capítulo 6 - Avaliação do Olimpo 70
6.1 Estudo de Observação 70
6.1.1 Definição do Estudo de Observação 71
6.1.2 Planejamento do Estudo de Observação 71
6.1.2.1 Contexto 71
6.1.2.2 Treinamento 72
6.1.2.3 Projeto Piloto 72
6.1.2.4 Participantes 72
6.1.2.5 Instrumentação 72
6.1.2.6 Critérios 74
6.1.2.7 Hipótese Nula 74
6.1.2.8 Hipótese Alternativa 75

6.1.2.9 Variáveis Independentes 75
6.1.2.10 Variáveis Dependentes 75
6.1.2.11 Análise Qualitativa 75
6.1.2.12 Capacidade Aleatória 76
6.1.2.13 Classificação em Bloco 76
6.1.2.14 Balanceamento 76
6.1.2.15 Mecanismos de Análise 76
6.1.2.16 Validade Interna do Estudo 77
6.1.2.17 Validade Externa do Estudo 78
6.1.2.18 Validade de Construção do Estudo 78
6.1.2.19 Validade de Conclusão do Estudo 79
6.1.3 Execução do Estudo de Observação 79
6.1.3.1 Seleção dos Participantes 79
6.1.3.2 Instrumentação 79
6.1.3.3 Procedimento de Participação 84
6.1.3.4 Execução 85
6.1.4 Análise dos Resultados do Estudo de Observação 89
6.1.4.1 Avaliação Quantitativa 89
6.1.4.2 Eliminação de Valores Extremos 90
6.1.4.3 Teste Paramétrico 92
6.1.4.4 Análise da Execução das Tarefas 93
6.1.4.5 Análise da Precisão e Cobertura 98
6.1.4.6 Análise Qualitativa 98
6.2 Estudo Experimental 100
6.2.1 Definição do Estudo Experimental 100
6.2.2 Planejamento do Estudo Experimental 100
6.2.2.1 Contexto 100
6.2.2.2 Treinamento 101
6.2.2.3 Projeto Piloto 101
6.2.2.4 Participantes 101
6.2.2.5 Instrumentação 101
6.2.2.6 Critérios 103
6.2.2.7 Hipótese Nula 103
6.2.2.8 Hipótese Alternativa 104

6.2.2.9 Variáveis Independentes 104
6.2.2.10 Variáveis Dependentes 104
6.2.2.11 Análise Qualitativa 105
6.2.2.12 Capacidade Aleatória 105
6.2.2.13 Classificação em Bloco 105
6.2.2.14 Balanceamento 106
6.2.2.15 Mecanismos de Análise 106
6.2.2.16 Validade Interna do Estudo 106
6.2.2.17 Validade Externa do Estudo 106
6.2.2.18 Validade de Construção do Estudo 106
6.2.2.19 Validade de Conclusão do Estudo 107
6.2.3 Execução do Estudo experimental 107
6.2.3.1 Seleção dos Participantes 107
6.2.3.2 Instrumentação 107
6.2.3.3 Procedimento de Participação 112
6.2.3.4 Execução 113
6.2.4 Análise dos Resultados do Estudo de Observação 117
6.2.4.1 Avaliação Quantitativa 117
6.2.4.2 Eliminação de Valores Extremos 118
6.2.4.3 Teste Paramétrico 120
6.2.4.4 Análise da Execução das Tarefas 120
6.3 Conclusão 121
Capítulo 7 - Conclusão 122
7.1 Trabalhos Futuros 125
Referências Bibliográficas 127
Apêndice A - Modelo de Dados 138
Apêndice B - Tabela de média de avaliações 139
Apêndice C – Conceitos Navegados 143
Apêndice D – Diagramas de Classes 147
Apêndice E – Ontologia 149

Índice de Figuras

Figura 1.1 - Organização do Conhecimento 2
Figura 1.2 - Arquitetura do KCE 3
Figura 2.1 - Hierarquia do Conhecimento (MOREY, FRANGIOSO, 1997) 7
Figura 2.2 - Espiral do Conhecimento (NONAKA & TAKEUCHI, 1995) 9
Figura 2.3 - Fatores Críticos de Sucesso para Criação de Comunidades de Prática (MCDERMOTT, 2000) 12
Figura 3.1 - Texto Exemplo 27
Figura 3.2 - Extrato Exemplo 28
Figura 4.1 - Arquitetura do Olimpo (PEREIRA *et al.*, 2006) 46
Figura 4.2 Exemplo de cadeia de conhecimento navegada 48
Figura 4.3 - Arquitetura interna de um módulo cliente 49
Figura 4.4 - Arquitetura do KCE (PEREIRA *et al.*, 2006; SILVA *et al.*, 2006) 53
Figura 4.5 - Lista de domínios que não serão observados pelo Argus 55
Figura 4.6 - Interface do plug-in Argus 55
Figura 4.7 - Exemplo do arquivo WSDL (W3C, 2007c) do serviço da Internet Hera 56
Figura 4.8 - IO gerada pelo módulo Hermes 59
Figura 4.9 - Navegação das páginas da Internet 60
Figura 4.10 - Ontologia da comunidade 60
Figura 4.11 - Exemplo relacionamento entre páginas e conceitos 60
Figura 5.1 - Arquitetura de ontologia em uma formação de grupo oportunística (SUPNITHI, 1999) 66
Figura 5.2 - Exemplo da ontologia criada 67
Figura 6.1 - Exemplo da interface solicitada na Tarefa 1 73
Figura 6.2 - Menu do Módulo Hermes para obter a cadeia navegada pelo participante 80
Figura 6.3 - Menu do Módulo Hermes para obter a recomendação de cadeia de conhecimentos 80
Figura 6.4 - Formulário para identificar o perfil e a avaliação dos participantes 82
Figura 6.5 - Participantes distribuídos pela fatia de conhecimento em Java 87
Figura 6.6 - Há quantos anos trabalha com Java 88
Figura 6.7 - Conhecimento das bibliotecas básicas da linguagem Java 88

Figura 6.8 - Resultados obtidos no estudo de observação 90
Figura 6.9 - Tempo em segundos para conclusão das tarefas 93
Figura 6.10 - Número médio de nós nas cadeias de conhecimento 95
Figura 6.11 - Média de número de páginas navegadas por rodada por experiência em Java 96
Figura 6.12 - Média do tempo (s) por rodada por experiência em Java 97
Figura 6.13 - Avaliação média das cadeias de conhecimento recomendadas 99
Figura 6.14 - Avaliação média das cadeias de conhecimento recomendadas por rodada 100
Figura 6.15 – Exemplo da tela principal da tarefa 102
Figura 6.16 - Tela de navegação em diretórios 102
Figura 6.17 - Tela de apresentação de texto 3D 103
Figura 6.18 – Médias das distribuições da experiência dos participantes 115
Figura 6.19 - Resultados obtidos no estudo experimental 118
Figura C.1 - Legenda das figuras de conceitos navegados 143
Figura C.2 - Conceitos navegados para cumprir a Tarefa 1 na rodada 1 143
Figura C.3 - Conceitos navegados para cumprir a Tarefa 2 na rodada 1 144
Figura C.4 - Conceitos navegados para cumprir a Tarefa 3 na rodada 1 144
Figura C.5 - Conceitos navegados para cumprir a Tarefa 1 na rodada 3 145
Figura C.6 - Conceitos navegados para cumprir a Tarefa 2 na rodada 3 145
Figura C.7 - Conceitos navegados para cumprir a Tarefa 3 na rodada 3 146
Figura D.1 Diagrama de classes do serviço Web Hera 147
Figura D.2 - Diagrama de classes do *plug-in* Argus 148

Índice de Tabelas

Tabela 4.1 - Exemplos de coeficientes associados aos marcadores HTML (JOACHIMS *et al.*, 1997; DESMONTILS, JACQUIN, 2001). 51
Tabela 4.2 – Exemplo de classificação de páginas 60
Tabela 6.1 – Informações detalhadas sobre os participantes do estudo de observação 86
Tabela 6.2 - Experiência com Java 87
Tabela 6.3 - Resultados obtidos no estudo de observação 89
Tabela 6.4 - Resultados obtidos no estudo de observação com corte T 90
Tabela 6.5 - Estatísticas descritivas sobre os resultados do estudo (após a eliminação de valores extremos) 91
Tabela 6.6 - Resultados intermediários para o Teste T para análise do tempo (90%) 92
Tabela 6.7 - Resultados intermediários para o Teste T para análise do número de páginas visitadas (90%) 92
Tabela 6.8 – Meses de experiência dos participantes por assunto 110
Tabela 6.9 - Quantidade de programas escritos pelo participante. 110
Tabela 6.10 – Numero de participantes X Tema X Grupo. 114
Tabela 6.11 – Numero de participantes X Experiência X Grupo. 114
Tabela 6.12 – Médias dos meses de experiência dos participantes por assunto 116
Tabela 6.13 – Número médio de programas desenvolvidos X Tamanho aproximado dos programas em número de linhas de código 116
Tabela 6.14 – Resultados obtidos no estudo experimental 117
Tabela 6.15 - Resultados obtidos no estudo experimental com corte T 118
Tabela 6.16 - Estatísticas descritivas sobre os resultados do estudo (após a eliminação de valores extremos) 119
Tabela 6.17 - Resultados intermediários para o Teste T para análise do tempo (90%) 120
Tabela 6.18 - Resultados intermediários para o Teste T para análise do número de páginas visitadas (90%) 120
Tabela B.1 - Número de páginas visitadas por usuário x tarefa x rodada 139

Tabela B.2 - Tempo utilizado para pesquisa em segundos por usuário x tarefa x rodada 140
Tabela B.3 - Número de conceitos existentes nas cadeias geradas a partir da navegação por usuário x tarefa 141
Tabela B.4 - Avaliação das cadeias criadas pelo sistema por usuário x tarefa e por usuário x rodada 142

Capítulo 1 - Introdução

1.1 Motivação

A Internet tornou-se um importante meio de disponibilizar informações para pessoas que precisam adquirir novos conhecimentos com maior velocidade e em volume muito maior que no passado. Existem comunidades de prática que atuam como um meio para complementar os tradicionais métodos de ensino das salas de aula, para adquirir conhecimento (SILVA *et al.*, 2006), e para aprimorar o desempenho dos aprendizes (PAWLOWSKI *et al.*, 2000). Essas comunidades são chamadas Comunidades de Aprendizado (KUUSI, 1999; PAWLOWSKI *et al.*, 2000). Um dos fatores citados por de Wenger (1998; *et al.*, 2002) para o sucesso de comunidades de prática é o compartilhamento de conhecimento para aprimorar o conhecimento pessoal. Outro fator relacionado à criação de uma comunidade de prática de sucesso é auxiliar seus membros na construção de seus conhecimentos pessoais (TORNAGHI *et al.*, 2005).

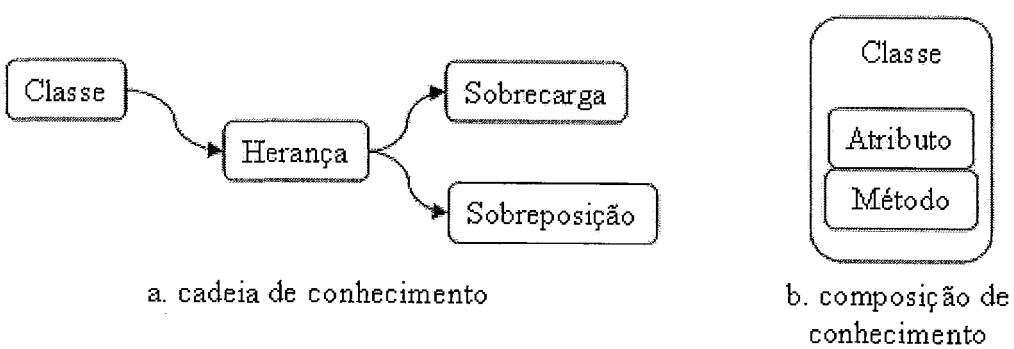
Neste trabalho foi considerado um processo para promover a construção semi-automatizada de cadeias de conhecimento. A disseminação e troca do conhecimento em comunidades de aprendizado, para complementar o processo de aprendizado poderiam ser posteriormente estimuladas através do compartilhamento dessas cadeias. A necessidade de estimular um número de indivíduos a trabalharem juntos sugere um problema para o domínio de CSCW (*Computer Supported Cooperative Work* ou Trabalho Cooperativo Suportado por Computador) (SOUZA *et al.*, 2005). Este processo foi, portanto, criado de forma que a interação com o usuário fosse a mínima possível, mas com flexibilidade aceitar a colaboração dos membros mais participativos. Dessa forma, todos os membros seriam beneficiados da utilização de técnicas de *knowledge desing* (LEITCH, 1986).

Knowledge design é definido como a ciência de selecionar, organizar e preservar o conhecimento em um imenso universo de conhecimento de uma maneira apropriada, de forma que ele possa ser percebido, processado e utilizado por pessoas de forma eficiente e efetiva. Seu objetivo é oferecer o conhecimento certo, para a pessoa certa, da maneira certa, no momento certo. De acordo com Xexéo (2004), a atividade de *design* tem sido descrita como parte de uma classe de problemas que não

possuem solução ótima, apenas soluções satisfatórias. Eles são complexos, muitas vezes de natureza interdisciplinar, e requerem um grupo de pessoas para resolvê-los. Projetar conhecimento é similar, em princípio, a projetar software. Ambos demandam tempo, reflexão cuidadosa e criatividade para que sejam executados corretamente. A maior diferença é que não é possível simplesmente "instalar" o conhecimento no cérebro de alguém, como é possível fazer com um software em um computador. Para isso é necessário um processo de aprendizado para construir o conhecimento na mente do aprendiz (LEITCH, 1986).

1.2 Contexto

Um grupo de pesquisa da COOPE/UFRJ desenvolveu um sistema para complementar o processo de aprendizado, promovendo a construção, disseminação e troca de conhecimento em uma comunidade de aprendizado. Esse sistema é chamado de *Knowledge Chains Editor* (KCE), e é baseado em um processo para criação de conhecimento pessoal através da troca de cadeias de conhecimento (SILVA *et al.*, 2006). Ele foi construído sobre um *framework* para criação de aplicações colaborativas ponto a ponto (ou *peer-to-peer*, *P2P*) chamado COPPEER (MIRANDA *et al.*, 2006). O diferencial deste processo é tornar explícito "como usar" o conhecimento disponível, "quem" o disponibilizou, "onde" ele se encontra, e "o que" esse conhecimento significa.



a. cadeia de conhecimento

b. composição de conhecimento

Figura 1.1 - Organização do Conhecimento

Uma cadeia de conhecimento é uma estrutura criada para organizar conhecimento. Ela é compostas de um cabeçalho (que contém informações básicas relacionadas à cadeia) e uma lista de unidades de conhecimento. A Figura 1.1.a apresenta um exemplo de como adquirir o conhecimento sobre "Classe" é pré-requisito para conhecer "Herança" que, por sua vez, é requisito para conhecer "Sobrecarga" e "Sobreposição". A outra forma de organizar o conhecimento é através

da composição. Quando uma unidade de conhecimento é formada pela composição de outras unidades de conhecimento, ela pode ser representada como a Figura 1.1.b. Neste exemplo, a unidade de conhecimento "Classe" é composta pelas unidades "Atributo" e "Método".

Conceitualmente, conhecimento pode ser decomposto em unidades de conhecimento menores (decomposição recursiva). Com o objetivo de simplificar, vamos considerar que existe uma unidade básica que pode ser representada como uma unidade de conhecimento (estrutura formada por um conjunto de atributos).

Para construir sua cadeia de conhecimento, o aprendiz pode usar o *Knowledge Chain Editor*. No caso de investigação, ele deve criar uma unidade de conhecimento cujo estado seja "dúvida". Nesse momento, o sistema inicia a busca. Ele envia mensagens para outros pontos que estejam rodando outras instâncias do sistema (*peers*) e espera pela resposta. Cada instância do sistema executa uma busca interna. Essa busca consiste em verificar a existência de alguma unidade de conhecimento similar àquele que está sendo buscado. Todas as unidades de conhecimento encontradas são retornadas à instância do sistema requisitante (Figura 1.2).

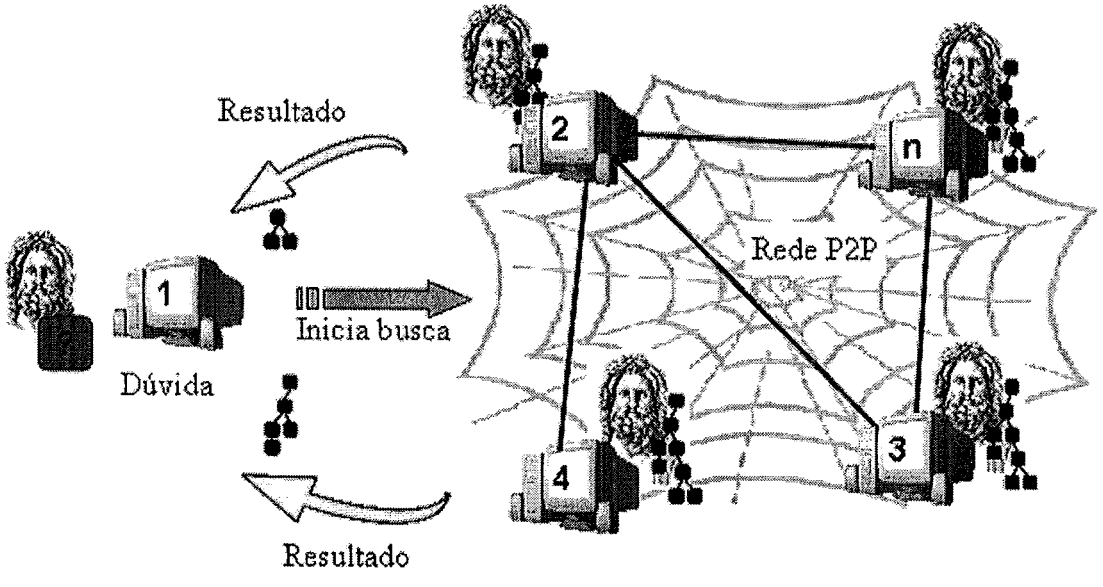


Figura 1.2 - Arquitetura do KCE

A criação da unidade de conhecimento com estado "dúvida" é obviamente motivada pela necessidade do aprendiz de obter aquele conhecimento. Até esse momento foi considerada a existência de dois fatores de motivação para a criação e disponibilidade de cadeias de conhecimento. O primeiro seria com o objetivo de receber o reconhecimento da comunidade, uma vez que cada unidade de

conhecimento criada possui o registro do seu autor. O segundo seria o caso onde o professor os torna disponíveis como um "trabalho", com a intenção de guiar o estudo dos seus estudantes.

No entanto, estamos conscientes que o aprendiz precisa de mais motivação para criar novas cadeias de conhecimento. Na tentativa de solucionar esse problema, apresentamos, neste trabalho, uma proposta para melhorar a criação de novas cadeias de conhecimento: um sistema chamado Olimpo, desenvolvido baseado na proposta apresentada em PEREIRA *et al.* (2006; 2007).

1.3 Objetivos

Nessa dissertação, descrevemos a arquitetura de informação e a implementação do sistema chamado Olimpo. O principal objetivo do Olimpo é recomendar potenciais cadeias de conhecimento que podem ser aceitas, modificadas ou até descartadas pelo aprendiz. Essas cadeias serão criadas a partir dos dados coletados pela monitoração da navegação do aprendiz enquanto este pesquisa determinado assunto na Web. A utilização dessas cadeias por outros aprendizes deve não só auxiliá-los na execução de tarefas específicas, mas também incentivá-los a aprofundar seus conhecimentos no objeto de estudo.

Web é o termo usado para designar a própria rede Internet/Intranet ou a tecnologia que nela é utilizada. "Recurso ou serviço oferecido na Internet (rede mundial de computadores), e que consiste num sistema distribuído de acesso a informações, as quais são apresentadas na forma de hipertexto, com elos entre documentos e outros objetos (menus, índices), localizados em pontos diversos da rede." (FERREIRA, 1999).

Para que essa tarefa seja executada, é necessária uma ontologia do domínio considerado. O objetivo dessa ontologia é determinar o subgrupo de conceitos (conceitos encontrados nas páginas navegadas) que foram navegados durante a pesquisa do aprendiz e relacioná-los às páginas da Internet.

Além da ontologia, o sistema é composto por diferentes módulos desenvolvidos para executar tarefas específicas do processo definido pela ferramenta. Um primeiro módulo chamado Argus observa a navegação do aprendiz através de páginas da Internet. Ele envia as páginas navegadas para o módulo Hera, que armazena informações como conteúdo e tempo de permanência em cada página.

Em seguida, o módulo Hermes tem a responsabilidade de minerar a navegação e o conteúdo das páginas para determinar o subgrupo dos conceitos da ontologia relacionados às páginas navegadas. Uma vez definido este subgrupo, Hermes cria um grafo orientado com esses conceitos. Com essas informações este módulo pode construir a cadeia de conhecimento potencial que será recomendada para o aprendiz.

É importante esclarecer que a nova cadeia de conhecimento deverá ser recomendada, em um primeiro momento, para o mesmo aprendiz que está executando a pesquisa na Web. Cabe a ele(a) decidir se deseja adicionar (ou não) a cadeia de conhecimento recomendada às informações sobre seu conhecimento pessoal contidas no sistema. Desse ponto em diante, se o aprendiz aceitar sua cadeia de conhecimento, ela pode ser distribuída entre os membros da comunidade.

1.4 Organização dos Capítulos

Para facilitar a compreensão do objetivo deste trabalho, o capítulo dois dessa dissertação oferece uma revisão da literatura sobre temas como conhecimento e sua aquisição, comunidades de prática e de aprendizado e como sistemas podem apoiá-las.

O capítulo três destina-se a explicar as técnicas de mineração da Web, dividindo essa atividade em mineração de conteúdo e de utilização. Aproveitamos ainda para apresentar técnicas de folksonomia, utilizada para auxiliar no processo de classificação das páginas.

A arquitetura proposta para o sistema Olimpo e a descrição de como ele foi desenvolvido para seu estudo de observação são assuntos abordados no capítulo quatro, enquanto o capítulo cinco descreve a ontologia criada neste trabalho.

O capítulo seis apresenta a metodologia utilizada no estudo de observação ao qual o sistema foi submetido e as análises geradas a partir dele. Finalmente, o capítulo sete conclui esta dissertação apontando as contribuições e possíveis melhoramentos da proposta.

Capítulo 2 - Recomendação em Comunidades de Aprendizado

Um sistema de recomendação é um software cujo objetivo principal é auxiliar seus usuários no complexo desafio de tomar decisões baseadas num universo sobrecarregado de informações imprecisas e duvidosas (GOLDBERG *et al.*, 1992; RESNICK, VARIAN, 1997). O papel do sistema de recomendação é descobrir o subconjunto de informações relevantes aos usuários, evitando que estes percam tempo em análises e pesquisas em vão. Quanto maior a sobrecarga de informação e a inexperiência do usuário, maior é o desafio de descobrir por conta própria o que realmente é importante.

Sob um ponto de vista mais comercial, sistemas de recomendação aprendem com as preferências do usuário e automaticamente sugerem produtos que atendam ao seu perfil (RESNICK *et al.*, 1994). Esses sistemas podem sugerir informação ou produtos que os usuários não estão exatamente à procura, mas que seus perfis de compras sugerem que possam ter interesse ou mesmo adquirir um produto.

Segundo Schafer *et al.* (2001), existem três tipos básicos de recomendação:

1. A que facilita os processos de busca de informação e comparação de produtos.
2. A que qualifica informações através de opiniões de outros usuários interessados no mesmo assunto.
3. A que traça um perfil dos usuários através dos seus históricos e encontra semelhanças no comportamento de diferentes usuários.

As informações recomendadas por um sistema de recomendação, podem ser obtidas explicitamente, através de críticas ou etiquetagem dos usuários; implicitamente, através da análise comportamental dos usuários; ou por interferência, quando o sistema aprende sobre o perfil do usuário quando esse executa uma ação explícita, por exemplo, uma compra.

Isso significa que, para o funcionamento do sistema de recomendação, é necessária a sua utilização por um grupo de pessoas, que nesse grupo haja subgrupos interessados no mesmo tipo de informação ou estejam dispostos a colaborar entre si.

A principal motivação deste trabalho é auxiliar pessoas no processo de aprendizado recomendando como obter o conhecimento adquirido por outros membros interessados no mesmo assunto. Nesse caso, não é apenas necessário que haja um grupo de pessoas, mas também que todas tenham o objetivo comum de aprender. Para entender como esse compartilhamento pode ajudar na aquisição de conhecimento alguns conceitos devem ser discutidos.

Neste capítulo discutiremos a evolução e aquisição do conhecimento, o agrupamento de pessoas em torno de conhecimentos comuns e as ferramentas para auxiliar os membros das comunidades a evoluírem em conjunto para alcançar seus objetivos individuais.

2.1 Conhecimento

Com uma rápida pesquisa na literatura sobre gestão de conhecimento ou outras áreas relacionadas, observamos que muitos autores empregam as palavras "informação" e "conhecimento" como sinônimos. No entanto, apesar da estreita relação entre elas, muitos autores distinguem seus significados e muitas vezes mostram a relação entre elas de forma hierárquica. É importante entender essa distinção para poder compreender a real dificuldade na difusão do conhecimento, que supera em muito a dificuldade da difusão de informação.

Morey & Frangioso (1997) representam um bom exemplo de autores que organizaram o conhecimento e a informação de forma hierárquica. Ou seja, definem que o conhecimento é hierarquicamente superior à informação, que por sua vez é definida sobre o dado. O dado é o registro de uma transação. A informação é o dado contextualizado e com sentido dentro desse contexto. O conhecimento, por sua vez, é o conjunto de habilidades criadas, a experiência adquirida sobre as informações acumuladas ao longo do tempo.

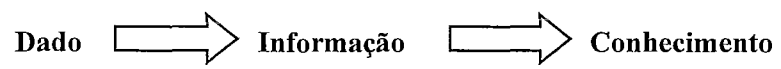


Figura 2.1 - Hierarquia do Conhecimento (MOREY, FRANGIOSO, 1997)

Dado seria, nesse caso, tudo aquilo que é tangível. Textos, gráficos, artigos, livros, coisas que podem ser facilmente trocadas entre as pessoas. Informação é quando um dado é inserido em um contexto conhecido e assume significado. Conhecimento, nesse caso, seria a compreensão das informações. A forma como uma pessoa assimila uma informação ou interconecta a outras informações em sua mente é

o que define o conhecimento. O estado mental de uma pessoa está em constante mudança dado o grande número de informações que ela recebe a todo o momento. Drucker (1998) acredita que conhecimento é a aplicação da informação para obter resultados. Esses resultados são vistos na sociedade, na economia, ou no avanço do próprio conhecimento.

Essa abordagem define conhecimento como um conjunto formado por experiências, valores, informações de contexto, criatividade aplicada à avaliação de novas experiências e informações. Dessa forma, o conhecimento não poderia ser separado das pessoas que o detêm. Qualquer forma de externalizar o conhecimento o transformaria imediatamente em informação que só voltaria a ser transformado em conhecimento quando assimilado por outro indivíduo. Esse, por sua vez, não irá adquirir esse conhecimento exatamente igual ao que originalmente foi externalizado, pois o conhecimento original estava associado ao estado mental da pessoa que o transformou em informação.

Outra abordagem defendida por vários autores (POLANYI, 1983; NONAKA, TAKEUCHI, 1995; BARCLAY, MURRAY, 1997) divide o conhecimento em duas categorias. Uma delas é onde o conhecimento está codificado em uma linguagem inteligível por outras pessoas e por isso pode facilmente ser transmitido para outras pessoas. Neste trabalho, chamaremos essa categoria de conhecimento **explícito**. A outra categoria é formada pelos conhecimentos pessoais, específicos para um contexto e difíceis de formalizar. Esses conhecimentos estão relacionados com a experiência e crenças pessoais. Chamaremos os conhecimentos dessa categoria de **tácitos**.

A abordagem mencionada anteriormente define um único tipo de conhecimento, o tácito. Segundo ela, o conhecimento explícito é simplesmente definido como informação. Na verdade, a real importância neste trabalho é dada ao conhecimento tácito e no desafio de explicitá-lo e compartilhá-lo, já que isso, por definição, é uma tarefa difícil.

Como podemos ver existem inúmeras definições e categorizações para conhecimento. Muitas são específicas de acordo com a área ou lugar onde o conhecimento é aplicado. Este trabalho foi embasado na definição de conhecimento tácito e explícito (POLANYI, 1983).

2.1.1 Aquisição do conhecimento

Independente das diferentes definições para conhecimento, a maior parte dos autores citados anteriormente concorda que há uma forma de conhecimento que fica interiorizada na mente das pessoas outra forma de conhecimento, ou informação, que pode ser explicitada e compartilhada. O ponto crucial é saber como adquirir conhecimento que possa ser usado para alcançar os objetivos individuais ou organizacionais.

Já na antiguidade existiram estudos sobre a criação de conhecimento. Desde então muitos filósofos e psicólogos estudaram o tema. Hoje diversos estudiosos pesquisam formas de ajudar pessoas e instituições no processo de criação de conhecimento, principalmente através de ferramentas.

Nonaka & Takeuchi (1995) propuseram um processo de criação do conhecimento focado em ambos os tipos: tácito e explicito. Esse processo ocorre ao longo do tempo com duração infinita, passando em espiral por etapas de socialização, internalização, combinação e externalização. A interação contínua entre o conhecimento tácito e o conhecimento explícito no decorrer das quatro etapas da Figura 2.2 permite que o conhecimento seja não só compartilhado, mas também combinado com conhecimentos anteriores. Assim, compartilhamento e criação de conhecimento fazem parte de um processo contínuo, infinito e recorrente.

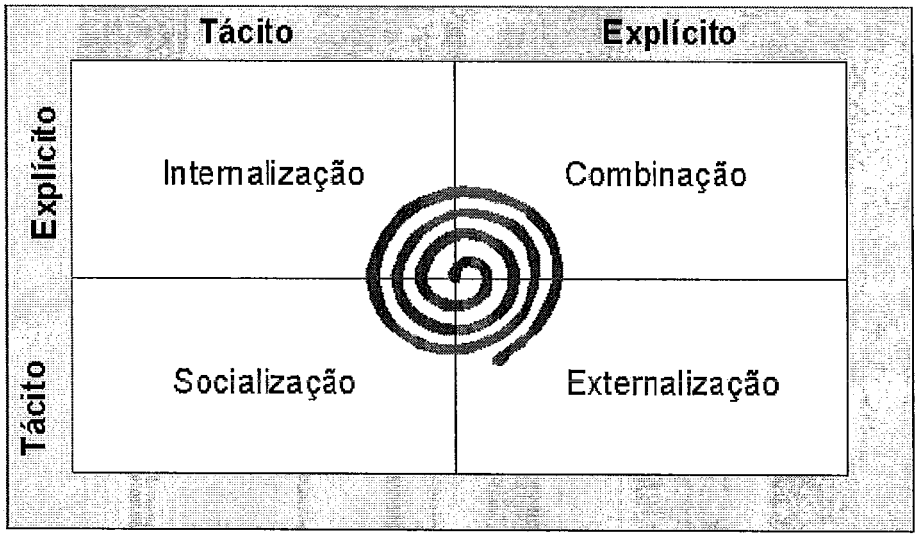


Figura 2.2 - Espiral do Conhecimento (NONAKA & TAKEUCHI, 1995)

A internalização é a etapa onde o indivíduo assimila o conhecimento explícito. Esse conhecimento pode ter origem em uma base de conhecimento de uma organização ou pode ter sido pesquisado pelo indivíduo em meios diversos. Muitas

vezes pode ser necessário exercitar alguma habilidade para internalizar esse conhecimento, como um músico precisa praticar uma música que foi externalizada em uma partitura. Apenas a leitura da partitura não garante a internalização da música. Nesse momento, o conhecimento explícito é transformado em conhecimento tácito.

O processo de combinação é a etapa anterior à externalização, quando o indivíduo começa a classificar os conceitos da informação previamente internalizada. Durante esse processo ele pode, intencionalmente ou não, acrescentar informações relevantes e até aplicar o conhecimento em outro contexto. Nesse processo, fatores em comum resultarão na combinação de conhecimentos. Após esse processo, o conhecimento passa pela etapa de externalização onde pode ser traduzido em novos conceitos capazes de serem justificados, categorizados e contextualizados em uma organização.

A socialização é o processo onde os indivíduos trocam experiências. Imitação e experimentação assistida são as principais formas de compartilhamento do conhecimento tácito. Além da distribuição de conhecimento explícito para que outros indivíduos possam iniciar um novo processo de internalização.

Podemos notar na Figura 2.2 que a etapa de socialização é o compartilhamento de conhecimento entre indivíduos, ou seja, o conhecimento é transmitido muitas vezes sem a necessidade de formalizá-lo em um documento. A etapa de externalização, por sua vez, corresponde à transformação do conhecimento tácito de uma pessoa em conhecimento explícito, ou seja, codificado. Em ambas as etapas, as comunidades de práticas podem atuar com um grande fator estimulador.

2.2 Comunidades de prática

Lave & Wenger (1991) entendem a aprendizagem como uma experiência que faz parte integrante da participação em comunidades de prática. Comunidades de práticas são grupos de pessoa que se reúnem, pessoalmente ou virtualmente, em torno de um tópico de interesse comum a todos integrantes (WENGER, 1998).

Uma comunidade de prática pode ter seu foco apontado para uma disciplina profissional, como bioengenharia ou análise financeira; uma habilidade, como tocar um instrumento; ou um tópico, como uma tecnologia específica, uma indústria ou um segmento de produção. Comunidades de práticas sempre fizeram parte da estrutura informal das organizações (MCDERMOTT, 2000).

Freqüentemente os membros de uma comunidade se reúnem em torno de um tópico que dedicaram meses, talvez anos de desenvolvimento. Tópicos que em muitos desses membros despertaram entusiasmo acima do interesse ordinário. No entanto, uma comunidade de prática não é apenas formada em torno de celebrações de interesses comuns. Em aspectos mais práticos, as comunidades focam em solução de problemas do dia a dia, novas ferramentas, desenvolvimentos no campo de pesquisa, coisas que funcionam ou não. Isso significa que os membros de uma comunidade a compõem não só pelo entusiasmo no assunto, mas também porque a troca de informações agrega valor prático.

Uma das principais forças que movem uma comunidade é que seus membros pedem e oferecem ajuda para solução de problemas técnicos. A freqüência da troca de informações ajuda os membros da comunidade a mostrarem suas deficiências e aprenderem juntos no espaço público de comunidade. Discussões francas de problemas reais constroem um grande senso de conexão e confiança entre os membros da comunidade. Como compartilham idéias e experiências, membros da comunidade freqüentemente desenvolvem uma forma comum de executar tarefas, um conjunto de práticas comuns e um grande senso de propósito comum. Algumas vezes isso é formalizado em diretrizes e padrões; em outras esse conhecimento se resume a um conjunto de informações sobre boas práticas que "todos sabem". No processo de ajudar um ao outro, compartilhando idéias e colecionando problemas resolvidos, todos acabam transformando-se em um grupo confiável de indivíduos.

Mcdermott (2000) acredita que as comunidades de prática são os veículos ideais para promover conhecimento tácito porque elas habilitam a interação interpessoal e engajam todo um grupo em desenvolver seu campo de prática. Como resultado eles podem difundir a visão desse pensamento colaborativo através de toda organização.

2.2.1 Fatores de sucesso para criação e manutenção de comunidades de práticas

Wenger, Mcdermott & Snyder (2002) apontam a voluntariedade como principal fator de sucesso em uma comunidade. A habilidade das comunidades de prática de gerar excitação, relevância e valor para seus membros é o que as mantém funcionando ao longo do tempo. Esses grupos são formados espontaneamente quando as pessoas precisam de ajuda, tentam solucionar um problema, desenvolvem novas

idéias e pesquisam. Isso significa que a participação deve, acima de tudo, ser espontânea e intencional.

Segundo Mcdermott (2000), existem dez fatores de sucessos, divididos em quatro grandes desafios, para criação e manutenção de comunidades de práticas capazes de compartilhar conhecimento tácito. Os desafios e os fatores embutidos em cada um deles ficam claramente representados na Figura 2.3.

I. Desafio da Gerência

1. Focar em tópicos importantes para o negócio e para os membros da comunidade.
2. Encontrar um membro respeitado para coordenar a comunidade.
3. Garantir que as pessoas têm tempo e encorajamento para participar.
4. Construir sobre os valores culturais centrais da organização.

II. Desafio da Comunidade

5. Envolver os líderes.
6. Criar relacionamento pessoal entre os membros da comunidade.
7. Desenvolver um grupo centrar ativo e entusiasmado.
8. Criar fóruns para pensamentos comuns, assim como sistemas para compartilhar informações.

III. Desafio Técnico

9. Facilitar a contribuição e o acesso ao conhecimento e práticas da comunidade.

IV. Desafio Pessoal

10. Criar discussões sobre lançamentos de ponta nos fóruns da comunidade.

Figura 2.3 - Fatores Críticos de Sucesso para Criação de Comunidades de Prática (MCDERMOTT, 2000)

I. **O Desafio da Gerência** está principalmente em gerenciar e administrar o conhecimento produzido na comunidade. Esse trabalho extra é freqüentemente rejeitado pelos membros da comunidade dado a dificuldade existente em conseguir a devida atenção da equipe profissional para essa atividade. No entanto quatro fatores podem mostrar que realmente apóia o compartilhamento de conhecimento nas comunidades:

1. **Focar em tópicos importantes para o negócio e para os membros da comunidade** – Para mostrar que a comunidade de pratica é importante e que

os tópicos ali discutidos terão impacto financeiro ou competitivo significante para a organização. Mas os tópicos também precisam ser sobre assuntos de entusiasmo pessoal.

2. **Encontrar um membro respeitado para coordenar a comunidade** – Comunidades são mantidas por pessoas que se importam com a comunidade, que possuem alguma paixão pelo tópico e pelas pessoas que participam. Elas mantêm os outros membros informados do que cada um está fazendo e criam oportunidades para pessoas se reunirem para compartilhar idéias.

3. **Garantir que as pessoas têm tempo e encorajamento para participar** – Um dos maiores fatores limitantes da efetividade de uma comunidade de prática em dividir conhecimento é o tempo que as pessoas têm para participar. A participação na comunidade, mesmo quando é muito valiosa, pode ser facilmente sobrepujada por pressão de tarefas externas.

4. **Construir sobre os valores culturais centrais da organização** – A cultura de uma organização é freqüentemente culpada pelas falhas na implementação de sistema de gestão de conhecimento. Argumenta-se que as pessoas não estavam dispostas a compartilhar suas idéias ou dedicar tempo para documentar seus pensamentos. Um estudo recente de cultura corporativa e gestão do conhecimento (MCDERMOTT & O'DELL, 2000) descobriu que independente do quão comprometido e da abordagem para gerir o conhecimento, a cultura é mais forte. Companhias que obtiveram sucesso em compartilhar conhecimento não tentaram mudar suas culturas para se adaptar à gestão do conhecimento. Elas concentraram esforços para adaptar seus métodos de gestão de conhecimento às suas culturas.

II. **O desafio da comunidade** é manter sua energia ao longo do tempo. O maior perigo de cultivar comunidades é que elas percam sua energia e caiam em apatia, deixando o coordenador com toda a responsabilidade de cuidar da comunidade. Muitos fatores podem ajudar a manter a energia da comunidade e mantê-la atualizada no seu campo de atuação:

5. **Envolver os líderes** – Construir uma comunidade normalmente começa encontrando, alimentando e desenvolvendo uma rede de comunicação que já

exista. Em geral há pessoas chaves que possuem um conhecimento especializado importante, ou que são bem relacionados e influentes membros daquela comunidade. Envolver essas pessoas é importante porque elas atraem outros membros, tornando assim a comunidade legítima.

6. **Criar relacionamento pessoal entre os membros da comunidade** – Coordenadores de sucesso visitam os membros de suas comunidades, descobrem em que eles estão trabalhando, os referenciam ou apresentam a outros membros da comunidade, trazem novas idéias ou descobrem oportunidades para a comunidade desenvolver sua prática.

7. **Desenvolver um grupo centrar ativo e entusiasmado** – A maioria das comunidades têm um grupo central que contribui ativamente; um grande grupo que consome informação, mas acrescenta pouco; e um grupo periférico maior ainda que só participa ocasionalmente. Os membros do grupo central não só contribuem como também se sentem responsáveis por ajudar a desenvolver a comunidade convidando ou facilitando a participação de pessoas que eles conhecem.

8. **Criar fóruns para pensamentos comuns, assim como sistemas para compartilhar informações** – Contato é a chave para o sentimento de associação, entusiasmo e confiança. Além de encontros individuais e contatos on-line, devem ser criadas oportunidades para a comunidade compartilhar idéias como um grupo.

III. **O Desafio Técnico** é projetar o lado social da tecnologia de informação. Há muita tecnologia boa para colaboração e compartilhamento de informação que tenta focar na funcionalidade dos produtos.

9. **Facilitar a contribuição e o acesso ao conhecimento e práticas da comunidade** – Como o mercado explode com tantos tipos diferentes de software de gestão de conhecimento, nós encontramos duas coisas particularmente importantes para comunidades. Primeiro, um software deveria facilitar e contribuir para a comunicação entre os membros da comunidade, além de usar comunicação da base de conhecimento da comunidade. Facilidade de uso está mais relacionada com a forma como o software se

integra com o trabalho diário das pessoas, o conhecimento elas precisam compartilhar, a forma como eles pensam sobre o domínio da sua comunidade.

IV. **O Desafio Pessoal** para a maioria dos membros da comunidade é desenvolver a capacidade da mesma. Os eventos de maior valor e uma comunidade focam com muita mais freqüência na resolução de problemas que na apresentação de práticas. Mas discutir problemas, compartilhar idéias inacabadas, ou pensar alto em público não é natural para a maioria das pessoas. Em geral, os membros acham difícil falar sobre seus problemas com pessoas que não conhecem.

10. **Criar discussões sobre lançamentos de ponta nos fóruns da comunidade** – No estágio inicial do desenvolvimento de comunidades, normalmente são promovidos encontros. Neles um membro respeitado pela comunidade pede ajuda para um problema qualquer e permite que os outros membros do encontro ofereçam suas opiniões. Essas ajudas legitimarão a discussão de problemas na comunidade. No entanto, as discussões promovidas pelos membros respeitados devem ser genuínas e as solicitações de problemas reais.

Wenger, Mcdermott & Snyder (2002) sugerem sete princípios que devem ser seguidos para estimular comunidades de práticas, são eles:

1. **Planejar para evolução** – As comunidades normalmente surgem a partir de redes de conhecimento entre pessoas. Para estimular uma comunidade e garantir seu sucesso é importante identificar pessoas que façam parte dessas redes além de outras que possam agregar valor a ela, o tipo de conhecimento que será compartilhado e qual o ambiente dessa comunidade. Estimular o contato pessoal ou virtual, através de facilidades de comunicação, das pessoas dessa rede irá criar espontaneamente uma comunidade de prática.
2. **Abrir um diálogo entre perspectivas internas e externas** – É fundamental que haja ao menos uma pessoa que esteja na comunidade e possua uma visão geral de todo o negócio da organização para que seja criado um diálogo entre o que se espera da comunidade e o que ela realmente está fazendo.
3. **Convidar diferentes níveis de participação** – A produtividade da comunidade em relação às idéias e soluções criadas aumenta quando há

diferentes participantes com os mesmos interesses, mas com pontos de vistas diferentes. Isso irá estimular a combinação de conhecimento. É importante ainda que haja diferentes níveis de comprometimentos, havendo assim um ou mais líderes que possam organizar a comunidade.

4. **Desenvolver espaços para comunidades públicas e privadas** – É importante que dois ambientes para trocas de experiências: um público e um privado. No primeiro, os integrantes das comunidades da organização poderão trocar experiências com pessoas externas à organização diversificando as idéias e soluções encontradas. No segundo ambiente, são discutidos os temas mais ligados à organização e que não devem ser divulgados externamente.
5. **Focar no valor** – Deve ser claro que a comunidade agrega valor a seus participantes, mesmo que seja difícil mensurar esse valor agregado. As comunidades que normalmente são criadas para resolver um problema específico recebem cada vez mais conhecimento através de seus membros. Isso irá agregar cada vez mais valor a ela.
6. **Combinar familiaridade e excitação** – O sentimento de cumplicidade entre os participantes deve ser levado a um nível tal que esses se sintam ligados na solução de problemas comuns. Promover eventos familiares e excitantes faz com que seus membros desenvolvam o relacionamento necessário que se sintam comprometidos com a comunidade.
7. **Criar uma rotina para a comunidade** – A rotina da comunidade deve ser normalizada, ou seja, as diferentes rotinas dos integrantes devem ser igualadas em uma rotina gerais da comunidade. Isso pode ser feito através de reuniões ou eventos como mencionados acima. Há vários indicativos de uma rotina de uma comunidade que mostram sua vitalidade, como a freqüência com que encontros pessoais ocorrem, o fluxo de pessoas que entram e saem, o nível de participação ativa, entre outros.

Em geral, comunidades de prática são informais e surgem espontaneamente. No entanto estão sendo largamente empregadas na gestão do conhecimento dado a pré-disposição dos integrantes a compartilhar experiências e informações. Muitas

organizações estão descobrindo a que essas comunidades podem oferecer. Com o interesse de organizações em estimular a criação de comunidades, estudos para encontrar formas de fazê-lo têm cada vez mais importância.

2.3 Comunidades de aprendizado

Kuusi (1999) define comunidade de aprendizado como uma maneira de descrever agentes básicos e instituições interagindo e construindo diferentes tipos de redes. As funções definitivas de uma comunidade de aprendizado são sua gerência de conhecimento comum ou logística de atividades de conhecimento resultando na adoção ou na produção de inovações.

Pawlowski (2000), por sua vez, define uma comunidade de aprendizado como sendo um grupo informal de indivíduos reunidos ao redor de um interesse comum, melhorando suas capacidades como aprendizes usando redes de computadores.

Comunidades de aprendizado são entendidas como uma especialização de comunidades de práticas. Nesse caso específico, os membros da comunidade juntam-se ao redor do interesse de adquirir um conhecimento específico. Comunidades de aprendizado, em geral, são utilizadas como uma forma de complementar o aprendizado tradicional em sala de aula ou cursos. Para outros autores, comunidades de aprendizado são construídas a partir de combinação de qualquer grupo de pessoas interessadas não só no aprendizado, mas por qualquer grupo de pessoas interessadas em educação. Assim, não apenas os alunos, interessados em aprender, formam comunidades de aprendizado, mas também um comitê escolar, um departamento educacional ou uma organização nacional de professores.

Neste trabalho consideramos comunidades de aprendizado como sendo apenas um grupo de indivíduos interessados em adquirir um conhecimento específico e que se reúnem em um grupo com objetivo de complementar o processo de aprendizado tradicional.

Como uma comunidade de aprendizado é essencialmente uma especialização de uma comunidade de prática, todos os tópicos descritos na seção anterior são válidos como fatores importantes para criar e manter uma comunidade de aprendizado. No entanto, nesse caso específico os desafios de motivar os membros, promover eventos, incentivar o uso de softwares, entre outros são mais fáceis de serem executados. Primeiro porque em geral as comunidades são menores e locais, já que na maioria das vezes são formadas por aprendizes de uma mesma organização.

Em segundo lugar porque os membros já se predispuseram a aprender o tema através de um processo de aprendizado tradicional. No entanto, os desafios referentes a motivar a participação ativa na comunidade continuam os mesmos.

2.4 Colaboração oportunística

O termo colaboração é largamente utilizado na atual literatura educacional e de computação. O campo do Apoio Computacional para o Aprendizado Colaborativo (CSCL) dá alguma indicação da diversidade da noção. Dillenbourg *et al.* (1994) fornecem uma discussão teórica sobre colaboração a qual, até certo grau, alimenta essa diversidade com o intuito de obter uma visão geral da pesquisa sobre a área.

Existirem muitos significados associados ao termo colaboração, esses significados são freqüentemente associados com objetivos educacionais diferentes. Entre as questões de maior significância estão:

- Se a tarefa é dividida em partes controladas por diferentes colaboradores ou se a colaboração requer um esforço síncrono sem nenhuma divisão de tarefa;
- Se a colaboração é vista como um estado ou como um processo;
- Se a colaboração é um meio para de aprender alguma esfera de domínio ou se a colaboração é, em algum sentido, o fim em si.
- Se os participantes em uma colaboração estão cientes da existência de uma relação contratual formal ou não.

O primeiro dos pontos acima é usado por Roschelle & Teasley (1995) para distinguir cooperação e colaboração: "O trabalho cooperativo é realizado através da divisão do trabalho entre os participantes, como uma atividade onde cada pessoa é responsável por uma porção da solução do problema..." ao passo que a colaboração envolve o "empenho mútuo dos participantes em um esforço coordenado para solucionar juntos o problema."

Em relação ao segundo ponto descrito anteriormente, poder-se-ia argumentar que cooperação e colaboração se excluem mutuamente. Por exemplo, se uma tarefa é dividida e diferentes participantes trabalham em cada parte dela, então durante esse período os participantes estão cooperando e não colaborando. Entretanto, a posição tomada aqui é de que a colaboração tem que ser considerada tanto como um estado

como um processo. Sob esse ponto de vista, os participantes cooperariam em um processo e manteriam a colaboração como um estado.

O terceiro ponto diz respeito à abordagem feita por Burton *et al.* (1997) juntamente com Brna & Burton (1997). Eles tomam a posição a partir da qual dentro do estado colaborativo pode haver processos que são cooperativos ou mesmo alguns que são freqüentemente associados com argumentação. Concentração na noção das "metas" da esfera de domínio da colaboração, como na definição de Roschelle & Teasley (1995), levou os pesquisadores a ignorarem algumas formas interessantes de colaboração. Por exemplo, é esperado que uma colaboração de alta qualidade pode bem incluir uma argumentação, e que essa argumentação exige que diferentes participantes adotem e tentem manter posições distintamente diferentes.

O quarto ponto é se existe ou não um contrato formal entre os participantes. Em contextos típicos de sala de aula, é bem comum hoje encontrar a noção de contrato de aprendizado, esboçando as obrigações contratuais entre escola, estudante e pais. Mesmo nas escolas onde contrato não é explícito, há um contrato formal implícito de que os estudantes irão freqüentar a escola para aprender suas lições, comportar-se de maneira razoável, etc. Em uma parceria colaborativa, quaisquer obrigações contratuais formais são suplementadas por um conjunto de obrigações implícitas (GRICE, 1975).

O contrato entre os dois colaboradores inclui uma obrigação formal para fornecer uma crítica do trabalho do outro colaborador e então discutir as disparidades entre seu próprio trabalho e o trabalho do outro participante (BULL, BRNA, 1997). Os dois participantes podem ser considerados em um estado de colaboração, vinculados um em relação ao outro para:

- Concordar mutuamente para colaborar;
- Manter um modelo de trabalho acerca das habilidades e conhecimentos de cada um;
- Ter um objetivo em comum (chegar a um acordo quanto à solução de uma tarefa);
- Manter crenças sobre a meta em comum, tal que ambos continuem com a mesma meta;
- Manter uma compreensão compartilhada do problema, o que implica que eles irão precisar discutir o estado dos seus progressos.

Há situações nas quais nenhum contrato formal implícito ou explícito existe entre os parceiros colaboradores. Isso sugere uma forma de colaboração oportunista que é mantida por um conjunto de obrigações e crenças implícitas tais como as de que cada participante pode fazer uma contribuição significativa para a solução do problema, ou que é educado responder às perguntas de uma maneira informativa. Isso se aproxima do conceito de diálogo cooperativo que dá embasamento a muitas abordagens feitas pela comunidade de lingüística computacional para a questão da geração de diálogo colaborativo.

2.5 Conclusão

Esse trabalho é voltado principalmente para complementar o aprendizado em comunidades de aprendizado ou de prática. Para que seu objetivo seja atingido é necessário auxiliar os membros dessas comunidades a externalizarem e compartilharem os conhecimentos que adquiriram durante um tempo finito de pesquisas.

De acordo com os fatores críticos de sucesso para criação e manutenção de comunidades de prática apresentados por Mcdermott (2002) no tópico 2.2.1, este trabalho pode auxiliar a vencer o desafio técnico (III, item 9) uma vez que basta que os membros naveguem em busca de informações que necessitem e já estarão aptos a compartilhar conhecimento.

Se considerarmos os fatores de sucesso apontados por WENGER, Mcdermott & Snyder (2002), veremos que a contribuição acontece nos fatores:

1. Juntamente com o conhecimento que é compartilhado é informado o membro que o originou, criando assim um relacionamento interpessoal entre os membros.
3. Troca informações de membros de diferentes níveis, estimulado iniciantes a obterem informação e experiência com membros mais experientes.

A forma utilizada para obter informação sobre a navegação dos usuários do sistema conta com a análise comportamental dos mesmos. O que nos importa é como determinado membro se comportou para adquirir determinado conhecimento.

O sistema proposto é dividido em várias metas menores para que seja alcançado seu objetivo geral. Ele deve semi-automatizar a colaboração oportunística

entre os membros da comunidade. Permitindo que essa aconteça não na sua informalidade, mas também de forma mais fácil e transparente possível.

Capítulo 3 - Mineração de Dados e Estratégias de Classificação de Conhecimento

A parte mais importante deste trabalho envolve a tarefa de classificar o conteúdo das páginas navegadas na Internet. Essa tarefa é dividida essencialmente em três partes: (1) extração da navegação do usuário a fim de obter informações das páginas e a ordem em que foram navegadas, (2) mineração dos textos extraídos das páginas e a classificação destes de acordo com uma ontologia dada e, por fim, (3) mineração de informações relevantes da ordem como as páginas foram navegadas.

As *tags,* etiquetas do código HTML (acrônimo para a expressão inglesa *HyperText Markup Language*) (W3C, 1999), e outros tipos de linguagens de marcação ou linguagens de script utilizadas nas páginas dificultam a extração do conteúdo. Técnicas de mineração de conteúdo devem ser empregadas neste caso. Técnicas de mineração da Web são utilizadas para extrair informações da navegação dos usuários.

Para esse trabalho adaptamos diversas técnicas de mineração da Web para que pudéssemos extrair informações da navegação dos usuários e do conteúdo das páginas navegadas para que fossem convertidas em uma estrutura hierárquica de conceitos. Neste capítulo, discutiremos essas técnicas.

3.1 Mineração da Web

Os princípios e técnicas de mineração de dados e descoberta de conhecimento podem ser aplicados com sucesso na *World Wide Web* para extrair conhecimentos úteis e interessantes sob diversos aspectos, seja do ponto de vista dos usuários ou dos desenvolvedores de *web sites*. A este tipo de aplicação da mineração de dados dá-se o nome genérico (COOLEY *et al.*, 1997) de mineração da Web ("*Web mining*"), que possui, no entanto, diversas ramificações com variados matizes.

Uma das classificações propostas para a mineração da Web (ZAÏANE, 1999; ZAÏANE, 2000) distingue entre mineração de conteúdo da Web ("*Web content*

mining"), mineração de estrutura da Web ("*Web structure mining*") e mineração de utilização da Web ("*Web usage mining*").

Segundo essa classificação, a mineração de conteúdo procura e extrai informações relevantes do próprio conteúdo dos documentos da Web. Neste contexto, estão incluídas a mineração de dados textuais, a mineração baseada em indexação de conceitos e as tecnologias baseadas em agentes. Este tipo de mineração está voltada principalmente para os usuários finais da WWW. A mineração de estrutura, por sua vez, procura inferir o conhecimento com base na própria organização dos documentos e nos *links* entre eles. É de interesse, principalmente, dos desenvolvedores e projetistas de *sites*. Finalmente, a mineração de utilização é aquela que procura extrair conteúdos relevantes a partir dos *logs* de utilização, ou seja, minera os próprios *logs* de acesso dos servidores da Internet na busca de padrões de uso. Assim como a anterior, também auxilia primordialmente os desenvolvedores e projetistas.

Uma classificação mais simplificada, porém, diferencia apenas entre mineração de conteúdo e mineração de utilização (COOLEY *et al.*, 1997). Aqui a mineração de conteúdo também tem, como foco principal, a busca do conhecimento presente nos próprios dados contidos nas páginas da Web. Porém, nesta classificação, as atividades que seriam de mineração de estrutura estão incluídas na mineração de utilização, já que os próprios métodos de preparação de dados, análise e busca de conhecimento sobre a utilização das páginas da Web salientam a necessidade de se ter informações estruturais sobre os *sites*. Neste trabalho é utilizada uma mistura de ambos os tipos de mineração.

Em um primeiro momento mineramos a utilização para saber quais páginas foram navegadas e quais são relevantes sob o ponto de vista da permanência do aprendiz. Posteriormente inserimos a mineração de conteúdo para fazer uma nova filtragem de relevância em relação ao conteúdo e para classificar as páginas visitadas.

3.2 Mineração de Conteúdo

A principal dificuldade na mineração de conteúdo é a falta de estrutura dos documentos armazenados na Internet. Contudo, a recuperação de informações na Web tem, nos últimos anos, se beneficiando do desenvolvimento de agentes inteligentes e das iniciativas de se conseguir um maior grau de estruturação dos dados disponíveis.

A técnica de mineração de conteúdo lança mão do texto para recuperação de informação. A abordada nessa dissertação é a extensão dessa técnica para

Sumarização Automática (SA), proposta por Larocca Neto *et al.* (2000), para extração de sentenças mais relevantes do texto.

O método parte da idéia de que, se uma palavra pode ser significativa em um texto por sua freqüência, ela também pode indicar a importância relativa das sentenças que a contêm. Trabalha-se com a noção da freqüência inversa de termos sentenciais (em geral, palavras), utilizando a medida chamada *Term Frequency–Inverse Sentence Frequency* ou, simplesmente, TF-ISF. Essa métrica é análoga à TF-IDF (*Term Frequency–Inverse Document Frequency*) de Salton & Buckley (1988), aplicada para a determinação de termos-chave de documentos. A fórmula para calcular o valor TF-ISF é:

$$TF - ISF(p,s) = TF(p,s) * ISF(p) \quad (1)$$

Onde TF(p,s) é o número de vezes em que a palavra "p" ocorre na sentença "s" e ISF(p) é a significância de "p" em "s" em relação à sua ocorrência no restante das sentenças do texto-fonte. ISF(p) é definida pela fórmula:

$$ISF(p) = \log\left(\frac{|S|}{SF(p)}\right) \quad (2)$$

Onde |S| é o número total de sentenças do texto e SF(p) o número de sentenças em que a palavra "p" ocorre. A partir da distribuição de relevância de "p" em cada sentença "s", dada por TF-ISF(p,s), pode-se obter o índice de relevância de cada sentença no texto, calculando-se a média TF-ISF de "s".

Larocca Neto *et al.* (2000) propõem que, quanto maior for a média TF-ISF de uma sentença, mais representativa ela será do conteúdo textual. Assim, extratos poderiam ser construídos estabelecendo-se um limite (*threshold*) mínimo para essa média: as sentenças acima do *threshold* comporiam o extrato. Esse valor pode variar dependendo do grau de compressão desejado pelo usuário.

Os seguintes passos são indicados para a aplicação do método, que é parcialmente dependente da língua natural em foco. Larocca Neto *et al.* (2000) trabalham com a língua inglesa. Para trabalhos que contemplam o português, é preciso modificar os módulos dependentes da língua e simplificar a proposta original, devido a limitações de recursos computacionais para o português.

3.2.1 Pré-processamento do texto de entrada

1. ***Case Folding*:** converte todos os caracteres de um documento em um só formato: maiúsculo ou minúsculo. Assim: casa, Casa, cAsa, etc, podem ser igualmente convertidas para o *token* 'casa'. Token em computação é um segmento de texto ou símbolo que pode ser manipulado por um parser, que fornece um significado ao texto; em outras palavras, é um conjunto de caracteres (de um alfabeto, por exemplo) com um significado coletivo.
2. **Remoção de *Stop Words*:** Chamam-se *stop words* as palavras de classe fechada (i.e., preposições, conjunções, artigos, numerais, etc.) ou outras, que, no algoritmo de mineração de texto, não tenham função para a identificação da relevância de segmentos textuais. Assim, considera-se que elas podem ser removidas do texto-fonte antes deste ser processado para a SA propriamente dita. Assim, uma lista de *stop words*, ou *stoplist*, constitui um conjunto variável de palavras da língua natural em foco. No inglês, por exemplo, *can*, *will*, *do* e *does* poderiam ser elementos de uma *stoplist*. Para o português, a *stoplist* considerada é a seguinte: {a, e, o, as, da, de, do, na, no, os, ou, em, um, das, dos, mas, nas, nos, por, que, quê, seu, uma, como, essa, esse, mais, nada, pela, seus, porque, porquê, dessa, dessas, esses, desse, desses, desta, deste, nosso, nossos, aquilo, aquele, aquela, aqueles, aquelas, ninguém, ainda, através, menos, porém, contudo, todavia, entretanto, entanto, se, não, é, etc...}.
3. **Lematização:** Determinação do radical de cada uma das palavras. De forma geral, esta etapa consiste em eliminar sufixos das palavras. Essa etapa é dependente da linguagem natural em foco.

Para o pré-processamento de textos-fonte em inglês, essas três etapas são executadas nessa ordem, sendo que as duas últimas podem, ainda, ser combinadas. Já para o correspondente processo em línguas de origem latina, nem sempre é possível estabelecer algoritmos de lematização apropriados. Neste caso, um processamento baseado em *n-grams* poderia substituí-lo. Um *n-gram* é uma parte de uma palavra consistindo de "n" caracteres. Por exemplo, a palavra DATA pode ser representada pelos trigramas _DA, DAT, ATA, TA_ ou pelos bigramas _D, DA, TA, entre outros. Radicais e *n-grams*, aqui, servem ao mesmo propósito: o de uniformizar a representação dos termos textuais para busca de termos relevantes para a SA.

3.2.2 Delimitação de segmentos textuais

Em geral, a unidade textual mínima considerada é a sentença e, portanto, utilizam-se os símbolos usuais da própria língua natural para delimitá-la (por exemplo, '.', '!', '?', seguidos ou não por um espaço em branco ou por um caractere que indique a quebra da linha sob análise).

Para a sentença S1 abaixo, por exemplo, os passos descritos acima resultam no fragmento S1'.

S1: Diamantino arremessou a bola.

S1': diamant arremess bol

3.2.3 Atribuição de pesos a cada palavra do segmento textual

A esta altura, o texto já fragmentado está disponível para o cálculo da freqüência de seus componentes e, portanto, para a identificação de seus segmentos relevantes. Para tanto, produz-se um vetor em que cada coordenada é dada por um par no formato [palavra,(TF,SF)], que armazena os valores individuais de freqüência de cada componente sentencial (freqüência do termo e freqüência na sentença). O vetor correspondente a S1', por exemplo, é dado por V1:

$$V1 = [[diamantino,(1,8)],[arremessou,(1,1)],[bola,(1,3)]]$$

3.2.4 Cálculo do peso TF-ISF de cada palavra da sentença

O peso de cada componente é calculado segundo a fórmula (1) e armazenado no vetor V1' (ilustrado abaixo), substituindo-se o par (TF,SF) de V1. Para S1', o vetor V1' atualizado resultaria, assim, em (valores fictícios):

$$V1' = [[diamantino,0.60],[arremessou,0.35],[bola,0.40]]$$

3.2.5 Cálculo da média TF-ISF para cada uma das sentenças

A média TF-ISF de cada sentença do texto é calculada com base na freqüência inversa de seus componentes, para se determinar sua freqüência relativa no texto. Assim, a média de S1 é dada por:

$$Média \quad S1 = \frac{(peso[diamantino] + peso[arremessou] + peso[bola])}{n^{o} \ total \ de \ palavras \ da \ sentença} \qquad (3)$$

Para os valores ilustrados em V1', essa média seria, portanto, Média S1= 0.45.

Após o cálculo da média de relevância de cada sentença no seu contexto textual, parte-se para a produção automática do extrato, descrita nos próximos passos.

3.2.6 Cálculo da MAX média

A MAX média é utilizada para calcular o limitante inferior para a seleção das sentenças do extrato em construção (vide próximo passo). MAX é obtido a partir da maior média TF-ISF (sempre no intervalo 0-1), de todas as sentenças do texto. Assim, para um texto exemplo composto por cinco sentenças (S1, S2, S3, S4, S5) abaixo e suas respectivas médias, MAX=0,6 (de S3).

Média TF-ISF(S1) = 0,45
Média TF-ISF(S2) = 0,25
Média TF-ISF(S3) = 0,60
Média TF-ISF(S4) = 0,17
Média TF-ISF(S5) = 0,56

Figura 3.1 - Texto Exemplo

3.2.7 Cálculo do limitante inferior para seleção de sentenças

Este constitui o cálculo numérico final para aplicação do método e pode ser feito com a intervenção do usuário ou gerente do sistema. Ao sugerir o comprimento desejado para seu extrato, por exemplo, "C", esse cálculo será baseado em uma aproximação de "C" que seja viável para se determinar o número de sentenças do extrato. Ou seja, "C" indica um limitante para a compressão do texto-fonte: quanto mais próximo de zero, maior será o extrato e, logo, menor o índice de compressão; quanto mais próximo de um, maior será o índice de compressão do texto-fonte.

O limitante é calculado pela fórmula:

$$Limitante = C * MAX \tag{4}$$

Supondo que o usuário queira um comprimento aproximado de seu extrato de C = 0,7, o limitante do texto ilustrado será:

$$\text{Limitante} = 0,7 * 0,6 = 0,42$$

3.2.8 Produção do extrato

A partir do limitante, basta extrair as sentenças do texto-fonte, em função de seu valor médio TF-ISF: para cada sentença S, se média(S) >= Limitante, S é

selecionada para compor o extrato. Assim, para o exemplo acima, as sentenças selecionadas serão S1, S3 e S5, levando ao extrato da Figura 3.2, composto por sua simples justaposição na ordem em que aparecem no texto.

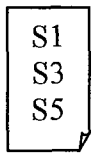


Figura 3.2 - Extrato Exemplo

3.3 Mineração de Utilização

A mineração de utilização da Internet procura descobrir os padrões de navegação dos usuários dos servidores da Internet. Esta idéia foi apresentada inicialmente por CHEN *et al.* (1996), Mannila & Toivonen (1996) e Yan *et al.* (1996). Os dados utilizados na descoberta desses padrões estão concentrados principalmente nos *logs* dos servidores da Internet, que armazenam as interações dos usuários com as páginas neles armazenadas, mas também podem ser encontrados nas próprias estruturas dos *sites* (informações sobre as referências e *links* entre as páginas) ou obtidos dos usuários a partir do uso de programas CGI, *cookies*, agentes e outros mecanismos.

A análise desses dados pode ajudar organizações a traçar estratégias de mercado para produtos, a avaliar o desempenho de campanhas promocionais, entre outras coisas. A análise do *log* de acesso e dos dados de registro dos usuários pode também fornecer informações preciosas sobre qual a melhor forma de estruturar um *site* da Internet com o objetivo de criar uma organização que otimize a navegação dos usuários na busca pelas informações que deseja.

Em organizações que usam tecnologia de intranet, esse tipo de análise pode mostrar a maneira mais eficiente de gerenciamento de comunicação entre o grupo de trabalho e de gerenciamento de infra-estrutura organizacional. Finalmente, para organizações que vendem anúncios na Web, a análise dos padrões de acesso de usuários ajuda a direcionar os anúncios a grupos de usuários específicos.

A maioria das ferramentas de análises da Web (CUSTOMERCENTRICS, 2002; SANE, 2002; ANALOG, 2007; WEBTRENDS, 2007) fornece mecanismos para informar a atividade dos usuários nos servidores e várias formas de filtragem de dados. Usando esse tipo de ferramentas é possível, por exemplo, determinar o número

de acessos ao servidor e os arquivos individuais dentro do espaço virtual das organizações, intervalos de tempo de visitas, e os domínios e as URLs (W3C, 2001) dos usuários do servidor de Internet. No entanto, em geral, essas ferramentas são projetadas para lidar com tráfico de servidores baixos a moderados e, além disso, elas normalmente fornecem pouca ou nenhuma análise de dados de relacionamento entre os arquivos e diretórios acessados no espaço virtual.

Segundo Spiliopoulou *et. al* (1999) existem duas faces a serem analisadas no comportamento do visitante de um *site*:

- Seus interesses e as informações que acessa;
- A maneira pela qual essas informações são acessadas.

Para uma análise do primeiro aspecto, podem ser utilizados questionários, pesquisas e outros mecanismos que permitam a classificação dos usuários segundo diferentes perfis, o que não se caracteriza propriamente como análise de utilização da Internet. No segundo aspecto, contudo, entra em cena a investigação do comportamento de navegação do usuário, que pode ser feito a partir da interpretação dos *logs* de uso gravados pelos servidores da Internet. Esses dois aspectos podem ser considerados complementares, e é dessa forma que as duas técnicas são empregadas nessa dissertação.

A mineração de utilização da Internet pode ser classificada em dois ramos principais (COOLEY *et al.*, 1997):

- **Descoberta de padrões de acesso gerais:** que procura analisar os *logs* em busca de padrões e tendências de utilização genéricas.
- **Descoberta de padrões customizados:** que tenta identificar os padrões específicos de um determinado usuário, a fim de se adaptar o próprio servidor da Internet ao mesmo. Este ramo está intrinsecamente ligado ao desenvolvimento de *sites* adaptativos (PERKOWITZ, ETZIONI, 1997; 1998; 1999; MAEDCHE *et al.*, 2001).

Cooley (*et al.*, 1997; 2000) divide a mineração de utilização da Internet em pelo menos três etapas distintas, cada uma delas com suas próprias características, procedimentos, métodos, entradas e saídas.

1. Preparação de dados é a etapa onde as diferentes fontes de dados de utilização da Internet são lidas, filtradas, limpas e integradas. Dependendo das necessidades da aplicação, esta fase pode, ou deve,

identificar os usuários e as suas sessões (conjunto de acessos a páginas), caso as fontes de dados que não contenham uma correta ou precisa identificação. Pode ser necessária, também, a identificação de transações de usuários;

2. Descoberta dos padrões de utilização é a etapa onde são aplicados métodos e algoritmos de mineração de dados em busca de padrões úteis e significativos;
3. Análise e visualização dos padrões é a etapa que disponibiliza e identifica os principais padrões encontrados, a partir das preferências e definições dos usuários.

3.3.1 Preparação de Dados

Com um conjunto de informações preciso, detalhado e bem organizado, a mineração de padrões deveria ser feita sobre um conjunto de sessões ou transações de usuários, contendo informações sobre quem acessou que *site*, que páginas foram visitadas e em que ordem, e por quanto tempo o usuário permaneceu em cada página (COOLEY *et al.,* 1999). Considera-se uma sessão de usuário como composta por todas as páginas acessadas por um usuário em uma determinada visita a um *site*. Uma transação é um agrupamento semanticamente significativo de páginas, como será visto mais à frente.

No entanto, a diversidade de fontes de dados e o fato destas fontes nem sempre serem precisas e detalhadas como se desejaria, a atividade inicial a ser realizada antes de qualquer processo de mineração de utilização deve ser sempre o pré-processamento dos dados a serem minerados, incluindo aí o desenvolvimento de um modelo de dados para os *logs* de acesso, a filtragem e limpeza dos dados brutos, a identificação de usuários, sessões e transações.

As fontes de dados de utilização de um *site Web* podem ser produzidas por agentes autônomos ou outras interfaces que façam o registro direto das ações dos usuários e que incluam uma correta e precisa identificação dos mesmos, assim como de suas sessões (SHAHABI *et al.* 2001). Em outros casos, os dados de uso poderiam ser gravados em arquivos ou bancos de dados por páginas com scripts que utilizassem *cookies* ou outros mecanismos tais como identificadores de sessões (HALLAM-BAKER, CONNOLLY, 1996a). Nestes casos, a identificação posterior de usuários e sessões será desnecessária. O mesmo não se pode dizer, porém, da filtragem dos

acessos, já que possivelmente estarão sendo gravadas visitas não só às próprias páginas da Internet, como também a arquivos de imagens, *postscript*, etc.

Vale ressaltar que a utilização de mecanismos de identificação explícita pode levar às mesmas preocupações de privacidade que serão abordadas mais adiante. Entretanto, as principais fontes de dados de utilização de um *site* são os próprios *logs* brutos gerados pelos servidores da Internet, devido, principalmente, à facilidade de obtenção dos mesmos. Os principais servidores da Internet do mercado disponibilizam *logs* de diversos tipos, registrando cada um dos acessos realizados pelos usuários ao visitarem os *sites*.

Tais *logs*, porém, possuem sérias deficiências quanto ao detalhamento e reconhecimento dos usuários e sessões, a ponto de alguns estudos argüirem que eles nem mesmo podem ser considerados como fontes válidas para a descrição e análise dos padrões de utilização de um *site*. Os dados contidos em um *log* de servidor da Internet não representam com total confiabilidade as sessões dos usuários devido a uma série de fatores:

- A presença de grande número de itens irrelevantes
- A ausência de identificação única dos usuários, sessões e transações.
- A inexistência dos registros das visitas a inúmeras páginas, devido a fatores tais como o uso de *cache* e servidores *proxies*.

3.3.1.1 Filtragem dos dados

Muitas vezes as fontes de dados utilizadas para mineração de utilização contêm muitas informações irrelevantes, tais como acessos a arquivos de imagens, som, vídeo, animações, etc. Isso torna fundamental que seja feita uma limpeza ou filtragem das fontes. Além disso, é importante salientar que tal limpeza é útil para qualquer tipo de análise dos *logs* de servidores da Internet, não somente para a mineração de utilização.

Os *logs* de servidores da Internet apresentam vários formatos, o que é uma séria preocupação a ser levada em conta por uma ferramenta de mineração de utilização da Web que faça a filtragem de dados. Alguns trabalhos (por exemplo, COOLEY *et al.*, 1999) analisam apenas arquivos que estejam no formato padrão *Common Log Format* – CLF (LUOTONEN *et al*, 1995), especificado pelo CERN e NCSA como parte do protocolo HTTP. Este padrão, por ser bastante limitado em termos das informações que armazena, foi ampliado posteriormente pelo W3C para o

Extended Log Format – ECLF (HALLAM-BAKER, BEHLENDORF, 1996), que adiciona, por exemplo, informações sobre a referência de origem da página (referidor ou *referrer*), ou seja, a página de onde partiu o acesso. Ao se utilizar o ECLF, é possível que se especifiquem quais os campos que se deseja gravar no *log*.

Uma entrada de *log* de utilização contém, normalmente, o registro do percurso de uma página de origem até uma página de destino, incluindo o IP da máquina cliente que originou a chamada, o tipo de acesso (POST ou GET) realizado, além de outros dados, a depender do padrão utilizado pelo servidor da Internet.

No protocolo HTTP (FIELDING *et al.,* 1997), um cliente faz ao servidor uma requisição para cada arquivo necessário à visualização de uma determinada página. Assim, um acesso a uma simples página da Internet provoca a gravação de várias entradas de *log* no servidor, para cada uma das imagens, scripts ou outros arquivos carregados juntamente com a página. Em geral, somente as entradas de *log* associadas aos acessos às páginas HTML serão de interesse na mineração de utilização, pois os demais arquivos, especialmente imagens, são baixados automaticamente à revelia do usuário. Assim, nem sempre serão úteis para um sistema que procure minerar os padrões de navegação do usuário, já que não foram explicitamente solicitados por este.

Para remover imagens, uma abordagem simples é retirar do *log* todas as entradas associadas às extensões conhecidas para elas, tais como GIF, JPG, JPEG, etc. O mesmo pode ser feito em relação a arquivos de sons ou outras fontes multimídia. O ideal, contudo, é que o sistema permita a configuração de quais serão os sufixos de arquivos que devem ser ignorados no processo de limpeza (COOLEY *et al.,* 1999). O analista, muitas vezes, pode estar interessado em acessos explícitos a determinados tipos de arquivos ou imagens. Pode haver até mesmo, adicionalmente, uma lista com os nomes de arquivos que devam ser explicitamente ignorados na mineração.

3.3.1.2 Identificação de usuários

Uma vez que o *log* foi filtrado, o próximo passo a se considerar é a identificação de usuários. Notadamente, os *logs* de servidores da Internet são bastante incompletos em relação a isto. Várias dificuldades podem ser levantadas, principalmente o uso de *cache* e a presença cada vez maior de servidores *proxy* (PIROLLI *et al.*, 1996; PITKOW, 1997). Estes dois fatores podem distorcer bastante os dados dos *logs* no que diz respeito à identificação dos usuários.

O *cache* local do navegador faz com que a mesma página possa ser acessada várias vezes pelo usuário, sem que o servidor da Internet registre no *log* tais acessos. O *cache* local do navegador é um método para tornar mais eficiente o acesso a páginas muito visitadas pelo usuário e que não sofram muitas modificações. A página fica armazenada no disco local, e, com isso, nas próximas vezes em que for acessada, não será feita uma nova requisição ao servidor da Internet.

O uso de servidores de *cache* em muitos provedores de acesso torna esse problema anterior ainda mais grave. Nesse caso, as páginas de maior utilização de **todos** os usuários do provedor serão armazenadas para uso futuro. Com isso, mesmo que um dado usuário nunca tenha acessado uma determinada página, ao fazê-lo pela primeira vez não será gerada uma entrada no *log* do servidor da Internet onde ela se localiza, caso a mesma já tenha sido armazenada no servidor de *cache* do provedor, por ter sido muito acessada por outros usuários.

Para contornar os problemas causados pelo uso de *cache*, pode-se forçar, na página da Internet, o "*by-pass*" do *cache*: obriga-se o navegador a recarregar a página sempre que ela for visitada, num processo conhecido como "*cache busting*". No entanto, essa opção pode ser desabilitada pelo usuário e, além disso, é um mecanismo que termina por tirar a principal vantagem do uso do *cache*, que é a maior velocidade de navegação (SHAHABI *et al.*, 1997).

Os servidores *proxy* também colaboram para tornar mais difícil a tarefa de identificação do visitante, pois o mesmo número IP em diversas entradas de *log* pode estar associado a diferentes máquinas clientes. O servidor *proxy* é uma maneira eficiente e segura que muitas organizações encontram para compartilhar o uso de um número IP por diversas máquinas da sua intranet. Assim, para os computadores da rede externa, todas aquelas máquinas serão vistas como uma só, já que utilizam o mesmo número.

Algoritmos ou estratégias que permitem testar diferentes combinações de números IP, nomes de máquina e informações temporais para se identificar o usuário foram desenvolvidos para contornar esse problema (PITKOW, 1997).

Pirolli *et al.* (1996), por exemplo, propõem que se identifique uma mudança de usuário sempre que houver uma mudança nas entradas do *log* para os campos que guardam o agente, software cliente ou sistema operacional, ainda que essas entradas provenham do mesmo número IP. Naturalmente, nesse caso, é necessário que os *logs* de utilização estejam gravando essas informações, o que nem sempre é o caso. Podem

ainda ser feitas considerações sobre o intervalo de tempo entre dois acessos consecutivos ou sobre quais páginas deveriam ser acessadas por duas entradas consecutivas para serem consideradas como originadas pelo mesmo usuário.

Uma outra heurística possível utiliza a topologia do *site* combinada ao *log* na tentativa de reconstituir os caminhos percorridos por cada usuário (PIROLLI *et al.*, 1996). A extração da topologia, ou estrutura do *site*, pode ser feita com o uso de *crawlers*. Se, pela análise da topologia do *site*, uma página visitada não puder ser acessada através de qualquer seqüência de *links* que parta de outra página acessada anteriormente, então isso indicará que esta visita foi realizada por um segundo usuário utilizando o mesmo IP.

Contudo, há limitações para tais análises: por exemplo, se dois usuários com o mesmo navegador, mesmo número de IP e a partir do mesmo tipo de máquina acessarem o mesmo conjunto de páginas, a probabilidade de ambos serem confundidos é grande. Por outro lado, um usuário com dois navegadores na mesma máquina ou que digite diretamente as URLs no campo de endereço do navegador poderia ser erroneamente tomado como múltiplos usuários.

A identificação do usuário pode ainda ser facilitada com o uso de *cookies*, da mesma forma que nas fontes de dados geradas por agentes outros, diferentes dos servidores da Internet. As entradas dos *logs* terão, assim, armazenadas as informações sobre quais *cookies* foram utilizados nos acessos.

Outra possibilidade é o registro explícito do usuário, que será convidado a entrar com os seus dados pessoais e senha sempre que iniciar a navegação por um determinado *site*. Assim, os *logs* registrariam também um identificador do usuário. Esta solução também é semelhante à adotada em outras fontes de dados, como visto anteriormente. O registro explícito é uma solução colaborativa; sem dúvida, a solução mais simples de se utilizar, mas nem sempre a mais factível.

Quando os usuários requerem explicitamente o uso do sistema, a identificação das sessões é bastante simplificada. Além disso, não existem aí as mesmas questões de privacidade advindas do uso de *cookies* e outros agentes, já que o usuário escolhe utilizar o sistema em troca dos seus benefícios. Contudo, no caso de sistemas que não forneçam um *feedback* imediato, o registro das ações do usuário levanta mais uma vez as mesmas questões.

Shahabi *et al.* (1997) mostram uma outra solução em que é enviado ao cliente um agente Java que será responsável por mandar de volta ao servidor informações precisas sobre a navegação do usuário.

Pitkow (1997) questiona, porém, que essas soluções possuem limitações. Os *cookies*, por exemplo, podem ser removidos pelo usuário, ou podem ser desabilitados na configuração do navegador. Finalmente, o registro explícito, apesar das vantagens que traz pelas informações demográficas adicionais que oferece, levanta questões de privacidade que podem fazer com que muitos usuários desistam de navegar num *site* ou mesmo forneçam informações incorretas.

Haja vista todas essas limitações, deve-se apelar, na maioria das vezes, a heurísticas que permitam identificar se uma requisição de página gravada no *log* veio do mesmo usuário. Caso a melhor solução seja o registro explícito dos usuários, torna-se necessário trabalhar a compreensão dos usuários a fim que contribuam de boa vontade com a mineração. Em caso contrário, as análises serão prejudicadas.

3.3.1.3 Identificação das sessões

O processo de identificação das sessões é semelhante ao de identificação dos usuários. Considera-se a sessão como uma passagem completa de um usuário por um *site*, desde a página por onde ele iniciou a passagem até a última página acessada. A sessão, portanto, inclui todas as páginas percorridas em uma visita completa do usuário ao *site*. Dessa forma, o problema da identificação das sessões começa na identificação dos usuários.

O método mais simples para a identificação de sessões individuais de cada usuário é utilizar um *time-out* de controle. De acordo com esse método, sempre que o tempo entre dois acessos consecutivos para um determinado usuário for maior do que este *time-out*, assume-se que ele iniciou uma nova sessão. Muitos produtos comerciais utilizam, como *time-out,* 30 minutos; Catledge & Pitkow (1995) chegaram ao valor de 25,5 minutos, a partir de dados empíricos. Após a análise estatística de um *site*, pode-se calcular um valor apropriado para este tempo, que será utilizado pelo algoritmo de identificação de sessões.

O método de *time-out* pode ser complementado com outra técnica, como demonstrado em Spiliopoulou *et al.* (1998; 1999). Além de considerar o *time-out* entre dois acessos consecutivos, assume-se também que uma nova sessão é iniciada quando a duração total de uma seqüência de acessos exceder um determinado limiar.

O uso de cachê ou de servidores *proxy* volta a causar problemas para a análise de navegação uma vez que dificultam a descoberta dos acessos que não foram registrados nos *logs* e conseqüentemente dificultam na identificação de sessões (COOLEY *et al.*, 1999). Ainda que se possa aqui utilizar soluções semelhantes, tal como a supressão do *cache*, pode-se também adotar um algoritmo de "completamento" de caminhos para tentar suprir essas lacunas. Por exemplo, se for acessada uma página não diretamente alcançável a partir da página imediatamente anterior, pode-se assumir que o usuário usou o botão de "VOLTAR" do navegador, passando a percorrer páginas que estavam no seu *cache*, até que, a partir de uma dessas páginas anteriormente visitadas, acessou a página atual.

Além disso, o usuário do navegador tem à sua disposição a opção de recarregar a página atual, o que fará com que uma nova entrada seja registrada no *log* do servidor. Isto, juntamente com os freqüentes retornos a páginas em *cachê,* torna ainda mais problemática a análise do *log*.

3.3.1.4 Identificação de transações

Para a próxima fase de descoberta de padrões será necessário descobrir regras de associação através da identificação transações. Uma transação difere da sessão por ser um agrupamento semanticamente significativo de referências de páginas (COOLEY *et al.*, 1999). Dessa forma, a transação pode considerar todas as páginas acessadas em uma sessão, algumas ou apenas uma delas. Esse conjunto será definido através dos critérios usados para identificá-la.

Cooley *et al.* (1997a; 1999) apresenta duas maneiras que podem ser empregadas para tentar identificar as transações a partir dos diferentes tipos de páginas (de navegação e de conteúdo):

- Considera-se uma transação como o conjunto de acessos sucessivos a páginas auxiliares, até se chegar a uma página de conteúdo, que indica o final da transação.

- A transação inclui todas as páginas de conteúdo (e somente elas) visitadas pelo usuário.

A mineração, na primeira abordagem, revela os caminhos comuns até uma determinada página, através da análise das chamadas transações de navegação ou de transações de navegação para conteúdo. Na segunda, mostra os relacionamentos e

associações entre as páginas de conteúdo, sem se interessar pelos caminhos que conduzem até elas, analisando apenas as transações de conteúdo.

Cooley *et al.* (1997a; 1999) comparam três principais métodos de identificação de transações:

- A identificação por duração da referência supõe que o tempo gasto numa página correlaciona-se com o fato desta ser uma página de conteúdo ou de referência para o usuário.
- A identificação por referências posteriores máximas divide as sessões dos usuários em transações e considera que uma transação é composta por uma seqüência de páginas visitadas a partir de uma página inicial até a última página acessada imediatamente antes da próxima referência reversa (CHEN *et al.*, 1996). Uma referência reversa é definida como uma página já presente no conjunto de páginas visitadas na transação. Por sua vez, uma referência posterior é uma página ainda não visitada na transação. Uma referência posterior máxima é, portanto, a última página acessada pelo usuário antes dele tentar voltar para uma página já visitada na transação, denotando assim o final desta.
- A identificação por janelas de tempo particiona a sessão em intervalos com duração menor que um determinado parâmetro. Aqui, não se utiliza o modelo de páginas de conteúdo e auxiliares, mas sim a suposição de que uma transação tem um tempo médio de duração. Por isso, não serão produzidos os dois tipos diferentes de transações encontrados pelos métodos anteriores.

3.3.2 Descoberta de padrões

Existem várias técnicas de mineração de utilização da Web já consagradas e largamente empregadas que podem ser adaptadas para serem empregadas nas análises desejadas. Uma vez que as sessões ou transações foram identificadas, essas técnicas podem ser utilizadas para análises estatísticas, análises dos caminhos percorridos, descoberta de regras de associação e padrões seqüenciais, agrupamento e classificação.

As análises estatísticas são em geral as mais simples e procuram simplesmente dados de caráter geral, tais como o número de *hits* por página, as páginas acessadas

mais freqüentemente, as páginas mais usadas como ponto de partida ou de saída no *site*, o tempo médio de acesso de cada página, etc. Por isso são as análises mais comuns de serem encontradas em ferramentas existentes.

A análise dos caminhos pode gerar grafos direcionados onde cada nó é uma página e cada aresta representa uma referência entre as páginas; ou poderia representar similaridades entre páginas; ou o número de usuários que saíram de uma para outra página. Pode-se tentar determinar padrões de como determinado conjunto de páginas é percorrido por um grupo de usuários, ou seqüências longas de referência a partir da estrutura dos grafos obtidos, assim como os caminhos mais percorridos, além de outros padrões úteis.

Quando tentamos descobrir regras de associação em um banco de dados de transações, que é composto por um conjunto de itens, o objetivo é na verdade descobrir quando a presença de um determinado conjunto de itens implica na presença de um outro item na mesma transação. Quando essa análise recai sobre a utilização das páginas armazenadas nos servidores da Internet, cada item representa uma página acessada e uma transação é o conjunto de acessos de um usuário em uma determinada visita ao servidor.

Quando um banco de dados de navegação na Web é analisado para o reconhecimento de regras de associação os antecedentes das regras a serem avaliados incluem não apenas as páginas individuais, mas também subconjuntos (sem ordenação) de páginas e subseqüências ordenadas das mesmas. Além disso, acrescenta um critério para a seleção de regras baseado na comparação dos comprimentos das páginas (LI, 2001).

Outra técnica largamente empregada e que se mostra muito útil na fase da mineração de utilização é a descoberta de padrões seqüenciais. Essa técnica, quando empregada na Internet pode mostrar que certo percentual de usuários que acessaram determinada página num servidor também fizeram uma compra on-line em uma outra página no intervalo de uma semana. Além disso, esse padrão pode ser associado ao tempo e, dessa forma, determinar qual o intervalo em que um grupo de páginas foi mais acessado, ou procurar as características em comum dos clientes que visitaram estas páginas em um intervalo específico.

Além dos padrões mais triviais, deve-se também procurar descobrir padrões de acesso com propriedades estatísticas interessantes (SPILIOPOULOU *et al.*, 1999). A dominância estatística não é sempre interessante, já que os padrões dominantes quase

nunca acrescentam conhecimento a um analista experiente. As características que tornam um padrão interessante são de caráter mais genérico, tais como caminhos quase nunca percorridos ou que percorram páginas com um assunto em comum.

Em Srikant & Agrawal (1996) observamos que existe uma diferença fundamental entre a mineração de utilização da Web e a mineração de dados tradicional que dificulta a adaptação das técnicas tradicionais para serem utilizadas na Web. O principal objetivo das ferramentas de mineração convencionais é descobrir as seqüências mais freqüentes. No caso da mineração de utilização é conveniente que seja possível identificar seqüências raras, porém confiáveis. Uma forma de tentar contornar esse problema é remover as seqüências não interessantes em sessões consecutivas de mineração e pós-mineração (ZAKI *et al.*, 1998).

Nesta fase podemos empregar ainda técnicas de classificação e agrupamento, não só para reunir as páginas semelhantes entre si, mas também para fazer o mesmo com as seqüências de páginas. Dessa forma, a análise dos padrões de navegação é facilitada e é permitindo a comparação destes com os perfis de usuários, se disponíveis. Com o algoritmo RDBC (*Recursive Density Based Clustering*) (SU *et al.*, 2001) é possível fazer o agrupamento de páginas com base na freqüência de suas utilizações, e não no seu conteúdo. Ele permite ainda classificar as páginas visitadas num servidor a partir de informações demográficas sobre os usuários.

COOLEY *et al.* (1999) defendem o uso de um filtro de *sites* que reduza o número de regras inúteis encontradas e otimize o tempo de processamento dos algoritmos de mineração. O uso desse filtro pode diminuir as medidas de suporte e confiança dos algoritmos de mineração, a fim de aumentar a quantidade de regras e padrões úteis. Além disso, ele pode auxiliar a apontar certas características de uso do *site*.

O filtro poderia, por exemplo, checar se a página inicial está realmente sendo utilizada como ponto de partida pelos usuários, ou se outras páginas o estão; ou poderia ignorar as regras triviais, tais como uma regra que apenas confirme um link direto entre duas páginas; ou, ao contrário, perceber a falta de uma regra esperada, no caso contrário, quando tal *link* direto não está sendo utilizado pelos visitantes.

3.3.3 Análise dos padrões

Uma vez que os padrões de utilização foram extraídos da base de dados de navegação, inicia-se a análise desses padrões. Para isso, o usuário deve lançar mão de

programas estatísticos, gráficos de visualização e consulta. Com essas informações os projetistas podem, por exemplo, comparar o conhecimento descoberto sobre a utilização dos seus *sites* com a perspectiva que possuíam sobre como ele deveria ser utilizado.

Kato *et al.* (2000) apresentam uma ferramenta que analisa a relevância conceitual entre páginas e a conectividade dos seus *links*, para avaliar as expectativas do projetista ao desenhar o *site*. Medindo a co-ocorrência de acessos entre diferentes páginas, a ferramenta avalia os padrões de uso dos visitantes. Dessa forma, ela auxilia o projetista na análise dos padrões de utilização disponibilizando de maneira gráfica os resultados das análises, através de um sistema de coordenadas polares que torna fácil a comparação das trilhas seguidas pelos usuários com os problemas detectados, dando pistas bastante úteis de como o *site* pode ser reestruturado.

Outra ferramenta chamada WebViz (PITKOW, BHARAT, 1994) utiliza um paradigma baseado nos percursos ou caminhos da Web (as seqüências de *links* entre as páginas), pelos quais se extraem, a partir dos *logs* de utilização, subseqüências dos padrões como as páginas são percorridas, os chamados *Web paths*. Dessa forma, a Web é vista como um grafo cíclico direcionado, no qual os nós são páginas e as arestas são referências entre elas. O usuário pode, assim, visualizar e analisar os trechos de seu interesse, desprezando os irrelevantes.

Segundo Dyreson (1997) e Kimball & Merz (2000), os dados de utilização da Web possuem muito em comum com aqueles de um *data warehouse*. Isso porque os dados de utilização são sempre agregados ao final dos *logs*, os quais crescem rapidamente ao longo do tempo, impossibilitando a análise de todo ele, e levando à necessidade de sumarização para fins de desempenho. Outra semelhança é a existência de requisitos de segurança, pois certas porções dos *logs* não devem ser vistas pelo analista. Essas semelhanças implicam na grande influência que as técnicas de *data warehousing* e as técnicas associadas de OLAP (ZAÏANE *et al.*, 1998) têm sobre a análise dos padrões encontrados.

Foi exatamente essa influência que levou Kimball & Merz (2000) a apresentarem *"data webhousing"*, uma abordagem do processo de mineração de utilização da Internet focada essencialmente em *data warehousing*. O trabalho não só apresenta temas comuns aos outros trabalhos de mineração de utilização, fazem-no de modo integrado, incluindo a utilização de técnicas e ferramentas OLAP, como também mostram o desenvolvimento de dois esquemas complementares de *data*

warehouse em estrela para o armazenamento de dados minerados dos *logs* dos servidores da Internet. Num deles, é criada uma tabela fato para representar cada acesso ou clique do visitante; no outro, a tabela fato representa as sessões.

3.4 Folksonomia

A pesquisa para utilização de *Web* Semântica como forma de otimização das buscas de informações na Web conta com a ajuda de uma técnica simples, porém poderosa: o *bookmarking* social. Trata-se do registro de URLs de "favoritos" (*bookmarks*) em *sites* como del.icio.us (DELICIOUS, 2007) e Technorati (2007). Porém, o diferencial destes serviços da mera listagem de apontadores em uma página *online* é o processo de geração de metadados (ou seja, dados sobre dados) através da associação de *tags* (etiquetas) a referências e materiais.

No *tagging*, em vez do cadastramento padronizado de informações como "autor" e "ano de publicação", os internautas ao incluírem um novo *link* em sua lista pública de *bookmarks* podem registrar quaisquer palavras que julgarem ser associadas a um dado material. Com técnicas de *tagging* permite que para cada endereço armazenado, o software de marcadores ofereça a oportunidade para que o usuário dê um conjunto de marcadores ("*tags*") para o endereço. Esse processo vem sendo chamado de "folksonomia", neologismo criado pelo arquiteto de informação Vanderwal (2005) a partir dos termos *folk* e taxonomia. Ou seja, em vez de uma categorização por especialistas que segue rígidos padrões taxonômicos, a folksonomia seria uma classificação social de "baixo para cima".

Segundo Mathes (2004), a folksonomia representa uma mudança fundamental, pois é derivada não de profissionais ou criadores de conteúdo, mas de usuários de informações e documentos. Desta forma, ela diretamente reflete as escolhas de enunciação, terminologia e precisão.

As *tags* vêm sendo usadas não apenas para conferir significado para a quantidade de textos na Web, mas também para facilitar o registro e recuperação de imagens. O *site* de publicação de conteúdo pessoal Multiply (Disponível em http://multiply.com) oferece o mesmo sistema classificatório, permitindo que cada pessoa crie etiquetas de classificação para suas imagens digitais ou textos a partir de livres associações (e não de um vocabulário controlado, como na taxonomia). Por exemplo, uma foto do pôr-do-sol em uma praia na Tailândia pode ser arquivada no *site* com as *tags* "praia", "Tailândia", mas também "beleza", "férias" e até mesmo

"vermelho". A partir dessas *tags,* outro internauta buscando fotos de tons avermelhados para a produção de um *site* sobre turismo poderá recuperar tal imagem.

Dessa forma, podemos concluir que a combinação de sistemas de redes sociais com o desenvolvimento de folksonomias baseadas em *tags* e disponibilizadas através de *Blogs* e *Wikis* cria uma base natural para um ambiente semântico.

No entanto, não podemos supor que a folksonomia é uma técnica totalmente confiável, pois em geral os sistemas e *sites* que a utilizam não possuem qualquer mecanismo para avaliar a classificação do conteúdo. E mesmo que houvesse intenção de fazer tal avaliação, ela se mostraria inviável tal o gigantesco número de usuários que esses sistemas possuem. Como em todo grupo dessa magnitude, existem aqueles mal intencionados que corrompem informações deliberadamente e prejudicam o grupo. Por outro lado, não podemos ignorar que a abertura para o trabalho coletivo é uma importante ferramenta para motivar aqueles realmente interessados no assunto e dispostos a compartilhar informações.

Assim, a folksonomia se mostra como uma técnica para somar esforços complementando outras técnicas na classificação e recuperação de informações, mas não pode ser encarada como única fonte de informações confiável por mais flexível e impreciso que um sistema seja.

3.5 Conclusão

Nesta dissertação são utilizadas várias das técnicas apresentadas neste capítulo de forma independente e complementar. Os dados brutos são extraídos durante a navegação do usuário. Uma vez com a seqüência de páginas navegadas pelo usuário, bem como o conteúdo de cada página navegada, é possível aplicar as técnicas separadamente para cruzar essas informações no futuro.

O conteúdo de cada página navegada é classificado utilizando uma adaptação da técnica de mineração de conteúdo conhecida como Sumarização Automática (SA) proposta por Larocca Neto *et al.* (2000). Essa técnica é conhecida dessa forma por resumir o texto da página apenas aos termos com maior relevância. O método parte da idéia de que, se uma palavra pode ser significativa em um texto por sua freqüência, ela também pode indicar a importância relativa das sentenças que a contêm. O fato do conteúdo da página estar em código HTML facilita bastante o reconhecimento dos termos de maior relevância.

Paralelamente, as técnicas de mineração de utilização são empregadas a partir das seqüências de URLs navegadas pelo usuário. Essas informações passam por três etapas básicas de mineração: (1) a preparação dos dados que filtra e limpa os dados coletados, separando as visitas relevantes das não relevantes, (2) a descoberta dos padrões de utilização que minera os dados em busca de padrões de comportamento dos usuários e (3) a análise dos padrões encontrados, onde o usuário final do sistema minerador tem a possibilidade de visualizar e tirar conclusões a partir dos padrões encontrados. Esse último tipo de mineração em especial enfrenta dificuldades para identificar usuários, suas sessões de navegação e transações. Neste capítulo apresentamos alternativas para solucionar esse problema.

Esses dois tipos de mineração tornam-se complementares quando o objetivo é encontrar padrões de navegação entre os assuntos das páginas e não entre cada URL especificamente. Assim, a mineração de utilização descobre os padrões de navegação através das páginas ao passo que a mineração de conteúdo é responsável por classificar as páginas, definindo o seu assunto principal. O cruzamento dessas informações fornece os padrões desejados neste trabalho.

O processo de classificação pode ser aperfeiçoado através da utilização de folksonomia para criação de uma base de conhecimento com cooperação dos usuários. Quanto mais páginas forem classificadas pelos usuários, maior a porcentagem de acerto na classificação automática de páginas que não foram previamente classificadas. Acreditamos que essa técnica é empregada com segurança porque é utilizada juntamente com as técnicas de mineração apresentadas neste capítulo.

Capítulo 4 - O Sistema Olimpo

Para alcançar o objetivo deste trabalho, considera-se adequado (ou, é proposto) construir cadeias de conhecimento automaticamente e recomendá-las para o aprendiz. Como foi dito anteriormente, o aprendiz pode aceitar, modificar ou até descartar essas cadeias de conhecimento antes de compartilhá-la com sua comunidade de prática através da ferramenta *Knowledge Chains Editor* (KCE) (SILVA *et al.*, 2006; PEREIRA *et al.*, 2006). Essas cadeias são chamadas neste trabalho de *Information Object*, ou simplesmente IO.

Como o KCE é uma ferramenta integrada ao COPPEER (MIRANDA, 2006), uma ferramenta muito flexível para criação de aplicações ponto a ponto (*P2P*) que oferece ferramentas colaborativas como *plug-ins*, o aprendiz pode compartilhar sua cadeia de conhecimento para outros aprendizes da comunidade.

Para criar uma cadeia de conhecimento através das páginas navegadas, este trabalho propõe a utilização de uma ontologia do domínio considerado pela comunidade. O objetivo é determinar o subgrupo de conceitos navegados (conceitos encontrados nas páginas navegadas) e os relacionar com as páginas. Para os experimentos executados neste trabalho, a ontologia criada tem o papel de uma *upper ontology*, ou seja, apresenta conceitos de alto nível para o domínio de linguagens de programação orientadas a objetos e foi devidamente instanciada para contemplar as especificidades da linguagem de programação Java (SUN, 2006). A criação dessa ontologia será detalhada no capítulo posterior.

Na seção 4.1 apresentaremos a arquitetura de informação proposta para o sistema Olimpo. Arquitetura de informação, usualmente abreviada para AI (não confundir com *Artificial Intelligence* em inglês), consiste na estruturação das informações de sistemas computacionais de forma lógica e na criação de soluções quanto à organização visual destas informações. Envolve a organização do fluxo de informação visando torná-la útil e inteligível. Essa arquitetura é apresentada com diferentes possibilidades de implementação, pois gostaríamos que ela fosse flexível em relação às tecnologias que poderiam ser aplicadas. Variações de tecnologia podem gerar diferentes informações que o sistema poderia oferecer.

Apresentaremos na seção 4.1 os papeis que cada módulo terá na manipulação e apresentação das informações as maneiras que e as diferentes maneiras que ele pode desempenhar este papel. Sempre que estivermos referenciando estes módulos como entidades abstratas da arquitetura de informação seus nomes serão acompanhados de uma sigla que representará sua principal função na arquitetura.

Na seção 4.3 apresentamos como o sistema foi implementado para o estudo de observação realizado neste trabalho, quais decisões foram tomadas e porque essas foram escolhidas. Neste caso os módulos serão referenciados apenas por seus nomes.

4.1 Arquitetura do Olimpo

O sistema possui módulos locais, que são executados na máquina do usuário, e módulos remotos, que são executados em um servidor central de forma que possa agregar as informações de vários usuários. No entanto, a própria estrutura modularizada torna essa arquitetura flexível, permitindo que estes módulos se movam entre clientes e servidores de acordo com a necessidade do trabalho e da disponibilidade de recursos, sem necessidade de muitas alterações em código.

O processo é iniciado com a demanda do usuário de adquirir determinado conhecimento, sendo o interesse alimentado pela necessidade de aprendizado ou pela curiosidade do aprendiz. A real necessidade ou a paixão pelo assunto são muito importantes para motivar o usuário-aprendiz a utilizar o sistema e compartilhar sua experiência. Uma vez que neste cenário, o aprendiz inicia sua busca por informação na Web.

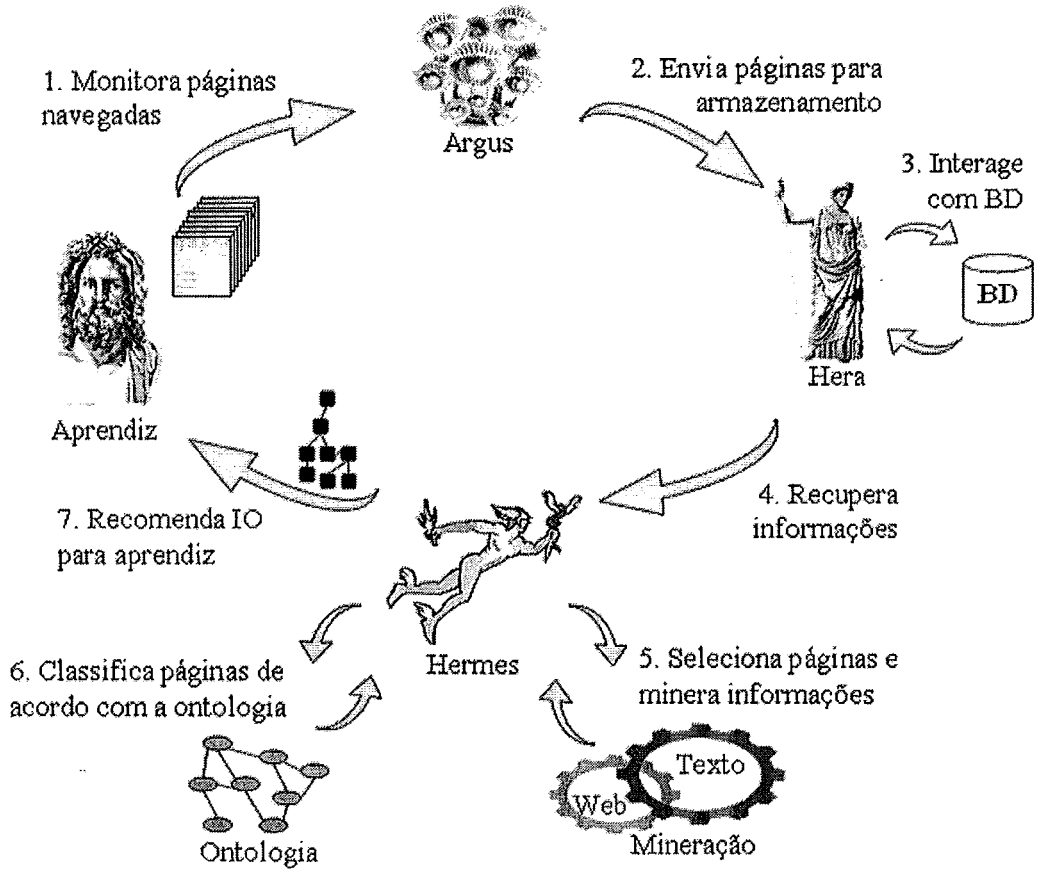


Figura 4.1 - Arquitetura do Olimpo (PEREIRA *et al.*, 2006)

4.1.1 Argus-MMN (Módulo Monitor de Navegação)

Enquanto o aprendiz navega pela Web em busca de informação, o módulo Argus-MMN é responsável por observar e armazenar as informações navegadas, tais como: URL (W3C, 2001), conteúdo, data e hora do acesso, tempo de permanência na página. Este módulo pode ser simplesmente um extrator de informação, capturando cada página visitada independente de sua relevância para a comunidade, ou pode executar uma pré-avaliação das páginas visitadas, decidindo as que são relevantes e armazenando apenas as suas informações.

O Argus-MMN pode ser desenvolvido para que seja executado na máquina de cada usuário da comunidade, interagindo diretamente com o navegador, ou pode ser desenvolvido para analisar um *log* de navegação de um servidor *proxy*, por exemplo. No entanto, a segunda opção pode limitar o armazenamento do conteúdo da página. A maioria dos servidores *proxy* não armazena o conteúdo das páginas visitadas, o que obrigaria o módulo a visitar posteriormente cada página para armazenar seu conteúdo ou avaliar sua relevância, se for desenvolvido para isso. Para páginas desenvolvidas

com linguagens de script como JSP (*JavaServer Pages*) (SUN, 2006) ou ASP (MICROSOFT, 2007), muitas vezes só é possível obter o código HTML (W3C, 1999) seguindo determinada navegação pelo *site*. Isso impediria o módulo de obter o conteúdo apenas com o endereço da página específica.

Outra responsabilidade deste módulo é permitir que o usuário do sistema possa preservar sua privacidade. Como foi observado no capítulo 3, é essencial que o usuário de um sistema de mineração da Web sinta-se seguro para utilizá-lo. Dessa forma, é muito importante que este módulo permita que não sejam armazenadas informações pessoais e sigilosas, como, por exemplo, seu extrato bancário enquanto acessa seu *Internet Banking*.

Este módulo pode ainda executar outras tarefas que dependem de interação com o usuário, como, por exemplo, permitir que ele classifique uma página espontaneamente. Isso acontecerá quando um determinado usuário quiser contribuir com a base de dados do sistema explicitando o assunto sobre o qual determinada página se refere. Este tipo de informação será muito útil para comparação de outras páginas com as páginas já classificadas pelos usuários. Isso significa que o módulo Argus-MMN é responsável por fazer a interface que irá obter classificações através de folksonomia.

Como as informações obtidas pelo Argus-MMN serão compartilhadas com outros módulos, é interessante que sejam manipuladas apenas por um módulo específico.

4.1.2 Hera-MMD (Módulo Manipulador de Dados)

Enquanto o módulo Argus-MMN monitora a navegação do usuário, todas as informações que obtém são enviadas para o módulo Hera-MMD. A função deste módulo é, além de armazenar as informações brutas colhidas pelo Argus-MMN, armazenar as informações que serão lapidadas posteriormente.

Este módulo também pode ser desenvolvido localmente caso o interesse esteja apenas em armazenar a navegação da cada usuário separadamente. No entanto, o desenvolvimento deste módulo em um servidor permite que as informações de todos os usuários sejam centralizadas e que possam ser analisadas de maneira global. Isso permitiria comparações das cadeias de conhecimento dos diferentes usuários, criação de uma cadeia de conhecimento comum e/ou ótima, identificar quais usuários

possuem cadeias com maior ou menor números de nós (o que poderia representar o quanto cada usuário navegou em busca daquele conhecimento).

4.1.3 Hermes-MAD (Módulo Analisador de Dados)

Hermes foi o deus enviado por Zeus para recuperar Io. Nessa arquitetura o módulo Hermes-MAD tem o mesmo papel. Ele é responsável por resgatar as informações brutas que são mantidas pelo módulo Hera-MMD, minerá-las e transformá-las na cadeia de conhecimento chamada de IO. Este módulo é, portanto, o que acumula maior inteligência no processo.

Uma vez que este módulo obtenha as informações de determinado usuário, ele aplica as técnicas de mineração de texto descritas no capítulo 3 para obter um extrato do texto e classificá-lo de acordo com os conceitos da ontologia de domínio (GUARINO, 1998). Se uma página for classificada em um conceito X e a próxima página navegada for classificada como sendo sobre o conceito Y, estes conceitos serão adjacentes na cadeia de conhecimento:

$$A \rightarrow B \rightarrow ... \rightarrow X \rightarrow Y \rightarrow ...$$

Figura 4.2 Exemplo de cadeia de conhecimento navegada

A arquitetura interna do módulo Hermes-MAD é fortemente baseada na arquitetura proposta por Magalhães (2002), com as modificações necessárias para a adaptação dos módulos ao problema aqui proposto. A Figura 4.3 ilustra a arquitetura interna do módulo, com todos os seis subsistemas. Ela é resultante da expansão do módulo Hermes-MAD da Figura 4.1.

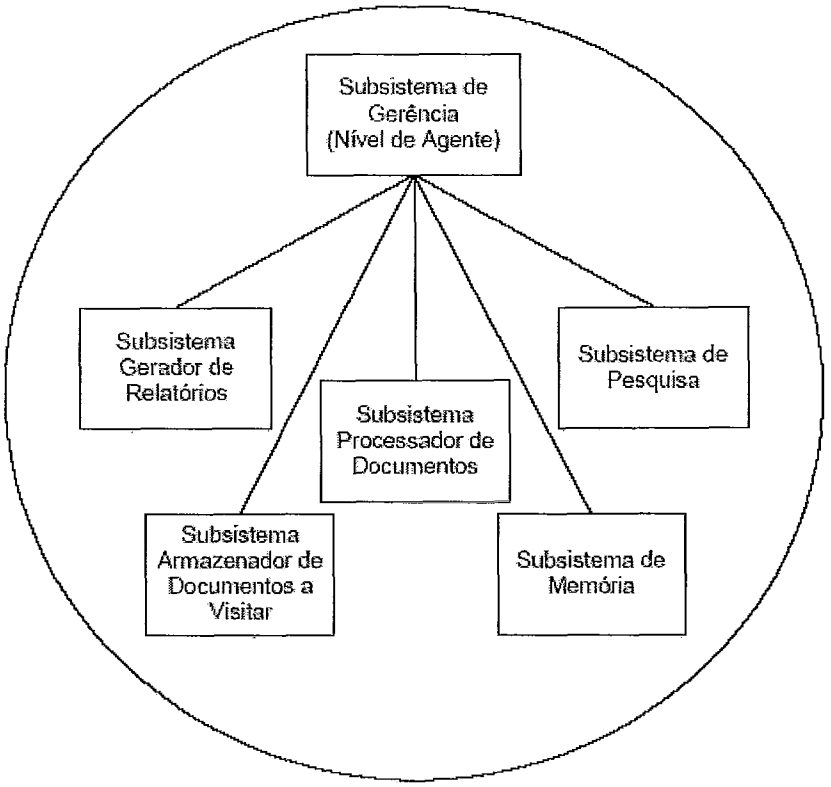


Figura 4.3 - Arquitetura interna de um módulo cliente

Cada um dos subsistemas é responsável por um aspecto específico:

- Subsistema Processador de Documentos (SPD): É responsável por realizar o processamento dos documentos a serem classificados, extraindo as informações relevantes como seu texto puro e analisá-los.
- Subsistema Armazenador de Documentos a Visitar (SADV): É responsável por analisar e armazenar caminhos para documentos que ainda serão visitados durante a classificação. No início do processo de classificação, este subsistema deve receber os documentos a serem classificados *a priori*. Este subsistema pode ser alimentado também pelo próprio módulo de software, caso os documentos classificados sejam capazes de indicar novos documentos – neste caso, para evitar que a busca torne-se excessivamente grande, há o conceito de altura do documento, que funciona como um corte na árvore de busca: todos os documentos inseridos inicialmente no subsistema recebem uma altura h. Quando um documento é processado e gera novos documentos a serem processados, tais documentos recebem a altura (h – 1). O

processo se repete até que h seja zero. Quando um documento de altura zero é processado, não se verifica se ele indica novos documentos. O motivo de deixar a responsabilidade de armazenamento separada em um subsistema é possibilitar que seja feita uma crítica inicial em relação aos documentos que ainda serão processados – por exemplo, é possível inserir prioridades a documentos, ou ainda inserir filtros para evitar que uma classe de documentos seja processada.

- Subsistema de Memória (SM): É o subsistema responsável por guardar, de forma permanente, a lista dos documentos já classificados (e não alterados desde a última classificação), evitando que sejam re-processados inutilmente.
- Subsistema de Pesquisa (SP): É o subsistema que encapsula o algoritmo utilizado para classificar os documentos.
- Subsistema Gerador de Relatórios (SGR): Este subsistema é responsável por definir a forma como os resultados obtidos a partir da classificação são utilizados.
- Subsistema de Gerência (SG): É o "cérebro" do módulo, sendo responsável por coordenar os demais subsistemas. Este é a parte ativa do módulo, representando um *thread* independente de execução.

Com uma freqüência determinada pelo usuário, o módulo Hermes-MAD selecionará as páginas que são relacionadas ao assunto discutido pela comunidade (o assunto deve ser conhecido porque é necessário possuir uma ontologia que o descreva). Isso será feito através da comparação do conteúdo da página com o conjunto de palavras-chave (associadas aos conceitos da ontologia) relacionadas ao assunto em questão. Isso significa que as páginas armazenadas são filtradas e apenas aquelas que são, de fato, do interesse da comunidade permanecem para a próxima etapa da análise. Isso também resolve em parte algum problema relacionado com a privacidade do usuário, uma vez que aquelas páginas que não são relevantes para a comunidade são descartadas.

Com este conjunto de páginas armazenadas, o sistema possui um grafo direcionado, já que a ordem de navegação foi armazenada. Como a motivação do sistema é criar um IO com os conceitos estudados pelo aprendiz, é necessário usar técnicas de mineração de texto para classificar as páginas de acordo com os conceitos

descritos na ontologia. Essa classificação é baseada na proposta de Desmontils & Jacquin (2001) e Joachims *et al.* (1997). No entanto, o uso de tesauro foi substituído por uma ontologia.

Neste momento, o sistema remove todas as *stop words* do texto da página. A seguir é necessário dar peso para cada palavra remanescente no conteúdo. Até aqui os marcadores HTML são mantidas para auxiliar na definição do peso de cada palavra, afinal, as palavras contidas no marcador "title" certamente têm peso maior que as contidas no marcador "b" (*bold*), que têm peso maior que as palavras que não estão contidas em marcadores.

Tabela 4.1 - Exemplos de coeficientes associados aos marcadores HTML (JOACHIMS *et al.*, 1997; DESMONTILS, JACQUIN, 2001).

Descrição do marcador HTML	Marcador HTML	Peso
Título do documento	*<title> </title>*	10
Keyword	*<meta name="keywords"... content=...>*	9
Hyper-link	* *	8
Fonte com tamanho 7	* *	7
Fonte com tamanho +6	* *	6
Fonte com tamanho 5	* *	5
...	...	...
Cabeçalho nível 1	* *	3
Título de imagem	*<img ... alt="..."> *	2
Fonte sublinhada	*<u> </u>*	2
Fonte em itálico	* *	2
Fonte em negrito	* *	2
...	...	...

Os pesos são dados de acordo com os valores dados na Tabela 4.1, que representa os pesos propóstos por Desmontils & Jacquin (2001) e Joachims *et al.* (1997)

Uma vez que a freqüência e o peso das palavras-chave de uma página são comparados com os conceitos de ontologia, a página recebe graus de relevância. Com este relacionamento entre páginas e conceitos da ontologia, o gráfico de páginas pode

ser transformado em uma cadeia de conhecimento. Essa cadeia pode ser recomendada ao aprendiz e ele pode decidir o que fazer com ela.

Como há vários módulos "trabalhando" para os aprendizes da comunidade, várias cadeias serão criadas. Com isso, é possível identificar conceitos ausentes na navegação de um aprendiz que já foi estudado por outro, e recomendar a cadeia de conhecimento do usuário mais avançado. Junto com essa cadeia podem ser recomendados os conceitos, as páginas e até quem estudou aquele conjunto de conceitos a fundo.

4.2 Sistema de recomendação de IO

Como foi mencionado na seção 3.2, a principal dificuldade na mineração de conteúdo é a falta de estrutura dos documentos armazenados na Internet. Essa falta de estrutura é responsável pela imprecisão das técnicas de mineração em relação à classificação de um conteúdo em uma lista de conceitos. Isso pode levar o módulo Hermes a classificar um subconjunto de páginas com conteúdo bem diversificado de forma imprecisa. Essa imprecisão poderia gerar um nó equivocado na árvore de conhecimento.

Por isso, é importante observar que o IO, produto gerado por este sistema, será recomendado para o mesmo aprendiz que o gerou. Dessa forma, ele pode alterá-lo antes de compartilhá-lo com seus colegas de comunidade sempre que identificar nós equivocados ou quando ele próprio chegar à conclusão que o seu modelo de estudo não foi adequado.

Para editar sua cadeia de conhecimento, o aprendiz pode usar a ferramenta KCE, definida por Silva (2006). Além de permitir a edição da cadeia de conhecimento, essa ferramenta permite a recomendação dessa cadeia para outros usuários. Neste caso, outro aprendiz precisa criar uma unidade de conhecimento (um nó conceito da cadeia) com o estado "dúvida". Neste momento, o sistema começa a busca. Ele envia mensagens para outros pontos e espera por suas respostas. Cada ponto executa uma busca interna, que consiste em verificar a existência de alguma unidade de conhecimento similar àquela desejada. Todas as unidades de conhecimento encontradas são enviadas ao ponto requisitante. Este processo é representado pela Figura 4.4.

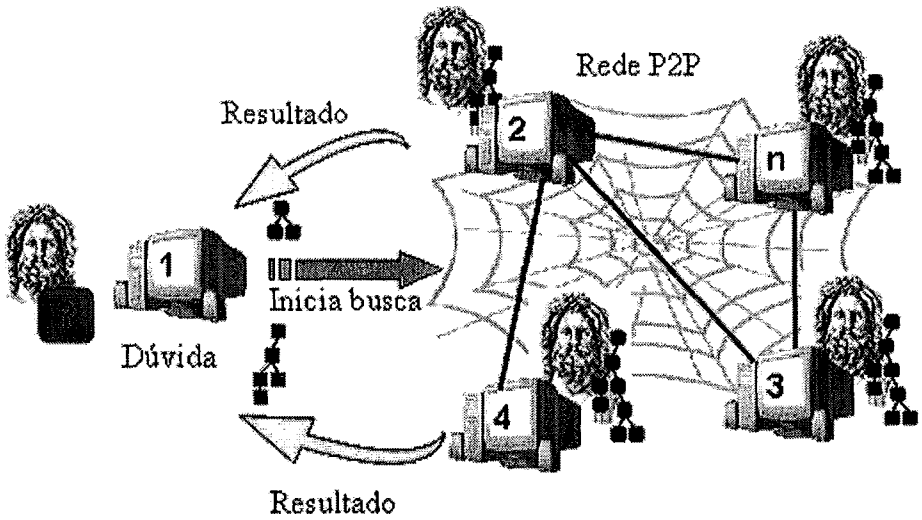


Figura 4.4 - Arquitetura do KCE (PEREIRA *et al.*, 2006; SILVA *et al.*, 2006)

4.3 Implementação do Sistema Olimpo

A arquitetura proposta anteriormente é flexível em relação a onde cada módulo pode ser instalado e quais tecnologias podem ser usadas para o seu desenvolvimento. No entanto, para o estudo de observação realizado neste trabalho, foi necessário tomar algumas decisões em relação a onde estes módulos seriam instalados e quais tecnologias seriam utilizadas.

Parte dos módulos foi desenvolvida para ser instalada na máquina do usuário e a outra parte foi desenvolvida para ser instalada em um servidor. Além disso, os módulos foram desenvolvidos em tecnologias diferentes por se mostrarem mais adequadas às suas tarefas específicas. A seguir, será explicado como cada módulo foi desenvolvido.

4.3.1 Desenvolvimento do Argus

Para este estudo de observação o módulo Argus foi desenvolvido para rodar na máquina do aprendiz que escolher utilizar o sistema. Ele é instalado como um *plug-in* no navegador Mozilla Firefox versão 2.0.0.3 (MOZILLA, 2007). Esse navegador foi escolhido por questões de:

- Portabilidade, já que possui versão para Microsoft Windows (MICROSOFT, 2007b), Mac OS X (APPLE, 2007) e para diversas distribuições de Linux (LINUX, 2007) e o *plug-in* pode ser instalado em qualquer plataforma da mesma forma.

- Facilidade de distribuição do *plug-in*, já que é o segundo navegador mais utilizado e com ascensão no número de usuários sobre o primeiro colocado (HU, 2005; GONSALVES, 2006; BEER, 2007).

Para o desenvolvimento de *plug-in* no Firefox, é necessário utilizar a linguagem XUL (XML *User Interface Language*) (MOZILLA, 2007b) que, como o próprio nome sugere, utiliza XML (W3C, 2006) para descrever os componentes de interface e a linguagem de script JavaScript (MOZILLA, 2007c) para programar as ações do *plug-in*.

Essa integração direta com o navegador facilitou o acesso às informações navegadas, como URL, conteúdo, data e hora da navegação. Além de permitir uma fácil interação como usuário para ligar e desligar o Argus, e cadastrar uma lista de domínios que não gostaria que fossem armazenados.

Dessa forma, dois problemas são resolvidos através da integração entre o Argus e o Firefox. O primeiro é que não é necessário minerar um arquivo de *log* de *proxy* para as URLs navegadas, além de não ser necessário visitar novamente essas páginas para obter seu conteúdo (fato que poderia gerar grandes problemas, como vimos anteriormente). O segundo é dar maior privacidade aos usuários uma vez que podem facilmente ligar e desligar o *plug-in* ou restringir seu universo de atuação, como mostra a Figura 4.5.

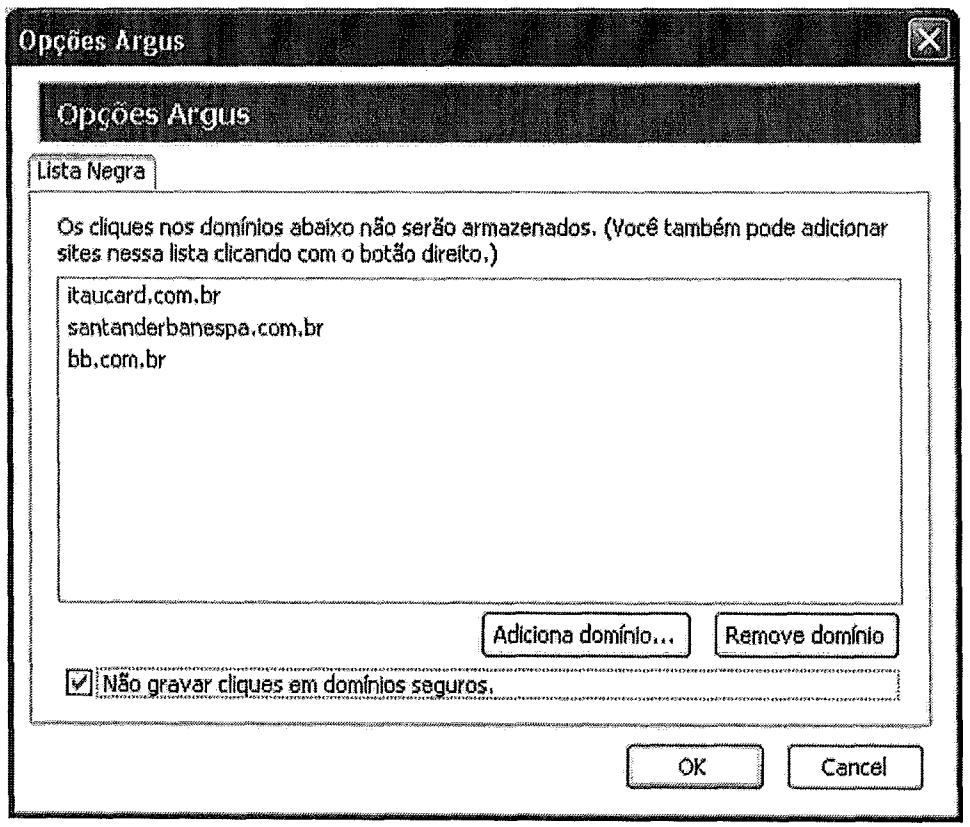


Figura 4.5 - Lista de domínios que não serão observados pelo Argus

É através de uma caixa de seleção disponibilizada no navegador pelo Argus (Figura 4.6) que o usuário pode contribuir com a base de dados do sistema. Para isso basta o usuário selecionar na caixa de seleção qual o conceito da ontologia que classifica a página corrente e clicar no botão "Classificar". Neste momento o conteúdo da página e o conceito da ontologia são enviados para o módulo Hera.

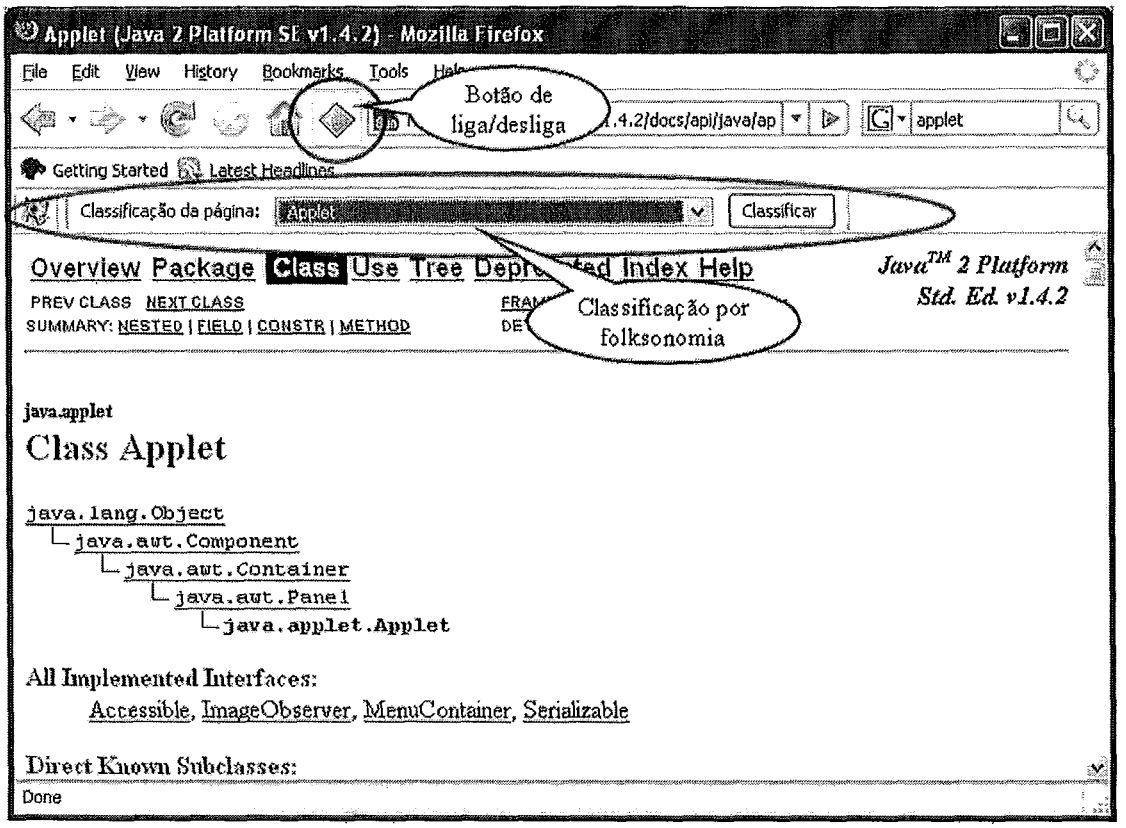


Figura 4.6 - Interface do plug-in Argus

Uma vez que o Argus estiver ligado, o módulo solicita ser avisado pelo Firefox dos eventos em relação à página navegada. Toda vez que o navegador detectar um clique em um *link*, ele avisa ao Argus que o conteúdo da página mudou. Neste momento, o Argus solicita que o Firefox o informe os dados que precisa a respeito daquela página, os encapsula em um objeto e os envia ao módulo Hera.

4.3.2 Desenvolvimento de Hera

O módulo Hera, por sua vez, fica localizado em um servidor da Internet e é disponibilizado como um serviço Web (W3C, 2007). Este tipo de serviço facilita a integração entre os diversos usuários do Argus com o módulo centralizador Hera, uma vez que cada instância de Argus monta seu próprio pacote XML com as informações

que deseja enviar e simplesmente o faz através da conexão HTTP do usuário. Esse tipo de comunicação evita problemas de segurança na máquina do usuário, já que não exige que seja feita nenhuma configuração de *firewall*, assim como não exige que nenhuma porta de comunicação específica seja aberta.

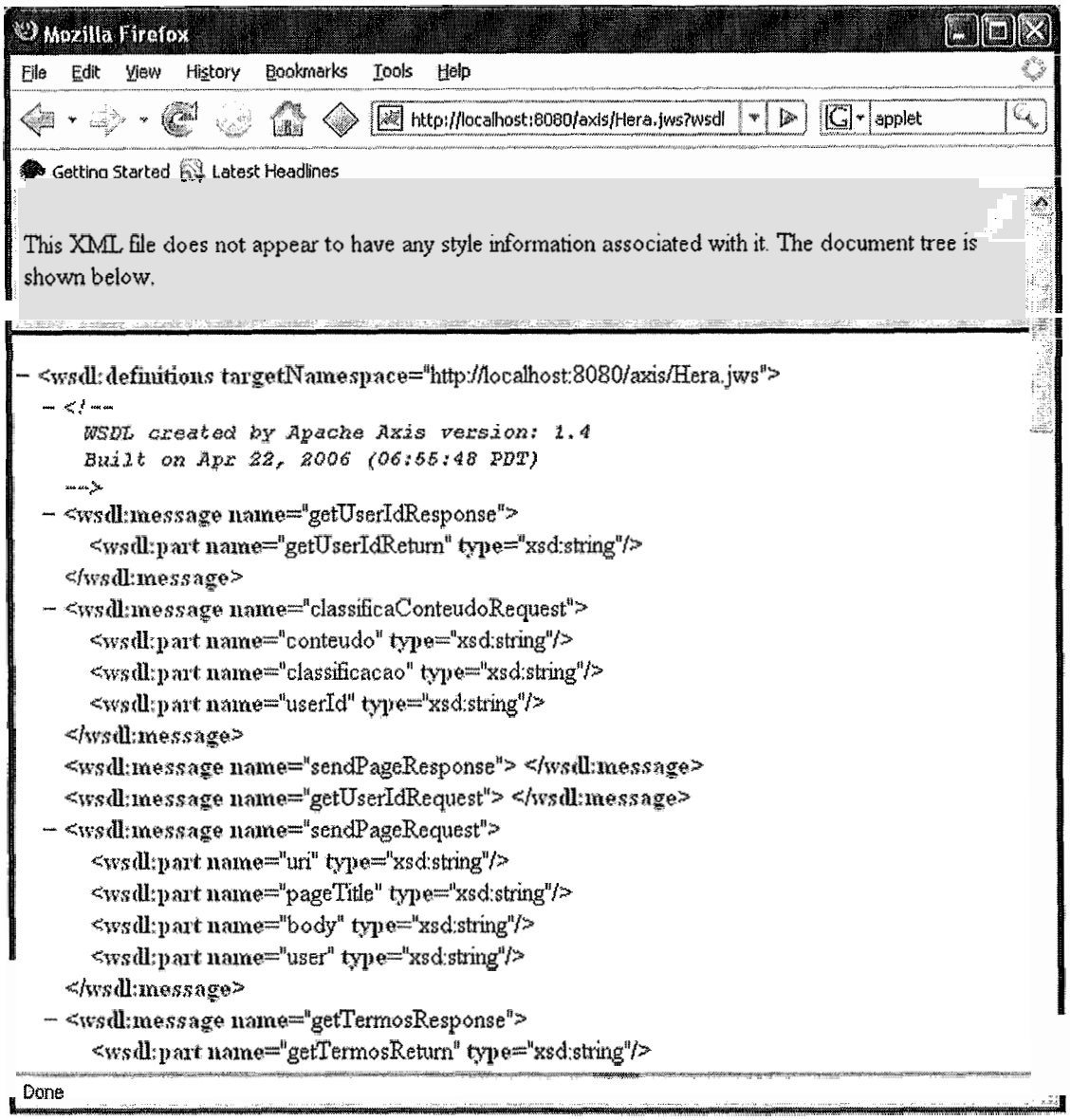


Figura 4.7 - Exemplo do arquivo WSDL (W3C, 2007c) do serviço da Internet Hera

Este serviço foi desenvolvido em Java com Apache Axis (APACHE, 2006), uma implementação de envio de mensagens SOAP (*Simple Object Access Protocol*) (W3C, 2007b) de acordo com a especificação da W3C. A Figura 4.7 representa parte do arquivo WSDL (W3C, 2007c) do serviço. Este arquivo é considerado o descritor padrão pra serviços Web.

Para cada pacote de informações que chega ao serviço Hera, é criada uma nova *thread* para salvar a informação no banco de dados, evitando assim perda de

informações por indisponibilidade do serviço. Caso este recurso não fosse utilizado o serviço Web poderia estar atendendo uma requisição de um módulo remoto enquanto um outro módulo enviasse uma informação para ser armazenada. Neste caso o serviço estaria indisponível e o novo módulo não conseguiria salvar seus dados.

Em relação à classificação de páginas através de folksonomia Hera possui dois serviços fundamentais. O primeiro é manter as instâncias de Argus com sua caixa de seleção de conceitos atualizada. Assim sempre que houver uma evolução na ontologia os *plug-ins* serão informados. O segundo serviço recebe e o armazena o conteúdo classificado pelos usuários no banco de dados.

Hera ainda cria novos usuários para novas instalações do Argus e o informa do identificador do novo usuário criado.

A outra interface de Hera é com o módulo Hermes, que recebe todas as páginas navegadas por um usuário sempre que solicita.

4.3.3 Desenvolvimento do Hermes

Neste trabalho o módulo Hermes foi desenvolvido para rodar como um agente JADE (Java *Agent DEvelopment Framework*) (JADE, 2007). Este módulo foi dividido em duas partes. Uma fica instalada no servidor central e interage diretamente com o serviço Web Hera. Essa parte do módulo é responsável por todo o processamento da classificação das páginas e geração do IO. A outra parte fica instalada nas máquinas dos usuários e é responsável por fazer apenas a interação com os mesmos.

A implantação deste módulo em um servidor central criaria a necessidade de uma máquina com configurações avançadas de processamento ou seria necessário uma excelente otimização do módulo para evitar consumo excessivo de processamento e memória, caso fosse utilizado para uma comunidade real com milhares de membros.

No módulo local, o usuário pode configurar a freqüência que deseja que o módulo Hermes analise os dados de sua navegação ou então pode simplesmente solicitar que a análise se inicie naquele momento. Em seguida, o módulo Hermes envia uma solicitação ao servidor para que a análise seja iniciada informando qual o usuário que a está solicitando. Uma vez com o pedido, o servidor irá obter as informações de navegação do determinado usuário. Com uma lista do conteúdo das

páginas visitadas ordenados pela ordem de visita, o Hermes inicia a análise de forma seqüencial.

Com o conteúdo de uma determinada página, o módulo utiliza as técnicas de mineração de texto descritas anteriormente. Primeiro aplica o *case folding*, colocando todo o texto em minúsculo. Em seguida, o módulo aplica peso a cada palavra de acordo com o marcador HTML que a contém, conforme foi apresentado na 0. Só então, todos os marcadores HTML ou de qualquer outra linguagem de script são removidos.

Uma vez que o Hermes possui um texto limpo, sem marcadores, ele o envia para o banco de dados MySQL (MYSQL, 2007), que aplica a lematização, remove as *stop words* e atribui os pesos das palavras para o modelo vetorial (SALTON, BUCKLEY, 1988). Agora que o conteúdo da página foi representado como um vetor o módulo utiliza a ontologia para classificá-lo.

Com uma ontologia criada para a aplicação, o módulo Hermes extrai as palavras-chave de cada conceito e armazena no banco de dados como se fossem documentos armazenados no modelo vetorial. Assim basta utilizar o mecanismo de indexação *full-text search* do MySQL para comparar o conteúdo com cada termo da ontologia e finalmente obter sua classificação mais provável.

Caso os usuários da comunidade tenham classificado páginas através de folksonomia, essas também estarão armazenadas no banco de dados, indexadas como vetor e relacionadas com conceitos da ontologia. Nestes casos, as páginas que estão sendo avaliadas não só são comparadas com as palavras-chave da ontologia como também são comparadas com as páginas classificadas pelos usuários.

À medida que este processo é repetido para cada página, a seqüência de conteúdo de páginas visitadas transforma-se em uma seqüência de conceitos visitados. Com essa seqüência de conceitos, o módulo monta um grafo orientado de conceitos estudados pelo usuário.

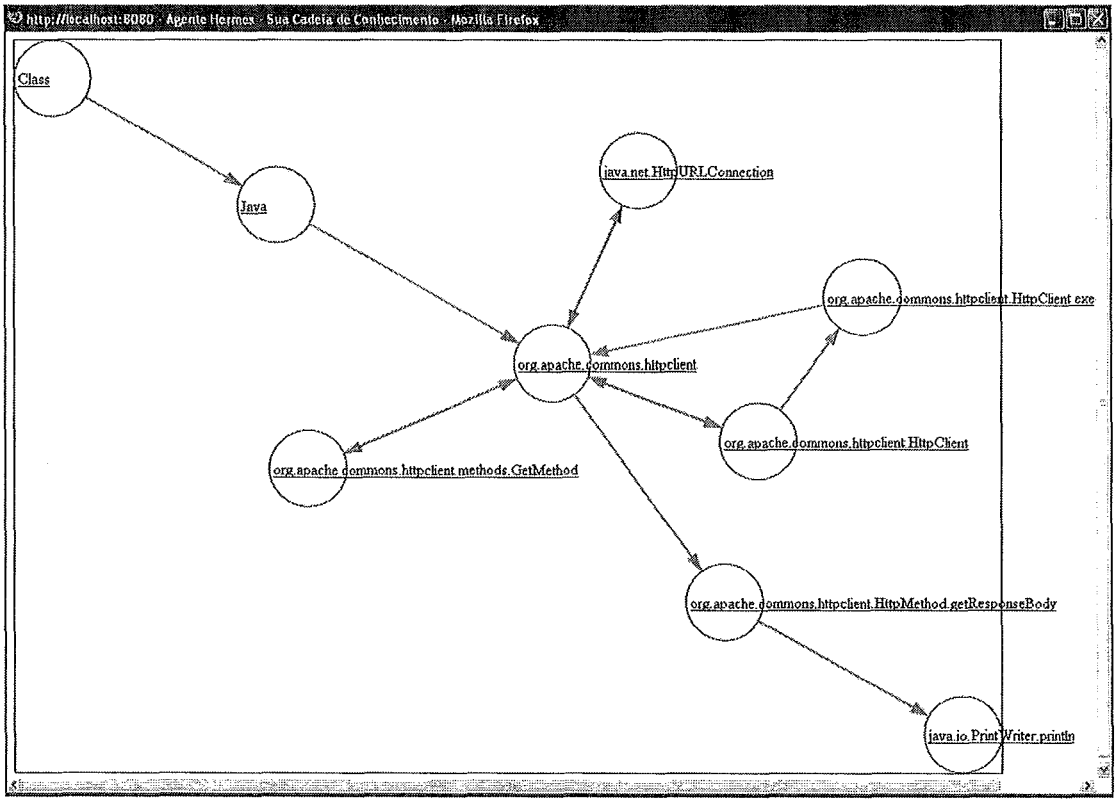


Figura 4.8 - IO gerada pelo módulo Hermes

Este grafo é disponibilizado ao usuário para que possa validá-lo ou alterá-lo, depois é disponibilizado no KCE para que possa ser compartilhado entre os outros membros da comunidade. Nele o usuário pode identificar o conceito estudado em cada nó da cadeia e acompanhar a seta para descobrir qual(is) o(s) próximo(s) conceitos a serem estudado(s).

4.4 Exemplo

O exemplo a seguir mostra como um IO é criado a partir da navegação do aprendiz através das páginas da Internet. A Figura 4.9 mostra as páginas navegadas pelo usuário e a Figura 4.10 representa a ontologia utilizada pela comunidade. É interessante ressaltar que as setas da Figura 4.10 não representam uma relação hierárquica entre os conceitos da ontologia.

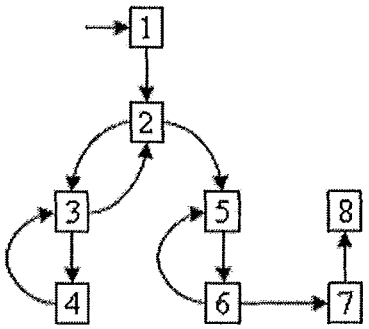


Figura 4.9 - Navegação das páginas da Internet

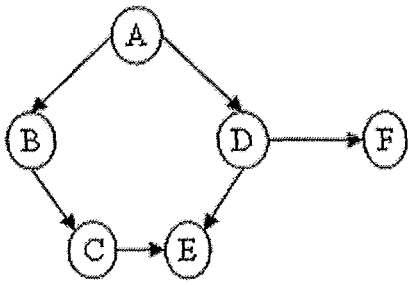


Figura 4.10 - Ontologia da comunidade

Em um primeiro momento, será executada a mineração de Internet e, de acordo com as palavras-chave encontradas na página Web, ela pode ser classificada parcialmente por um conceito da ontologia e parcialmente por outro. Neste caso, há um grau de relevância para cada conceito relacionado à página. Isso significa que, para cada página, o resultado é:

Tabela 4.2 – Exemplo de classificação de páginas

Página 1: A – 60%; B – 10%; C – 30%; ...
Página 2: A – 0%; B – 0%; C – 100%
...

Depois de relacionar as páginas da Internet com os conceitos mais relevantes da ontologia, o módulo Hermes criará um caminho de aprendizado na ontologia que se assemelha a uma ontologia de aprendizado: A → C → D → E → ...

A criação do IO será iniciada mapeando as páginas a esse tipo de ontologia de aprendizado.

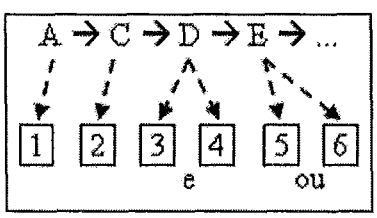


Figura 4.11 - Exemplo relacionamento entre páginas e conceitos

Dessa forma, o IO é criado usando a ontologia de aprendizado e todas as informações da navegação do aprendiz pelas páginas Web. Se este exemplo fosse baseado em uma ontologia sobre linguagem de programação Java, poder-se-ia concluir que é necessário estudar "Atributos", depois "Classe", para estudar "Objeto".

Um IO é uma estrutura formada por um conjunto de atributos que são agrupados em categorias: Geral (nome, descrição, palavras-chave, autor, data de criação, data da última utilização), Ciclo de vida (histórico, estado corrente, contribuições), Direitos (direitos de propriedade intelectual, condições de uso), Relação (o relacionamento entre conhecimentos-base), Classificação (o IO em relação ao sistema de classificação) e Anotação (comentários e avaliações dos IOs e dos seus criadores). Muitos destes atributos podem ser automaticamente preenchidos, o que facilita a criação de uma nova cadeia de conhecimento.

4.5 Conclusão

A arquitetura permite a classificação das páginas e a criação de uma cadeia de conhecimento representada por um grafo orientado de estruturas que carregam não só o conceito estudado pelo usuário, mas também as páginas que ele visitou para obtê-lo.

Se o aprendiz utilizar o Olimpo durante todo seu processo de aprendizado em *sites* de Internet, este sistema terá informações suficientes para compor uma IO aproximada dos conceitos navegados por ele durante este processo. Com essa cadeia de conceitos recomendada, o aprendiz pode utilizar o KCE para alterá-la de forma que:

- Represente melhor a realidade de como os conceitos foram realmente navegados, caso a cadeia de conceitos tenha sido montada pelo sistema de forma imprecisa;
- Represente melhor a ordem ideal de como os conceitos devem ser navegados por outro aprendiz que possa utilizar essa cadeia como referência, caso o aprendiz chegue a conclusão que a forma como adquiriu o conhecimento não tenha sido a melhor.

No entanto, a ontologia e a folksonomia têm papel muito relevante no processo de classificação empregado pelo módulo Hermes. Por isso é importante que fique claro como a ontologia foi criada para atender essa aplicação e como pode ser instanciada para diferentes assuntos. Assim como é relevante que fique claro como a

folksonomia auxilia este processo. Falaremos sobre esses assuntos no capítulo seguinte.

Capítulo 5 - Ontologia de Linguagem de Programação

Uma ontologia sobre o domínio abordado pela comunidade de usuários que utilizam essa aplicação é esperada para o seu correto funcionamento. Essa ontologia apóia a classificação de conteúdo através da mineração. Uma base de conhecimento também pode ser criada e alimentada com a colaboração dos usuários para ajudar na tarefa de classificação. Essa contribuição é conhecida como folksonomia.

Neste capítulo vamos dissertar sobre a construção da ontologia que a apóia o processo de navegação e como ela pode ser alterada para outros fins. Além disso, vamos apresentar como a folksonomia pode nos apoiar neste processo.

5.1 Ontologia

Existem várias (e, em alguns casos, divergentes) definições de ontologia apresentadas na literatura. Noy & Mcguinness (2001) apresentam ontologia como uma descrição formal e explícita de um domínio específico. Guarino (1998), no entanto, defende que as ontologias podem ser categorizadas em quatro tipos:

- Ontologias de alto-nível descrevem conceitos mais gerais como espaço, tempo, objeto, etc, que são independentes de um problema particular ou de um domínio específico.
- Ontologias de domínio descrevem o vocabulário relacionado a um domínio genérico como, por exemplo, medicina, automóveis, enologia, etc. Esse tipo de ontologia é obtido através da introdução de termos específicos em uma ontologia de alto-nível.
- Ontologias de tarefa descrevem tarefas genéricas ou atividades como diagnósticos ou vendas. Esse tipo de ontologia também é obtido através da especialização de uma ontologia de alto-nível.
- Ontologias de aplicação descrevem conceitos que estão diretamente relacionados tanto a um domínio particular quanto a uma tarefa específica. Freqüentemente esse tipo de ontologia é originado da especialização de ontologias de domínio e de tarefa.

Uma ontologia mais genérica pode tornar-se mais específica de acordo com a necessidade sem que seja empregado grande esforço no processo de especialização. Entretanto, para transformar uma ontologia originalmente específica em uma mais genérica pode ser uma tarefa realmente difícil.

Apesar de divergirem em sua definição, os pesquisadores, em geral, concordam que as ontologias são compostas por classes (ou conceitos), propriedades de cada conceito que descrevem suas várias características e atributos são chamadas de *slots* (ou roles, ou ainda simplesmente propriedades), e restrições dos *slots* chamados de *facets* (ou role *restriction*).

Noy & Mcguinness (2001) defendem que um conjunto de instâncias das classes de uma ontologia compõe uma base de conhecimento. No entanto, eles reconhecem que é difícil distinguir a fronteira onde termina uma ontologia e começa uma base de conhecimento.

Diversos padrões e linguagens para construção e compartilhamento de ontologias na Web estão sendo criados, todos baseados no XML, com algumas diferenças de sintaxe de marcação (*tags*). Alguns exemplos são o SHOE (HEFLIN, 2001), a *Ontology Exchange Language* (XOL) (KARP *et al.*, 1999), a *Ontology Markup Language* (OML e CKML)(KENT, 1999) e a *Resource Description Framework Schema Language* (RDFS) (W3C, 2004). Existe uma proposta de extensão do RDF e do RDFS chamada OIL (*Ontology Interchange Language*) (FENSEL, 2000) e seu sucessor DAML + OIL (HORROCKS *et al.*, 2002).

O DAML + OIL (DARPA *Agent Markup Language – Ontology Interchange Language*) é uma linguagem baseada no XML, desenhada para possuir muito mais capacidade que este na descrição de objetos e no seu relacionamento; para expressar semântica e criar um alto grau de interoperabilidade entre *sites* da Internet. O OWL é uma linguagem de marcação semântica para publicação e compartilhamento de ontologias na Internet e do DAML+OIL. Um exemplo de um editor que suporta a criação cooperativa de ontologias baseado na Internet é o Webonto (DOMINGUE, 1998).

Ontologia é uma tecnologia essencial para a *Web* Semântica (BERNERS-LEE *et al.*, 2001), porque fornece o vocabulário necessário que permite definir a estrutura e a semântica de documentos de modo que possam ser compreendidos pelas máquinas, possibilitando pesquisas inteligentes na Web. Permite também o compartilhamento do conhecimento de um domínio que pode ser compartilhado entre pessoas e aplicações.

5.1.1 Ontologias de Aprendizado

Ontologia de aprendizado colaborativo (SUPNITHI, 1999) é um sistema de conceitos para modelagem o processo de aprendizado colaborativo, como objetivo de aprendizado, tipo de grupo de aprendizado, e cenário de aprendizado. Quando as ontologias de aprendizados são empregadas na formação de uma comunidade de colaboração oportunística, elas normalmente são organizadas em camadas:

- A camada mais baixa é a de nível de agente corresponde à criação e manutenção da ontologia de aprendizado individual. Nela as aplicações coletam ou permitem que o usuário crie sua ontologia de aprendizado pessoal.
- A camada intermediária corresponde à ontologia de aprendizado colaborativo. Essa camada abstrai as particularidades remanescentes da camada de agente criando a abstração necessária para que essa ontologia seja levada para a camada superior e negociada entre os membros da comunidade de aprendizado. Essa camada, no entanto permite eventuais comunicações diretas entre as camadas para que os membros da comunidade possam alterar diretamente a ontologia de aprendizado para criação de novas propostas durante o processo de negociação.
- O nível mais alto é o de negociação e é onde é feita a negociação da ontologia. Essa camada comunica-se com a intermediária em várias iterações durante o processo de negociação da ontologia de aprendizado colaborativo definitiva da comunidade. E é nela que são resolvidos os conflitos entre as várias propostas feitas pelo grupo.

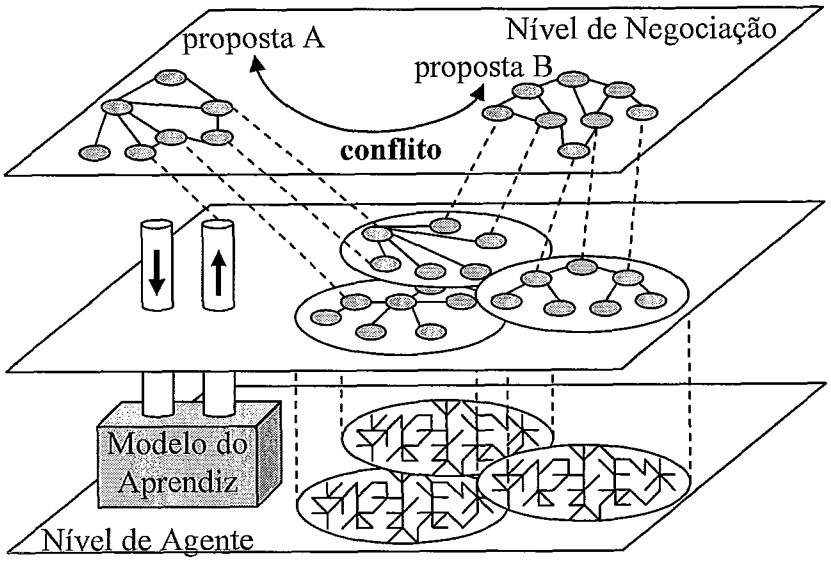


Figura 5.1 - Arquitetura de ontologia em uma formação de grupo oportunística (SUPNITHI, 1999)

5.2 Estudo de Caso da Ontologia

Como o estudo de caso do Olimpo foi desenvolvido para dar suporte a uma comunidade de aprendizado da linguagem de programação Java, uma ontologia de linguagem de programação orientada a objeto foi desenvolvida e posteriormente instanciada para Java.

A primeira ontologia criada foi uma ontologia de domínio que descrevia conceitos de orientação a objeto (OO). Em seguida, propriedades específicas foram acrescentadas na ontologia criada para incorporar funcionalidades mais próximas a um tesauro. A partir da ontologia inicial sobre linguagem de programação OO, um conceito chamado "*Concept*" foi criado uma propriedade chamada "*keyword*". Essa propriedade permite que sejam adicionados sinônimos ou palavras-chave que representem o conceito. Todos os demais conceitos da ontologia passaram a herdar do conceito "*Concept*" sua propriedade "*keyword*". Dessa forma, todos os conceitos da ontologia possuem a propriedade "*keywork*" onde o módulo Hermes pode buscar por palavras encontradas no texto das páginas e correlacioná-los com os conceitos da ontologia.

Com isso, a ontologia que até então era classificada como de domínio foi especializada para tornar-se uma ontologia de aplicação. No entanto, a simples remoção do conceito "*Concept*" faz com que ela volte a ser uma ontologia de domínio.

A ontologia de linguagem de programação OO foi instanciada para a linguagem Java para que fosse usada como base de conhecimento para atender a comunidade Java. Com os conceitos e relacionamentos instanciados, é possível comparar palavras-chave encontradas no processo de mineração das páginas com as palavras-chave da ontologia. A atribuição de peso para as palavras-chave torna possível a classificação probabilística da página de acordo com os conceitos da ontologia.

O relacionamento entre os conceitos da ontologia pode ser usado para auxiliar em decisões sobre o conceito representado pela página. Quando a página possui ocorrências de palavras-chave relacionadas a conceitos diferentes, sua classificação torna-se mais complexa, exigindo algum nível de decisão. Neste caso, o sistema verifica se estes conceitos possuem algum relacionamento, como por exemplo, ambos herdam de um conceito comum. Em seguida ele classifica essa página baseada neste relacionamento. No exemplo onde ambos possuem um conceito-pai, a página é classificada como representante do conceito pai.

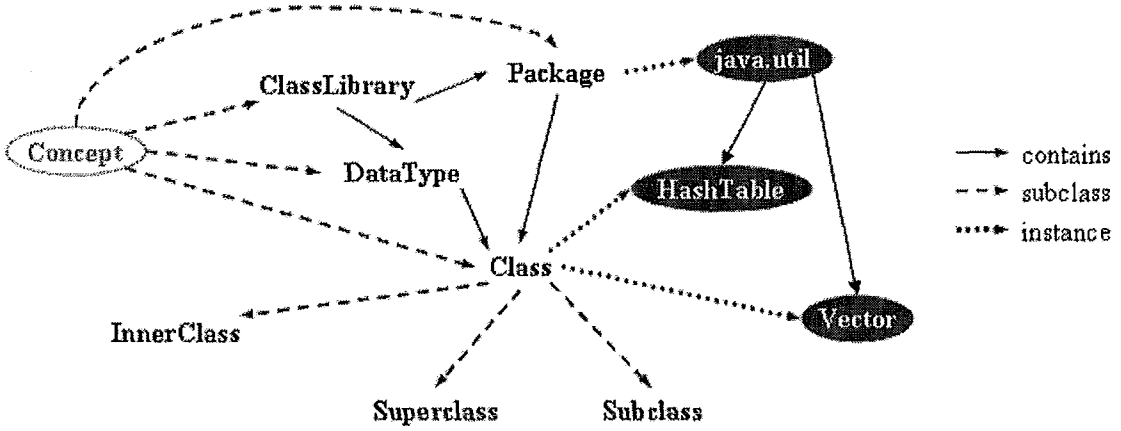


Figura 5.2 - Exemplo da ontologia criada

A Figura 5.2 representa uma pequena parte da ontologia onde podemos notar alguns conceitos relacionados pelo tipo de relacionamento "*contains*" que indica que um conceito contém outros, por exemplo, "*Package contains Class*" (Pacote contém Classe). O relacionamento *subclass* significa que um conceito é uma especialização de outro, por exemplo, "*InnerClass*" é uma especialização de "*Class*". E o relacionamento "*instance*" conecta um conceito a uma instância dele. Neste exemplo, "*java.util*" é um "*Package*" da linguagem Java, assim como "*HashTable*" e "*Vector*" são instâncias de "*Class*".

Ainda no exemplo dado pela figura, caso o módulo Hermes identifique que uma página possui muitas palavras-chave semelhantes ao conceito "*InnerClass*", mas também possui semelhanças significativas com "*Superclass*" e "*Subclass*" ele classificará a página com o conceito "*Class*" pois todas as alternativas possíveis estão relacionadas a este conceito, o que significa que a página pode estar listando conceitos sobre "*Class*".

5.3 Construção da Ontologia

A construção da ontologia mencionada nessa dissertação foi feita de forma semi-automática, uma vez que os conceitos foram criados de forma manual através da consulta de um mapa de conceitos da linguagem Java (SUN, 2007) e da inserção dos termos com a ferramenta Protégé (PROTÉGÉ, 2007).

As instâncias das diversas classes, pacotes, métodos e propriedades da linguagem Java foi feita através do desenvolvimento de uma ferramenta que lia o conteúdo da documentação de diversas APIs Java. Essa ferramenta recebia a URL inicial de uma API e percorria toda a estrutura do *site* inserindo na ontologia instâncias dos pacotes, classes, interfaces, métodos e campos encontrados nos documentos. Sendo assim, essa ferramenta constituiu a parte automática da construção da ontologia.

Durante a etapa manual da criação, utilizamos como base a metodologia para criação de ontologia proposta por Noy & Mcguinness (2001), que consiste de perguntas e atividades a serem realizadas durante a construção da ontologia. Varella (2007) descreve detalhadamente como cada passo da metodologia foi utilizado ou adaptado para a construção dessa ontologia de linguagem de programação OO.

A última versão da ontologia, usada no estudo de observação dessa dissertação possuía exatamente 46.593 (quarenta e seis mil quinhentos e noventa e três) termos e instâncias dos elementos dos pacotes básicos de Java, além dos encontrados nas APIs JavaMail (SUN, 2007b), e HttpClient (APACHE, 2007).

5.4 Conclusão

Para auxiliar o processo de classificação de conteúdo das páginas da Internet utilizamos uma ontologia de aplicação construída a partir de uma ontologia de domínio mais genérica e, por isso, facilmente extensível a outros contextos do mesmo domínio.

Como subproduto da utilização de ontologia no processo de classificação, este trabalho contempla as duas camadas mais baixas de uma ontologia de aprendizado colaborativo, uma vez que captura o processo de aprendizado pessoal do aprendiz, criando subsídio pra a camada mais baixa; e permite que os vários processos de aprendizado pessoal sejam trocados e fornece as fontes de consultas, criando as informações necessárias para a camada mais alta de negociação.

Capítulo 6 - Avaliação do Olimpo

No capítulo 4 foi apresentada a arquitetura geral proposta para o sistema Olimpo, explicando o papel de cada um de seus módulos e como eles interagem entre si. Para avaliar essa proposta nessa dissertação, um protótipo foi desenvolvido e aplicado em um estudo de observação.

Neste capítulo será descrito a organização e as conclusões obtidas no estudo de observação realizado para avaliar o Olimpo. O planejamento do estudo de observação foi definido baseado no modelo proposto por Travassos (2005).

A realização de um estudo experimental geralmente pode ser dividida em cinco fases: a definição, o planejamento, a execução, a análise e o empacotamento do estudo.

A definição do estudo consiste em resumir seus objetivos, seu foco de qualidade e os objetos que serão analisados. O planejamento envolve a descrição do perfil dos participantes, dos instrumentos, do processo de execução e uma avaliação crítica dos problemas que podem ser encontrados ao longo desta execução. A execução consiste na realização do estudo experimental pelos participantes, utilizando os instrumentos e o processo definidos no planejamento. A análise consiste na organização dos resultados gerados pelos participantes durante a execução e a realização de inferências sobre estes resultados. Finalmente, o empacotamento consiste na organização e armazenamento dos documentos construídos nas etapas anteriores, com o intuito de facilitar a repetição do estudo experimental no futuro (TRAVASSOS, 2005). A seguir estas etapas serão detalhadas:

6.1 Estudo de Observação

Um primeiro estudo de observação foi realizado em ambiente mais controlado com computadores idênticos e participantes com perfis parecidos. Esse estudo foi utilizado para extrair informações preliminares do sistema. E foi executado com supervisão do experimentador.

6.1.1 Definição do Estudo de Observação

O objeto de estudo desse trabalho é a utilização do sistema Olimpo e sua capacidade de gerar cadeias de conhecimento a partir da navegação dos participantes. Seu objetivo é, portanto, mostrar que é possível gerar essas cadeias através das navegações dos participantes e sugerir que essas cadeias possam ser de fato úteis para outros aprendizes.

O foco de qualidade é a representação dos conceitos presentes nas páginas navegadas corretamente representados em um grafo direcionado. Isto será medido através do cálculo da precisão da classificação das páginas e a cobertura do sistema em relação às páginas selecionadas como relevantes, paralelamente será mensurada a satisfação dos participantes com as cadeias geradas a partir da sua própria navegação. Adicionalmente gostaríamos de avaliar se houve qualquer tipo de ganho em relação ao uso de uma cadeia de conhecimento recomendada, no entanto, não nos propomos a medir o aprendizado dos participantes.

Utilizando uma notação baseada em GQM (VAN SOLINGEN, BERGHOUT, 1999), temos:

Analisar a utilização do sistema Olimpo; **com o propósito de** avaliar a viabilidade de sua utilização para geração automática de cadeias de conhecimento; **referente** à precisão da classificação das páginas Web e da satisfação dos usuários **do ponto de vista do** aprendiz, **no contexto de** aprendizado de utilização de bibliotecas de linguagem de programação Java.

6.1.2 Planejamento do Estudo de Observação

6.1.2.1 Contexto

Os participantes serão divididos pelo sistema em três grupos distintos. Cada grupo receberá uma tarefa inicial distinta. Cada tarefa descreverá um exercício básico na linguagem de programação Java, assunto escolhido para nortear este estudo devido ao elevado grau de interesse dos participantes. Ao receber suas tarefas, os participantes deverão executar o exercício fazendo as pesquisas necessárias na Web utilizando o navegador Mozilla Firefox versão 2.0.0.3 ou superior (MOZILLA, 2007), com o *plug-in* Argus instalado. O sistema registrará todas as atividades relacionadas à execução de cada tarefa e contabilizará o tempo necessário e o número de páginas visitadas para a sua conclusão.

6.1.2.2 Treinamento

O treinamento dos participantes que utilizarão o sistema Olimpo será realizado em um laboratório com 15 computadores idênticos, em sessão única com duração estimada de duas horas. O treinamento será composto de duas rodadas de tarefas.

Na primeira rodada de tarefas distribuídas, os participantes as executarão apenas com a ajuda de suas próprias pesquisas. Na segunda rodada, o grupo que receber inicialmente a Tarefa I, será responsável por executar a Tarefa II, o que receber inicialmente a Tarefa II executará a Tarefa III e o grupo que receber inicialmente a Tarefa III executará a Tarefa I. No entanto, dessa vez as tarefas serão executadas com o auxílio das cadeias de conhecimento geradas pelo grupo que executar a tarefa na rodada anterior. Na terceira rodada de tarefas, cada grupo será responsável por executar a tarefa que não tiver executado nas rodadas anteriores, mas dessa vez o fará com a ajuda de uma cadeia de conhecimento gerada por um especialista na linguagem.

6.1.2.3 Projeto Piloto

Antes da execução do estudo realizaremos um projeto piloto com uma estrutura idêntica a apresentada. Nesse piloto apenas três participantes simularão o comportamento de cada grupo. Este piloto não visa extrair qualquer resultado dos participantes, ele será executado apenas para encontrar problemas no material planejado para o estudo, permitindo que este material seja aprimorado antes de sua utilização.

6.1.2.4 Participantes

Os participantes do estudo serão um conjunto de desenvolvedores de software. O estudo não será executado em um ambiente da indústria de desenvolvimento de software, mas em um ambiente acadêmico. A capacidade de generalização deste estudo é discutida adiante, quando avaliamos as limitações e problemas que podem ser encontrados durante sua execução.

6.1.2.5 Instrumentação

Cada participante deverá atuar como desenvolvedor de um software. Serão desenvolvidos três softwares simples e todas as tarefas serão executadas por um terço dos participantes em todas as rodadas de execução de tarefas do estudo. A

especificação de cada tarefa foi dada aos participantes sem grandes detalhes, pois gostaríamos de simular uma situação onde o usuário precisa pesquisar como irá resolver determinado problema.

A seguir, uma breve descrição de cada tarefa:

Tarefa I: Protótipo de Interface de Navegador. A tarefa especificava a construção de uma interface de um navegador bem simples, semelhante à apresentada na figura abaixo. Essa interface deveria conter os três botões apresentados abaixo: "*Go*", para enviar a requisição HTTP (FIELDING, 1997); "*Back*", para voltar à última página navegada; e "*Forward*", para retornar para uma página navegada posteriormente (usado apenas quando se usou o botão "*Back*").

É importante ressaltar que nessa tarefa era necessário apenas o desenvolvimento dos componentes de interface. As ações que cada um dos componentes deveria executar seriam desenvolvidas separadamente em outra tarefa.

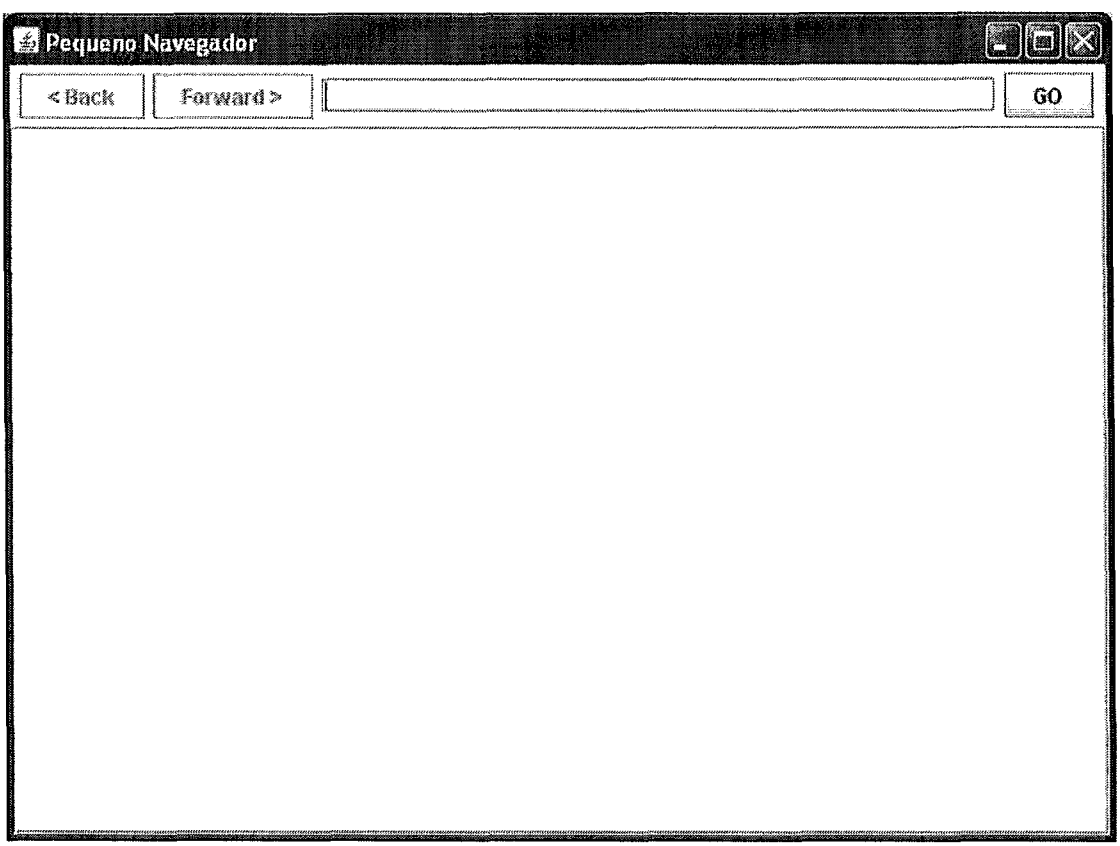


Figura 6.1 - Exemplo da interface solicitada na Tarefa 1.

Tarefa II: Classe que Retorna o HTML de uma URL. Essa tarefa é complementar a Tarefa I, porém não há qualquer tipo de dependência entre as duas. Nela foi solicitado ao participante que desenvolvesse uma classe simples que recebesse a URL (W3C, 2001) de uma página da Internet e retornasse seu conteúdo

HTML (W3C, 1999). Para isso seria necessário fazer uma requisição HTTP para obter o código HTML.

Tarefa III: Classe que Envia E-mail. Nesta tarefa os participantes deveriam desenvolver uma classe simples que recebesse um endereço de e-mail do remetente, seu servidor SMTP (KLENSIN, 2001), um endereço de e-mail do destinatário, o título da mensagem e o corpo da mensagem de e-mail. Com essas informações a classe deveria enviar o e-mail para o endereço do destinatário.

6.1.2.6 Critérios

O foco de qualidade do estudo exige critérios que avaliem os ganhos proporcionados pela utilização do sistema Olimpo e as dificuldades encontradas pelos usuários na sua utilização. Os ganhos obtidos pela utilização do sistema serão avaliados quantitativamente, através do tempo e número de páginas visitadas pelos participantes durante a execução das tarefas. Reconhecemos que o tempo e o número de páginas visitadas podem não ser os aspectos mais relevantes na pesquisa para realização das tarefas. Entretanto, estes critérios foram selecionados dada a necessidade de critérios quantitativos para comparação do desempenho dos participantes.

O módulo Argus responsável pela interação com os usuários no estudo será responsável por contabilizar estes dados, que determinarão o desempenho dos participantes. As dificuldades encontradas pelos usuários na utilização do sistema serão avaliadas através de dados qualitativos, levantados com base no questionário que será preenchido no fim do estudo.

6.1.2.7 Hipótese Nula

A hipótese nula é uma afirmativa que o estudo experimental tem como objetivo negar. No estudo atual, a hipótese nula determina que a utilização do sistema Olimpo não produz benefícios na aquisição de um conhecimento específico. De acordo com os critérios selecionados, esta hipótese se traduz na inexistência de diferenças significativas no tempo e no número de páginas navegadas para aquisição de conhecimento que permitissem a conclusão do desenvolvimento de softwares por participantes utilizando o sistema em relação a desenvolvimento de softwares utilizando-se apenas a navegação tradicional pela Web.

$$H_0 : \mu_{\#páginas\ \ sem\ \ sistema} = \mu_{\#páginas\ \ com\ \ sistema} \wedge \mu_{tempo\ \ sem\ \ sistema} = \mu_{tempo\ \ com\ \ sistema}$$

6.1.2.8 Hipótese Alternativa

A hipótese alternativa é uma afirmativa que nega a hipótese nula. O estudo experimental tem como objetivo provar a hipótese alternativa, refutando assim a hipótese nula. No estudo atual, a hipótese alternativa determina que os participantes do estudo que utilizarem o sistema Olimpo terão resultados superiores aos participantes que utilizarem apenas a navegação na Web. De acordo com os critérios selecionados, esta hipótese se traduz em menor número de páginas navegadas e tempo para conclusão das tarefas executadas por participantes utilizando as técnicas propostas em relação a tarefas executadas utilizando-se as a navegação convencional pela Web.

$$H_1 : \mu_{\#páginas\ \ sem\ \ sistema} > \mu_{\#páginas\ \ com\ \ sistema} \vee \mu_{tempo\ \ sem\ \ sistema} > \mu_{tempo\ \ com\ \ sistema}$$

6.1.2.9 Variáveis Independentes

A principal variável independente do estudo é um indicador que determina se cada participante utilizou ou não o sistema Olimpo (escala nominal). A formação e a experiência dos participantes, medidas em uma escala nominal, também são informações independentes coletadas durante o estudo que poderão ser utilizadas durante a análise para a formação de blocos.

6.1.2.10 Variáveis Dependentes

As variáveis dependentes são o número de páginas visitadas e tempo de desenvolvimento dos softwares. O número de páginas será medido como inteiro, de acordo com o número de páginas que o participante visitar que forem selecionadas como relevante pelo sistema. O tempo de desenvolvimento será medido em segundos (escala inteiro), indicando o tempo necessário para a conclusão do software

6.1.2.11 Análise Qualitativa

Com o objetivo de avaliar a satisfação dos participantes em relação às cadeias geradas pelo sistema e a qualidade do material utilizado no estudo. A análise qualitativa será realizada através de um questionário. A satisfação dos participantes será avaliada por perguntas envolvendo a classificação de quanto a cadeia de conhecimento gerada representou a navegação do participante e quanto a cadeia recomendada ajudou na execução da tarefa.

6.1.2.12 Capacidade Aleatória

Pode ser exercida na seleção dos participantes do estudo e na distribuição dos objetos de análise entre os participantes. Idealmente os indivíduos que realizarão o estudo devem ser selecionados aleatoriamente dentre o universo de candidatos a participantes, ou seja, dentre o conjunto das pessoas disponíveis que atendam aos critérios especificados na seção 6.1.2.4. Entretanto, se a seleção aleatória não for possível, ao menos os objetos de análise devem ser distribuídos aleatoriamente entre os participantes.

6.1.2.13 Classificação em Bloco

Os participantes serão divididos pelo sistema em três blocos. Essa divisão será feita por ordem de registro no sistema, onde o primeiro pertencerá ao grupo A, o segundo ao grupo B e o terceiro ao grupo C. Repetindo o ciclo para os demais participantes. A divisão em grupos permite que cada grupo execute uma determinada tarefa sem auxílio do sistema enquanto gerar dados para a segunda rodada do estudo.

A coleta de dados sobre a formação e a experiência dos participantes também permitirá sua futura classificação e a organização de blocos durante a análise dos dados.

6.1.2.14 Balanceamento

Durante a realização do estudo nos limitaremos a distribuir um número similar de participantes nos três grupos. Durante a análise, após a eliminação dos valores extremos, procuraremos um balanceamento, se este for possível.

6.1.2.15 Mecanismos de Análise

O estudo proposto se classifica com um experimento de dois tratamentos sobre um mesmo objeto, onde as variáveis dependentes são representadas na escala inteira. Além disso, o estudo compara as médias dos resultados obtidos pelos participantes dos dois grupos. Um potencial mecanismo de análise para este tipo de estudo é o teste T. O teste T (WOHLIN *et al.*, 2000) é um procedimento paramétrico de análise estatística baseado na distribuição T, que compara se as médias de dois grupos são equivalentes (de acordo com um grau de certeza, tal como 90% ou 95%) de acordo com suas variâncias.

Dois índices são amplamente empregados para avaliação de sistemas de busca e recuperação de informação: precisão e cobertura. Cada um definido pela respectiva fórmula:

$$precisão = \frac{|N \cap R|}{|R|} \tag{5}$$

$$cobertura = \frac{|N \cap R|}{|N|} \tag{6}$$

Onde N é o conjunto de respostas ideal e R o vetor resultado recuperado pelo sistema de busca.

O sistema aqui apresentado não é um sistema de busca e recuperação de informação, no entanto adaptamos esses índices para avaliar a precisão da classificação das páginas e a cobertura do sistema em relação à separação das páginas relevantes das não relevantes. Para isso, dois especialistas avaliarão cada URL visitada pelos participantes e definiram se essa são ou não relevante para o assunto em estudo, e classificarão se a classificação dada pelo sistema estava de acordo com o conteúdo da página.

Dessa forma as fórmulas foram adaptadas para:

$$precisão = \frac{|Cc \cap C|}{|C|} \tag{7}$$

$$cobertura = \frac{|R \cap C|}{|R|} \tag{8}$$

Onde Cc representa o número de páginas classificadas corretamente, C representa as páginas consideradas relevantes e selecionadas para classificação (páginas consideradas não relevantes não são classificadas) e R representa o número de páginas realmente relevantes.

6.1.2.16 Validade Interna do Estudo

A validade interna de um estudo é definida como a capacidade de um novo estudo repetir o comportamento do estudo atual com os mesmos participantes e objetos com que ele foi realizado. A validade interna do estudo é dependente do número de participantes executando o estudo. Esperamos contar com pelo menos doze

participantes, o que garante um bom nível de validação interna do estudo. Certamente, um número maior de participantes melhoraria a validade interna do estudo. Outro ponto que pode influenciar o resultado do estudo é a troca de informações entre os participantes que já realizaram o estudo e os que não o realizaram. Para evitar este problema, requisitaremos explicitamente que os participantes não troquem informações a respeito do projeto. Tentaremos ainda realizar o estudo em uma única sessão, onde os participantes trabalharão paralelamente, sem trocar informações. Finalmente, o projeto não será apresentado como uma competição, inibindo a comparação de resultados entre os participantes.

6.1.2.17 Validade Externa do Estudo

A validade externa do estudo mede sua capacidade de refletir o mesmo comportamento em outros grupos de participantes e profissionais da indústria, ou seja, em outros grupos além daquele em que o estudo foi aplicado. Acreditamos que o maior problema em relação à validade externa do estudo é a falta de interesse dos participantes no próprio estudo. Alguns indivíduos podem realizar o estudo de forma desleixada, sem um interesse real no desenvolvimento das tarefas em menor tempo. A validade externa do estudo é considerada suficiente, visto que o presente estudo visa avaliar a viabilidade de aplicação do sistema de recomendação de cadeias de conhecimento. Demonstrada esta viabilidade, novos estudos podem ser planejados para refinar o universo de aplicação das técnicas.

6.1.2.18 Validade de Construção do Estudo

A validade de construção do estudo se refere à relação entre os instrumentos e participantes do estudo e a teoria que está sendo provada por este. Neste caso, escolhemos um domínio de aplicação amplamente conhecido, neutralizando o efeito da experiência dos participantes no domínio. Esta escolha evita que experiências anteriores gerem uma interpretação incorreta do impacto da arquitetura proposta. Os dados que serão utilizados para calibrar o processo de referência dizem respeito a tarefas com grau de dificuldade conhecido pelos avaliadores, reduzindo o erro de ajuste no modelo. Além disso, o estudo não visa avaliar a correção das cadeias de conhecimento, mas a capacidade de utilização dessas cadeias no aprendizado. Assim, as cadeias serão compatíveis com as tarefas propostas aos participantes.

6.1.2.19 Validade de Conclusão do Estudo

A validade de conclusão do estudo mede a relação entre os tratamentos e os resultados, determinando a capacidade do estudo em gerar alguma conclusão. Não encontramos grandes dificuldades em relação à capacidade de conclusão do estudo, visto que esta pode ser traçada a partir de um mecanismo de análise estatística amplamente utilizado, como o teste T (as escalas das variáveis dependentes e independente assim o permitem). O teste T possui alto poder estatístico e não assume normalidade ou qualquer outra distribuição nos dados analisados. Além disso, o estudo utiliza medidas objetivas, o que neutraliza a influência humana sobre os dados apurados e analisados.

6.1.3 Execução do Estudo de Observação

6.1.3.1 Seleção dos Participantes

Para a presente execução do estudo, selecionamos os participantes dentre os alunos dos cursos de Mestrado e Doutorado em Banco de Dados do programa de pós-graduação em Engenharia de Sistemas da COPPE. Estes participantes atendem às restrições indicadas no planejamento (vide seção 3.2, parágrafo *Participantes*), visto que são desenvolvedores de software. Os participantes foram selecionados por conveniência, representando um subconjunto não aleatório do universo de alunos da instituição supracitada.

6.1.3.2 Instrumentação

O *plug-in* Argus foi desenvolvido para o navegado Mozilla Firefox e será responsável por interagir com o usuário. Argus comunicava-se com o serviço Web Hera quando o participante solicitava uma tarefa para iniciar o estudo.

Hera possui um mecanismo que distribui as três diferentes tarefas de forma homogênea entre os participantes. Dessa forma, o primeiro participante que iniciasse o experimento recebia do próprio sistema a incumbência de executar a Tarefa I, o participante seguinte receberia a Tarefa II e o seguinte a Tarefa III, repetindo o ciclo para os participantes seguintes.

Uma vez que o *plug-in* estivesse configurado, o participante poderia acionar o observador do Argus e iniciar a tarefa que recebeu. Uma vez que tivessem concluído a tarefa, deveriam desligar o observador do Argus e clicar na opção referente à tarefa

que recebeu no menu "Suas Cadeias" do Módulo Hermes. Este módulo irá classificar as páginas navegadas e retornar para o participante a cadeia de conhecimento navegada por ele nessa etapa.

Figura 6.2 - Menu do Módulo Hermes para obter a cadeia navegada pelo participante

Ao terminar a primeira rodada de tarefas, os participantes voltaram a clicar no botão "Iniciar nova tarefa" para receber do serviço Web Hera a próxima tarefa que deveria executar. A redistribuição de tarefas segue a seguinte lógica:

- Participante de executou a Tarefa I recebe a Tarefa II;
- Participante de executou a Tarefa II recebe a Tarefa III;
- Participante de executou a Tarefa III recebe a Tarefa I.

O participante deveria reativar o observador do Argus e nessa rodada solicitar uma cadeia de referência para a execução da tarefa que recebeu. Para isso, o participante deveria clicar na opção referente à tarefa que recebeu no menu "Cadeias de Referência" no módulo Hermes. Este módulo retornaria ao participante uma cadeia de referência de um dos participantes (escolhido aleatoriamente) que executaram aquela tarefa na rodada anterior. Com essas cadeias recomendadas, os participantes tinham a possibilidade de clicar em determinado conceito e obter a lista de páginas que foram visitadas e classificadas com aquele conceito ordenadas por grau de relevância.

Figura 6.3 - Menu do Módulo Hermes para obter a recomendação de cadeia de conhecimentos

Mais uma vez, ao terminar a execução da tarefa o usuário deveria desativar o observador e ver a cadeia de conhecimento gerada pelo Hermes para a execução dessa segunda rodada de tarefas.

A terceira rodada de tarefas deve ser iniciada como as demais com a diferença que nessa, quando o participante solicitar uma cadeia de conhecimento como referência para executar sua tarefa, o Módulo Hermes irá, automaticamente, identificar que se tratava da terceira rodada e irá fornecer ao participante uma cadeia específica previamente criada por um especialista exclusivamente para a realização dessa tarefa.

Como as cadeias recomendadas na etapa anterior, os participantes também tiveram acesso a páginas de referência relacionadas a cada um dos conceitos da cadeia.

Mais uma vez o Módulo Hermes disponibilizou para o usuário a cadeia gerada pela sua navegação para execução dessa rodada de tarefas.

Todos os participantes utilizaram o sistema Olimpo e ele mantinha o controle de quando determinado participante poderia ou não utilizar uma cadeia de conhecimento de referência.

Para a análise qualitativa os participantes deveriam clicar no botão "Responder..." na janela de configuração do *plug-in* Argus, para que o sistema disponibilizasse um formulário onde poderiam:

- Responder perguntas que os permitem uma auto-análise sobre seus conhecimentos na linguagem Java;
- Avaliar a qualidade das cadeias geradas a partir de suas próprias navegações;
- Avaliar o quanto às cadeias recomendadas os ajudaram a executar as tarefas.

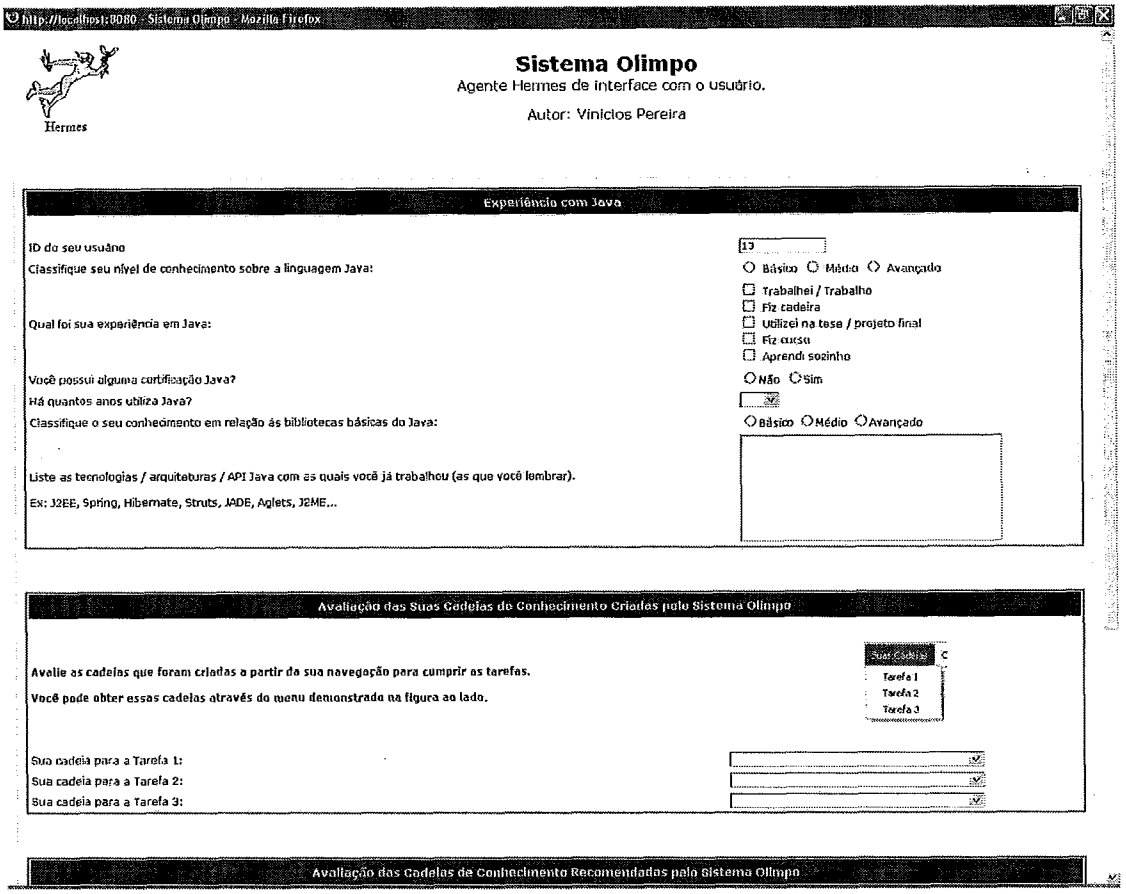


Figura 6.4 - Formulário para identificar o perfil e a avaliação dos participantes

Foram feitas então as seguintes perguntas:

- Experiência com Java

1) Classifique seu nível de conhecimento sobre a linguagem Java:
   a) Básico
   b) Médio
   c) Avançado
2) Qual foi sua experiência com Java? (Múltiplas opções)
   a) Trabalhei
   b) Fiz cadeira
   c) Utilizei na tese / projeto final
   d) Fiz curso
   e) Aprendi sozinho
3) Você possui alguma certificação Java?
4) Há quantos anos você utiliza Java?
5) Classifique o seu conhecimento em relação às bibliotecas básicas do Java
   a) Básico

b) Médio

c) Avançado

6) Liste as tecnologias/arquiteturas/API Java com as quais você já trabalhou (as que você lembrar). Ex: J2EE, Spring, Hibernate, Struts, JADE, Aglets, J2ME...

- Avaliação das Suas Cadeias de Conhecimento Criadas pelo Sistema Olimpo

7) Sua cadeia para a Tarefa 1:

a) Não representou sua navegação

b) Representou sua navegação com imprecisão

c) Representou sua navegação

d) Representou sua navegação com precisão

8) Sua cadeia para a Tarefa 2:

a) Não representou sua navegação

b) Representou sua navegação com imprecisão

c) Representou sua navegação

d) Representou sua navegação com precisão

9) Sua cadeia para a Tarefa 3:

a) Não representou sua navegação

b) Representou sua navegação com imprecisão

c) Representou sua navegação

d) Representou sua navegação com precisão

- Avaliação das Cadeias de Conhecimento Recomendadas pelo Sistema Olimpo

10) Cadeia recomendada para a Tarefa 1:

a) Não ajudou em nada na execução da tarefa

b) Ajudou muito pouco na execução da tarefa

c) Ajudou na execução da tarefa

d) Ajudou muito na execução da tarefa

11) Cadeia recomendada para a Tarefa 2:

a) Não ajudou em nada na execução da tarefa

b) Ajudou muito pouco na execução da tarefa

c) Ajudou na execução da tarefa

d) Ajudou muito na execução da tarefa

12) Cadeia recomendada para a Tarefa 3:

a) Não ajudou em nada na execução da tarefa

b) Ajudou muito pouco na execução da tarefa

c) Ajudou na execução da tarefa

d) Ajudou muito na execução da tarefa

6.1.3.3 Procedimento de Participação

Nesse estudo existe apenas um procedimento de participação, uma vez que todos os participantes executam uma tarefa sem ajuda, uma tarefa com recomendação de uma cadeia de conhecimento de um colega e uma tarefa com recomendação de uma cadeia gerada por um especialista.

Procedimento de participação:

1. Todos os participantes forma reunidos no local de realização do estudo de observação.
2. O participante recebe treinamento com base nas transparências.
3. O participante executa o navegador Mozilla Firefox.
4. O *plug-in* Argus associa um número ao participante.
5. O participante solicita ao Argus a primeira tarefa que deve executar.
6. O Argus define por qual tarefa o participante deve começar.
7. O participante ativa o registro de páginas do Argus e inicia sua pesquisa para conclusão da tarefa.
8. O participante executa a tarefa enquanto pesquisa na Web informações para a execução.
9. O participante conclui a tarefa e desativa o registro de páginas do Argus.
10. O Argus apresenta ao participante a cadeia de conhecimento gerada a partir da sua navegação.
11. O participante solicita uma nova tarefa.
12. O Argus define uma nova tarefa ao participante.
13. O participante solicita a recomendação de uma cadeia de conhecimento sobre a tarefa que está sendo executada.
14. O Argus recomenda uma cadeia retornada pelo módulo Hermes.
15. O participante ativa o registro de páginas do Argus e inicia sua pesquisa para conclusão da nova tarefa.
16. O participante executa a tarefa enquanto pesquisa na Web informações para a execução.

17. O participante conclui a tarefa e desativa o registro de páginas do Argus.
18. O Argus apresenta ao participante a cadeia de conhecimento gerada a partir da sua navegação.
19. O participante solicita uma nova tarefa.
20. O Argus define uma nova tarefa ao participante.
21. O participante solicita a recomendação de uma cadeia de conhecimento sobre a tarefa que está sendo executada.
22. O Argus recomenda uma cadeia retornada pelo módulo Hermes.
23. O participante ativa o registro de páginas do Argus e inicia sua pesquisa para conclusão da nova tarefa.
24. O participante executa a tarefa enquanto pesquisa na Web informações para a execução.
25. O participante conclui a tarefa e desativa o registro de páginas do Argus.
26. O Argus apresenta ao participante a cadeia de conhecimento gerada a partir da sua navegação.
27. O participante informa ao Argus que finalizou suas tarefas.
28. O Argus apresenta o questionário para que o participante preencha.
29. O participante preenche o formulário e envia os programas criados para o responsável pelo estudo.

6.1.3.4 Execução

O estudo de observação do sistema Olimpo contou com a participação de onze alunos de mestrado inscritos na cadeira de Trabalho Cooperativo Suportado por Computador (*Computer Supported Cooperative Work* - CSCW), cadeira oferecida pela linha de Banco de Dados do Programa de Engenharia de Sistemas da COPPE. O estudo aqui descrito foi modelado para durar duas horas, tempo de duração da aula da cadeira.

Neste estudo de observação os participantes foram divididos pelo sistema em três grupos distintos. Cada grupo recebeu uma tarefa inicial distinta. Cada tarefa descreve um exercício básico na linguagem de programação Java, assunto escolhido para nortear este experimento devido ao elevado grau de interesse dos participantes. Ao receber suas tarefas, os participantes deveriam executar o exercício fazendo as

pesquisas necessárias na Web utilizando o navegador Mozilla Firefox versão 2.0.0.3 ou superior (MOZILLA, 2007). O sistema registrou todas as atividades relacionadas à execução de cada tarefa e contabilizou o tempo necessário para a sua conclusão.

Tabela 6.1 – Informações detalhadas sobre os participantes do estudo de observação

ID	Ordem das Tarefas	Conhecimento Java	Experiência	Conhecimento Bibliotecas	Tecnologias
1	2, 3, 1	Avançado	Usou no trabalho Usou na faculdade Aprendeu sozinho Experiência de 3 anos	Avançado	J2EE, Hibernate, JADE
2	1, 2, 3	Avançado	Usou no trabalho Usou na faculdade Aprendeu sozinho Experiência de 3 anos	Avançado	J2EE, JSF, Hibernate, Seam, Cocoon, DOM
3	2, 3, 1	Médio	Usou na faculdade Experiência de 3 anos	Médio	JME
4	1, 2, 3	Básico	Usou no trabalho Fez curso Aprendeu sozinho Experiência de 1 ano	Médio	J2EE, Hibernate, Struts
5	3, 1, 2	Nenhum		Nenhum	
6	1, 2, 3	Médio	Usou no trabalho Aprendeu sozinho Experiência de 1 ano	Básico	Hibernate, Struts, EJB, Servlet, Xalan, JDOM, JSP, JDBC
7	2, 3, 1	Médio	Usou no trabalho Aprendeu sozinho Experiência de 3 anos	Médio	J2SE, J2EE, Hibernate, Struts
8	3, 2, 1	Básico	Usou na faculdade Experiência de 1 ano	Básico	JENA, Swing
9	3, 2, 1	Básico	Usou na faculdade Experiência de 2 anos	Básico	J2EE, JSP

10	1, 2, 3	Avançado	Usou no trabalho Fez curso Aprendeu sozinho Possui certificação Experiência de 4 anos	Médio	J2EE, Hibernate, Struts, Aglets
11	1, 2, 3	Médio	Usou no trabalho Aprendeu sozinho Experiência de 3 anos	Médio	J2SE, APPLETS

Através das respostas dadas pelos próprios participantes foi possível identificar que a maioria (37%) dos participantes julgava seu conhecimento na linguagem em um nível médio, enquanto duas partes iguais de 27% julgavam seus conhecimentos básicos ou avançados. Apenas uma pessoa possuía nenhum ou muito pouco conhecimento na linguagem.

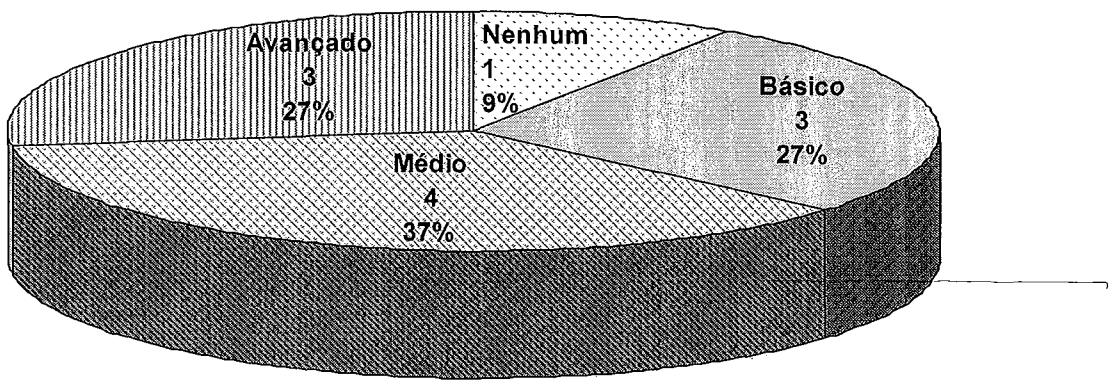


Figura 6.5 - Participantes distribuídos pela fatia de conhecimento em Java

As tabelas e figuras a seguir representam quantos participantes responderam ter ou não determinada habilidade.

Tabela 6.2 - Experiência com Java

Experiência	Número de Participantes	Porcentagem
Trabalhou	7	63,64 %
Fez cadeira na faculdade	4	36,36 %
Utilizou na tese / projeto final	4	36,36 %
Fez curso	3	27,27 %
Aprendeu sozinho	7	63,64 %

Possui alguma certificação Java	1	9,09 %

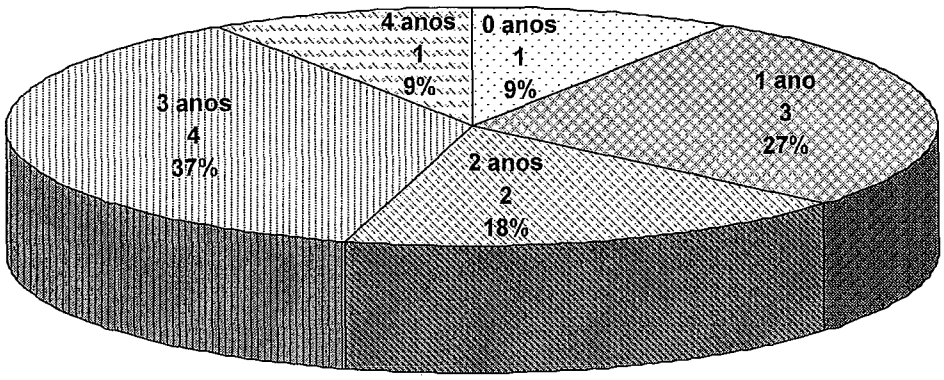


Figura 6.6 - Há quantos anos trabalha com Java

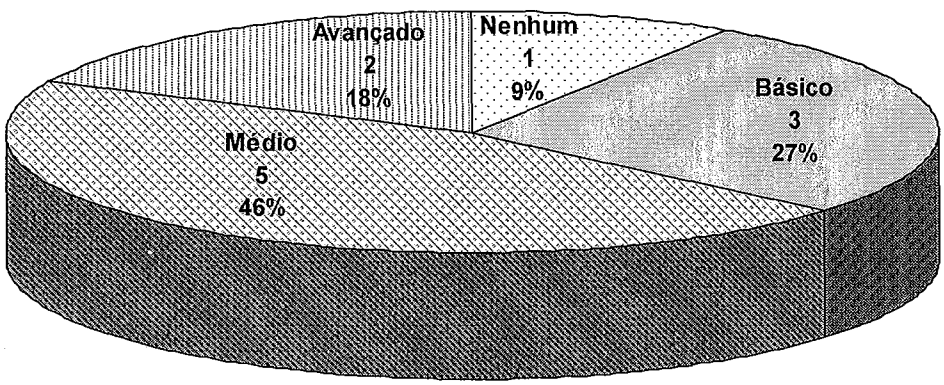


Figura 6.7 - Conhecimento das bibliotecas básicas da linguagem Java

Os participantes foram orientados a citar tecnologias em Java às quais já tiveram contato. Foram mencionadas: JEE, Hibernate, JADE, JSF, Seam, Cocoon, JME, Struts, EJB, Servelet, Xalan, JDOM, JSP, JDBC, J2SE, JENA, Aglets e Applets.

A partir dessas informações, os participantes, em sua grande maioria, foram considerados aptos a representarem profissionais da área por mostrarem qualificação razoável. De todos participantes do experimento, apenas um não alcançou os objetivos das tarefas e, por isso, não foi considerado na análise com o restante do grupo. Este caso foi analisado separadamente.

6.1.4 Análise dos Resultados do Estudo de Observação

6.1.4.1 Avaliação Quantitativa

Tabela 6.3 - Resultados obtidos no estudo de observação

ID	1ª Rodada			2ª Rodada			3ª Rodada		
	Tarefa	Nº. páginas	Tempo (s)	Tarefa	Nº. páginas	Tempo (s)	Tarefa	Nº. páginas	Tempo (s)
1	2	6	1598	3	2	217	1	1	120
2	1	15	2731	2	10	682	3	5	199
3	2	Não finalizou a tarefa.		3	Não finalizou a tarefa.		1	Não finalizou a tarefa.	
4	1	13	708	2	1	120	3	13	1196
5	3	36	1947	1	10	643	2	7	119
6	1	Não finalizou a tarefa.		2	Não finalizou a tarefa.		3	Não finalizou a tarefa.	
7	2	5	272	3	4	929	1	1	120
8	3	22	2369	2	5	807	1	16	1712
9	3	26	4213	2	2	305	1	2	305
10	1	8	413	2	5	945	3	9	848
11	1	1	120	2	3	65	3	13	628

A análise quantitativa do estudo de observação pode ser separada em análise de número de páginas visitadas e análise de tempo. Cada uma destas análises foi realizada em duas etapas: eliminação de valores extremos e teste paramétrico de média. Os resultados obtidos no estudo experimental são apresentados na Tabela 6.3.

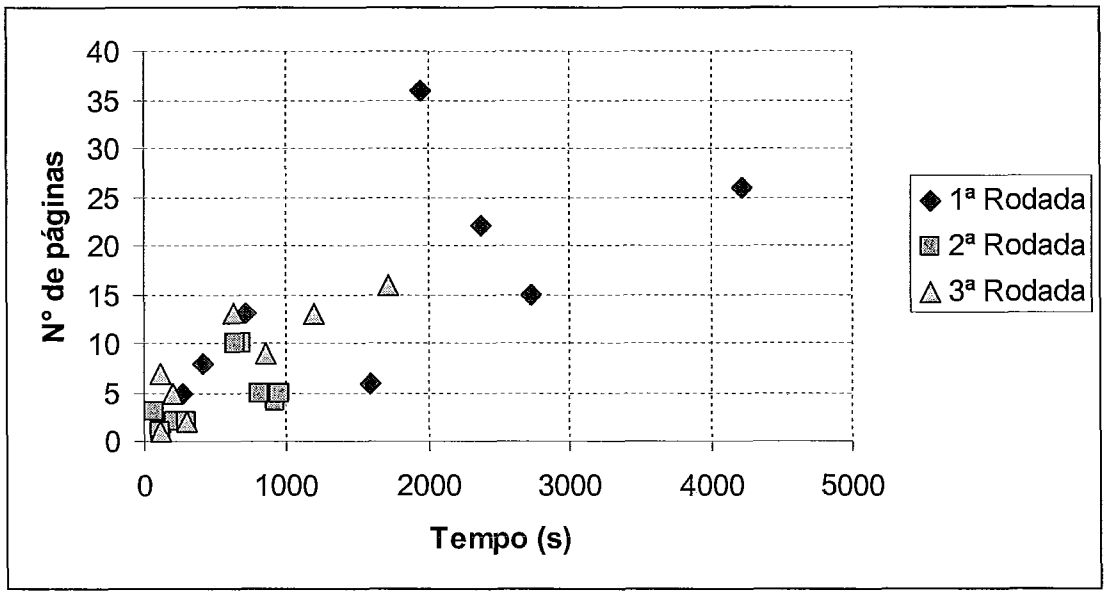


Figura 6.8 - Resultados obtidos no estudo de observação

6.1.4.2 Eliminação de Valores Extremos

Tabela 6.4 - Resultados obtidos no estudo de observação com corte T

ID	1ª Rodada			2ª Rodada			3ª Rodada		
	Tarefa	Nº. páginas	Tempo (s)	Tarefa	Nº. páginas	Tempo (s)	Tarefa	Nº. páginas	Tempo (s)
1	2	6	1598	3	2	217	1	~~1~~	120
2	1	15	2731	2	10	682	3	5	199
4	1	13	708	2	~~1~~	~~120~~	3	~~13~~	~~1196~~
5	3	~~36~~	1947	1	~~10~~	643	2	7	119
7	2	5	~~272~~	3	~~4~~	~~929~~	1	~~1~~	120
8	3	22	2369	2	5	807	1	~~16~~	~~1712~~
9	3	~~26~~	~~4213~~	2	2	305	1	2	305
10	1	8	413	2	5	~~945~~	3	9	848
11	1	~~1~~	~~120~~	2	~~3~~	~~65~~	3	~~13~~	628

Tendo coletado as informações de tempo e número de páginas navegadas através do módulo Argus, passamos para a etapa de eliminação de valores extremos. Nesta etapa, aplicamos um corte baseado na distribuição T em 99% (FREUND, PERLES, 1998). Este método de corte é análogo ao corte pela curva normal, devendo

ser utilizado quando não se pode assumir que a população analisada segue uma distribuição normal e o universo de análise contém um número reduzido de informações ($n < 30$).

Na análise de tempo de conclusão da tarefa, o corte T eliminou três participantes do grupo que não utilizou as cadeias de conhecimento recomendadas, quatro participantes do grupo que utilizou as cadeias criadas a partir da navegação dos outros participantes e dois participantes do grupo que utilizou as cadeias criadas pelo especialista. Na análise do número de páginas visitadas, o corte T eliminou três participantes do primeiro grupo, quatro participantes do segundo grupo e cinco participantes do terceiro grupo. A Tabela 6.4 apresenta os resultados do estudo de observação, grifando os valores eliminados pelo corte T. A Tabela 6.5 apresenta estatísticas descritivas sobre os resultados do estudo após a eliminação dos valores extremos.

Tabela 6.5 - Estatísticas descritivas sobre os resultados do estudo (após a eliminação de valores extremos)

Estatística		Nº. páginas	Tempo (s)
Sem cadeias de conhecimento	Média	11,50	1627,67
	Mediana	10,50	1772,50
	Mínimo	5	413
	Máximo	22	2731
	Desvio padrão	6,47	915,53
Com cadeias de conhecimento dos participantes	Média	3	530,80
	Mediana	2	643
	Mínimo	2	217
	Máximo	5	807
	Desvio padrão	1,41	255,54
Com cadeias de conhecimento dos participantes	Média	4,75	334,14
	Mediana	5	199
	Mínimo	2	119
	Máximo	7	848
	Desvio padrão	2,06	290,84

6.1.4.3 Teste Paramétrico

Os dados que permaneceram após a eliminação de valores extremos foram submetidos a uma análise paramétrica baseada na distribuição T (WOHLIN *et al.*, 2000). Na análise de tempo, o teste T foi conclusivo no sentido de afirmar que o tempo médio que os participantes que utilizaram a cadeias de conhecimento de outros participantes levaram para concluir as tarefas propostas foi inferior ao tempo médio que os participantes que não utilizaram cadeias de conhecimento levaram para concluir as tarefas. Os resultados intermediários do teste T para o critério tempo são apresentados na Tabela 6.6.

Tabela 6.6 - Resultados intermediários para o Teste T para análise do tempo (90%)

Desvio	T_0	Liberdade	Distribuição T	Resultado ($T_0 < -T$)
594,61	-5,05	4	2,13	$\mu_{\text{tempo s/ sistema}} > \mu_{\text{tempo c/ sistema}}$
235,59	-2,71	2	2,92	Inconclusivo

O teste T para análise de tempo que comparou o tempo médio entre os participantes que utilizaram as cadeias de conhecimento geradas através da navegação dos outros participantes e o tempo médio dos participantes que utilizaram as cadeias geradas pelo especialista foi inconclusivo, o que não nos permite afirmar nada em relação ao uso de geração de cadeias de conhecimento mais detalhadas. Entretanto, observamos que, com relação ao critério tempo, a mediana do grupo que utilizou as cadeias geradas pelo especialista é menor que a mediana do grupo que utilizou as cadeias geradas pela navegação dos outros participantes. Observamos ainda que a diferença entre as medianas é significativamente maior do que a diferença entre as médias. Esta diferença indica que as distribuições que modelam os grupos não são simétricas e que a distribuição do grupo que utilizou as cadeias do especialista está concentrada em torno de tempos menores.

Tabela 6.7 - Resultados intermediários para o Teste T para análise do número de páginas visitadas (90%)

Desvio	T_0	Liberdade	Distribuição T	Resultado ($T_0 < -T$)
6,45	-3,18	3	2,35	$\mu_{\#\text{ páginas s/ sistema}} > \mu_{\#\text{ páginas c/ sistema}}$
3,46	-0,50	2	2,92	Inconclusivo

A Tabela 6.7 nos mostra que uma análise semelhante à realizada para o tempo pode ser aplicada ao número de páginas visitadas. No entanto, para esse caso

específico não pudemos observar que a diferença entre as medianas indique que o uso das cadeias de conhecimento geradas pelo especialista possam trazer alguma redução dessa variável.

6.1.4.4 Análise da Execução das Tarefas

Em relação à execução das tarefas, era esperado que na primeira rodada o número de páginas visitadas e o tempo utilizado para sua conclusão fossem maiores que nas demais rodadas, uma vez que os participantes não receberiam qualquer ajuda. Na segunda rodada de tarefas, era esperado que, com a ajuda da cadeia de outro participante, o número de páginas visitadas e o tempo para conclusão da tarefa fossem reduzidos até que minimizassem na terceira rodada com auxílio da cadeia de conhecimento do especialista.

A Figura 6.9 mostra o número médio de páginas visitadas para execução de cada tarefa em cada rodada. Nela é possível observar que a Tarefa 1 comporta-se exatamente como o esperado, reduzindo o número de páginas em casa rodada. No entanto, as tarefas 2 e 3 sofrem uma redução na segunda rodada, mas o número de páginas voltam a subir na terceira rodada. No caso específico da Tarefa 2, o número de páginas da terceira rodada chega a ser superior ao da primeira.

Este comportamento também pode ser observado na Figura 6.9, que mostra o tempo em segundos necessário para a execução de cada tarefa em cada rodada do estudo de observação. Mais uma vez, na Tarefa 2 o tempo utilizado na terceira rodada foi maior que o necessário na segunda rodada.

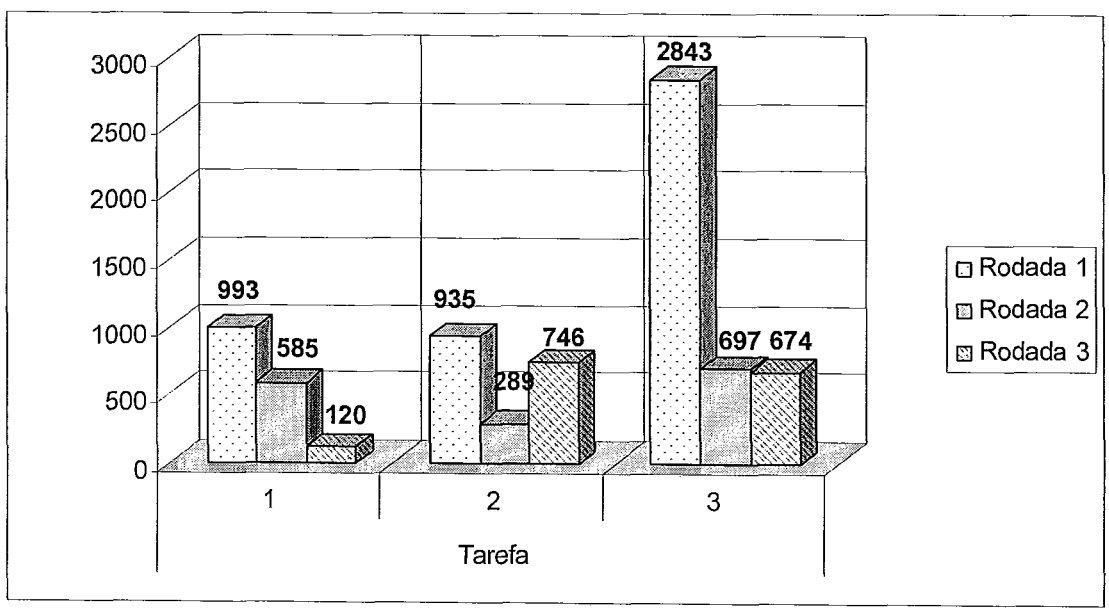


Figura 6.9 - Tempo em segundos para conclusão das tarefas

O sistema atingiu perfeitamente seu objetivo de agilizar e melhorar o foco do estudo dos participantes em relação à segunda rodada de tarefas. No entanto, este comportamento inesperado na terceira rodada só será compreendido após a análise das avaliações feitas pelos participantes das cadeias de conhecimento.

Após o término do estudo de observação alguns participantes tomaram a iniciativa de expressar uma avaliação geral do sistema, para isso fizeram comentários presenciais ou enviaram e-mails onde alguns salientavam o fato das cadeias recomendadas pelo especialista possuírem uma estrutura didática melhor, uma vez que possuíam referências a termos básicos como classe, objeto, pacotes gráficos, etc. Ao passo que as cadeias recomendadas a partir da navegação dos outros participantes eram mais objetivas e orientadas a solução do problema.

Um dos participantes do estudo de observação escreveu o seguinte na mensagem de e-mail enviada: "... considerei a cadeia de referência da tarefa 2 (cadeia de referência com ajuda especial) bastante útil para guiar o aprendizado nessa área de conhecimento."

Essa estrutura mais detalhada e didática estimulou os participantes a navegar por páginas conceituais que não só solucionariam seus problemas imediatos, mas aumentariam seus conhecimentos na linguagem. Este comportamento se refletiu na média de aprovação das cadeias, conforme apresentado na Figura 6.14. Nela podemos observar que os participantes, em média, avaliaram que as cadeias recomendadas pelo especialista foram melhores que as recomendadas pelos outros participantes, mesmo que essas fossem menos objetivas.

De fato, uma análise mais próxima comparando as cadeias geradas pela navegação dos participantes e as cadeias criadas pelo especialista mostram que este segundo grupo de cadeias é bem mais detalhado. A Figura 6.10 compara o número médio de nós nas cadeias geradas pelos participantes com as cadeias geradas pelo especialista em cada tarefa. A diferença é de duas a sete vezes o número de nós nas cadeias de conhecimento do especialista.

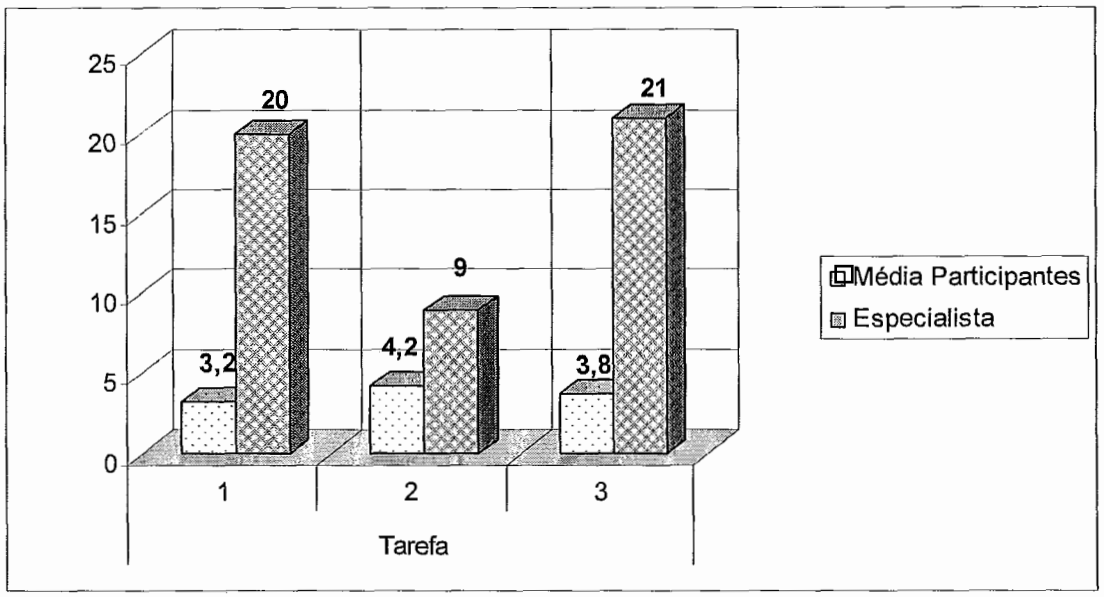


Figura 6.10 - Número médio de nós nas cadeias de conhecimento

No Apêndice C é possível observar nas figuras C2, C3 e C4 que comparam os conceitos que foram navegados para cumprir as tarefas na primeira rodada (onde os participantes não tiveram ajuda) em relação aos termos recomendados pelo especialista. Nessas figuras podemos observar que poucos termos foram de fato navegados para que a solução fosse encontrada. Por outro lado, nas figuras C5, C6 e C7, que comparam os conceitos navegados pelos usuários com a ajuda da cadeia de conhecimento do especialista, vários conceitos navegados pelos participantes coincidem com os conceitos sugeridos pelo especialista. Havendo inclusive visitas a termos conceituais como Java e classe, onde os participantes tinham ciência que não encontrariam a solução do problema.

Se a recomendação de uma cadeia mais completa estimulou os participantes a estudarem conceitos de Java que não estavam diretamente ligados a resolução das tarefas, então poderíamos inferir que os participantes com menos experiência em Java teriam mais conceitos nessa cadeia que não conheciam a fundo. Partindo dessa hipótese poderíamos esperar que quanto menor a experiência do participante, mais páginas não objetivas ele visitou na terceira rodada e, consequentemente, mais tempo ele levou para cumprir a tarefa.

A análise representada na Figura 6.11 mostra que a média de páginas visitadas pelos participantes com experiência avançada em Java já é relativamente pequena na primeira rodada de tarefas, onde não houve qualquer ajuda do sistema. Na segunda rodada, onde o sistema recomendou uma cadeia de conhecimento, houve uma suave

queda no número de páginas visitadas. Na terceira rodada, com auxílio da cadeia de conhecimento gerada pelo especialista, os participantes com experiência avançada reduziram ainda mais sua média de páginas visitadas. Essa análise sugere que para usuários com conhecimento mais profundo no assunto é possível que a recomendação de cadeias de conhecimento torne o aprendizado mais objetivo e ágil. A Figura 6.12, que mostra o tempo médio de execução das tarefas discriminado por experiência em Java dos participantes, nos permite a mesma análise.

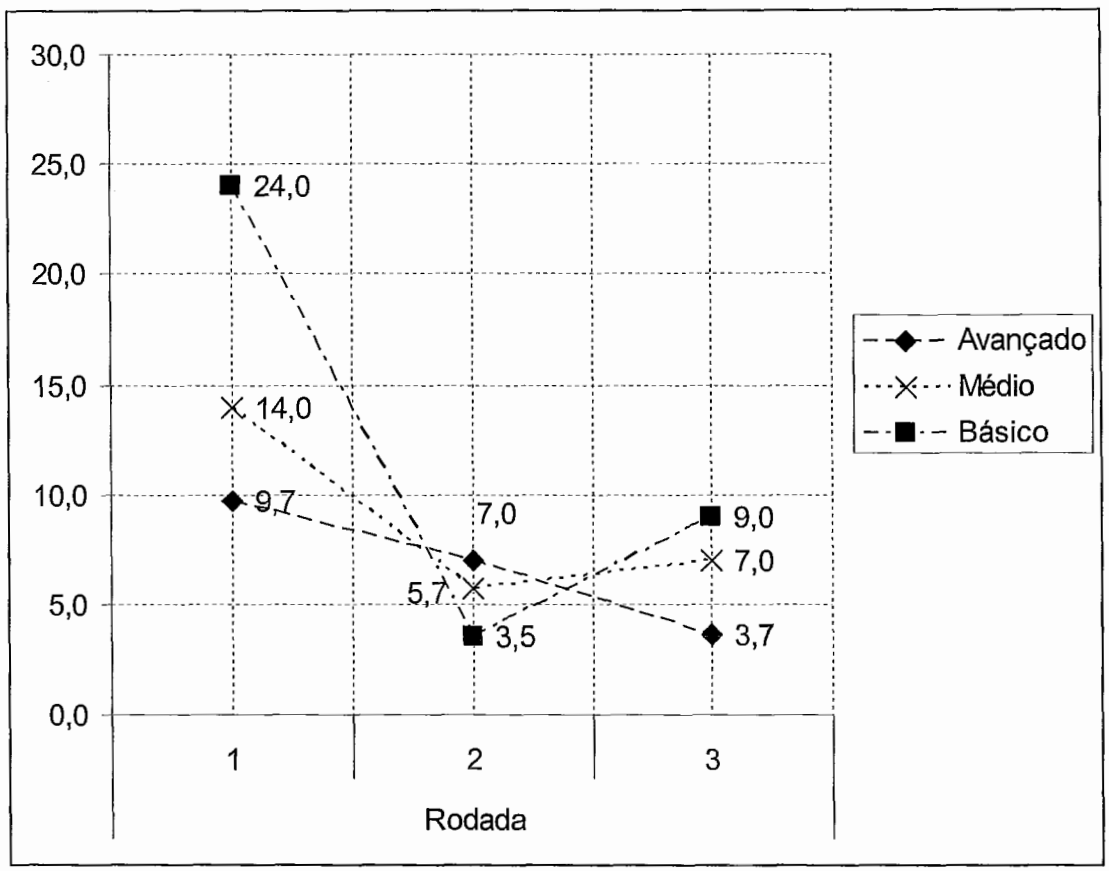


Figura 6.11 - Média de número de páginas navegadas por rodada por experiência em Java

Por outro lado, quando analisamos a média de páginas visitadas pelos participantes com experiência média em linguagem de programação Java, percebemos que houve um ligeiro aumento na terceira rodada em relação à segunda. Esse aumento é tão sutil que não chega a fazer com que o tempo médio de execução acompanhe esse aumento, como podemos observar na Figura 6.12. Isso pode sugerir que, também nesse nível de experiência, a recomendação de cadeias de conhecimento torna o aprendizado mais objetivo e ágil. No entanto, é possível que a recomendação de uma cadeia mais completa e detalhada induza o aprendiz a navegar por novas informações ou talvez o induza a dispersão do seu objetivo de aprendizado.

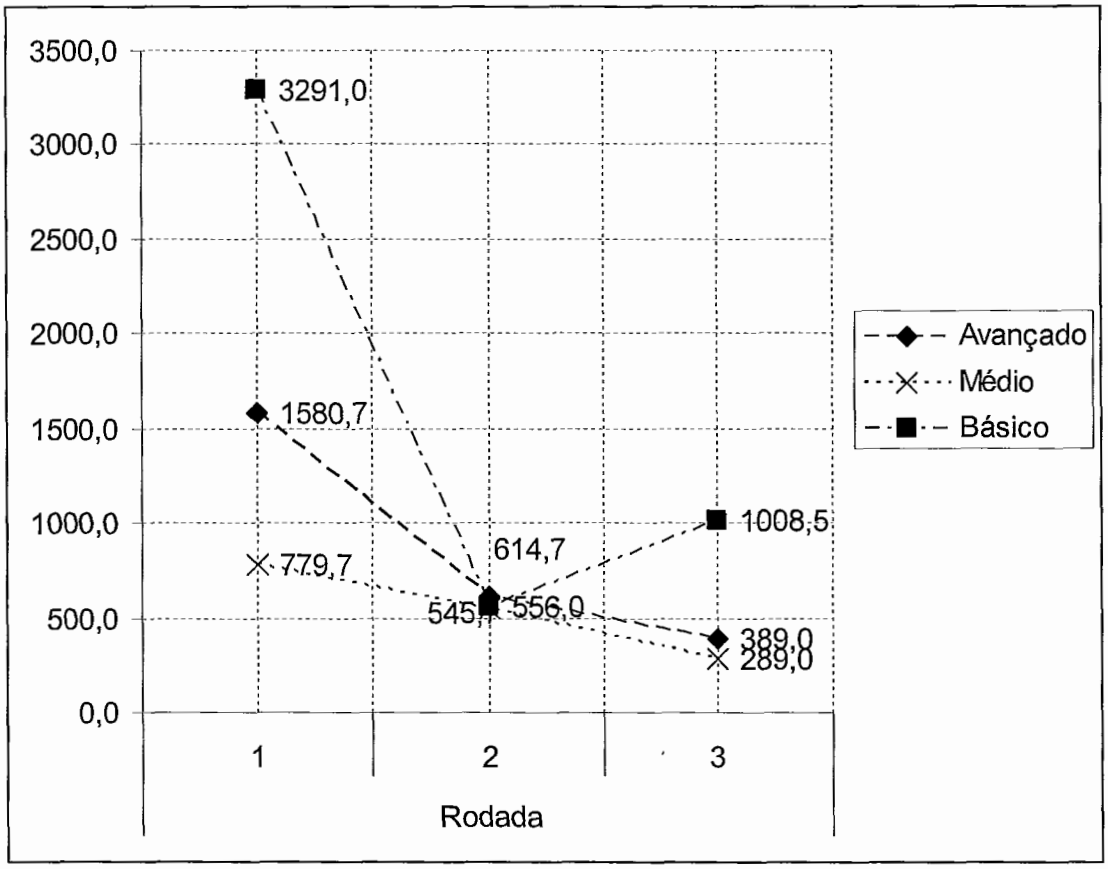


Figura 6.12 - Média do tempo (s) por rodada por experiência em Java

O comportamento observado na análise dos participantes com experiência média fica muito mais evidente quando empregamos a mesma análise nos participantes com experiência básica em Java. Para esses, tanto o número médio de páginas visitadas, quanto o tempo médio da execução das tarefas, sobe consideravelmente. Apesar desses valores não ultrapassarem as médias obtidas na primeira rodada de tarefas, esse aumento pode indicar que a mesma curiosidade que foi instigada para os participantes com experiência média também foi provocada nos participantes com experiência básica. No entanto, esses têm conhecimento mais restrito e permaneceram mais tempo navegando pelo conteúdo recomendado. Uma análise mais pessimista poderia sugerir que a falta de conhecimento desses participantes os levaram a uma dispersão maior nos conceitos de uma cadeia de conhecimento mais detalhada.

Essa análise sugere que o sistema pode não só permitir uma navegação de estudo mais focada como também estimular a aquisição e troca de conhecimento se usado ao longo do tempo por uma comunidade.

6.1.4.5 Análise da Precisão e Cobertura

Dois especialistas em linguagem de programação Java analisaram todas as páginas navegadas e às classificaram de acordo com os conceitos da ontologia de Java, além disso, eles indicaram se as páginas eram relevantes ou não. Após a avaliação dos especialistas identificamos que os conceitos descritos na seção 6.1.2.15 receberam os valores:

- Cc = 250 páginas
- C = 391 páginas
- R = 343 páginas
- $|Cc \cap C| = 250$ páginas (pois $Cc \subset C$)
- $|R \cap C| = 337$ páginas

Assim o sistema atingiu índices de precisão e cobertura de 64% e 98% respectivamente. É importante ressaltar que estes índices foram obtidos a partir de um estudo de observação onde os participantes possuíam tarefas e limites de tempo que mantinham o foco da sua navegação na aquisição do conhecimento necessário. Poucas páginas irrelevantes foram propositalmente acessadas ao longo do estudo de observação. Por outro lado, durante o experimento alguns participantes lançaram mão do recurso de folksonomia oferecido pela ferramenta. Observamos que o número de classificações corretas aumentou significativamente à medida que este recurso era utilizado. Isso significa que com o passar do tempo a precisão poderia aumentar com a utilização da folksonomia.

6.1.4.6 Análise Qualitativa

Foi solicitado aos participantes que analisassem a qualidade das cadeias geradas pelo sistema a partir de suas próprias navegações. Para isso o participante deveria atribuir uma nota de 0 a 3 às cadeias geradas para cada tarefa atribuindo a cada nota o significado:

- 0 - não representou sua navegação;
- 1 - representou sua navegação com imprecisão;
- 2 - representou sua navegação;
- 3 - representou sua navegação com precisão.

As cadeias recomendadas aos participantes para auxiliá-los em suas tarefas também foram avaliadas com notas entre 0 e 3. Essas notas por sua vez representavam:

- 0 - não ajudou em nada na execução da tarefa;
- 1 - ajudou muito pouco na execução da tarefa;
- 2 - ajudou na execução da tarefa;
- 3 - ajudou muito na execução da tarefa

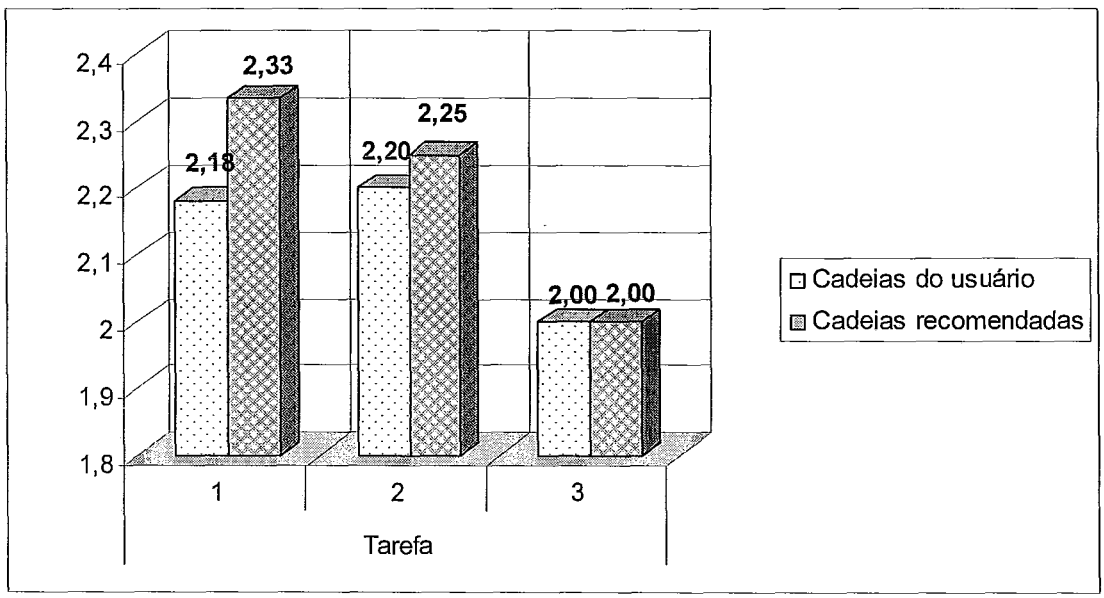


Figura 6.13 - Avaliação média das cadeias de conhecimento recomendadas

A Figura 6.13 nos mostra a satisfação dos participantes com as cadeias de conhecimento que receberam através das médias atribuídas. Como podemos observar, todas as cadeias de conhecimento criadas a partir de suas próprias navegações em todas as rodadas receberam avaliação média entre "representou sua navegação" e "representou sua navegação com precisão". Assim como todas as cadeias de conhecimento recomendadas foram avaliadas, na média, entre "ajudou na execução da tarefa" e "ajudou muito na execução da tarefa".

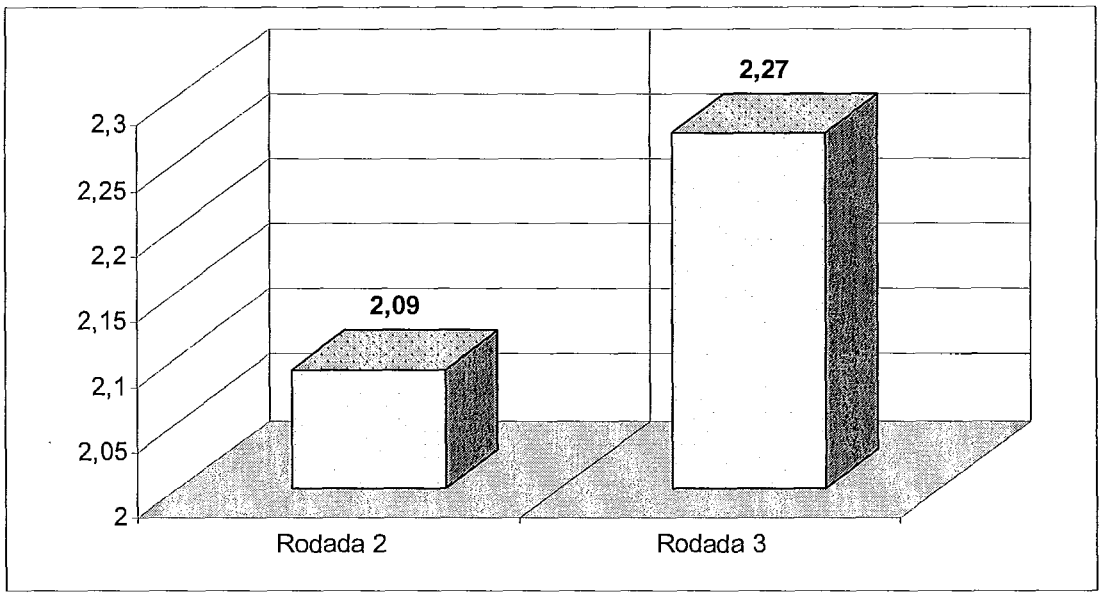


Figura 6.14 - Avaliação média das cadeias de conhecimento recomendadas por rodada

6.2 Estudo Experimental

Esse estudo experimental foi realizado com participantes distribuídos, cada um no seu próprio computador pessoal e com diferentes perfis profissionais e acadêmicos dentro da área de ciência da computação. Esse estudo não foi executado com a supervisão do experimentador.

6.2.1 Definição do Estudo Experimental

A definição desse estudo experimental é exatamente a mesma do estudo de observação realizado anteriormente. Sua realização será uma forma de comprovar se o mesmo comportamento será observado em um ambiente menos controlado e mais heterogêneo.

6.2.2 Planejamento do Estudo Experimental

6.2.2.1 Contexto

Os participantes serão divididos em dois grupos distintos. O grupo A receberá uma tarefa que descreverá um exercício de programação em Java um pouco mais complexo que os sugeridos no estudo anterior a esse. Ao receber suas tarefas, os participantes desse grupo deverão executar o exercício fazendo as pesquisas necessárias na Web utilizando o navegador Mozilla Firefox versão 2.0.0.3 ou superior (MOZILLA, 2007), com o *plug-in* Argus instalado. O sistema registrará toda a

navegação relacionada à execução da tarefa e contabilizará o tempo necessário e o número de páginas visitadas para a sua conclusão. Além disso, registraremos quantos usuários conseguiram executar a tarefa completamente.

Em seguida, o grupo B irá executar a mesma tarefa, também utilizando o Mozilla Firefox com o *plug-in* Argus, mas dessa vez os participantes contarão com a recomendação de cadeias de conhecimento criadas a partir da navegação dos participantes do grupo A.

6.2.2.2 Treinamento

O treinamento dos participantes será feito através de um manual que explicará detalhadamente como o *plug-in* Argus funciona e deixar bem claro quais passos eles devem seguir para executar o experimento. Antes de iniciarem o experimento será recomendado aos participantes que executem uma tarefa mais simples para se familiarizar com o *plug-in*. Essa navegação será desconsiderada para o estudo experimental.

6.2.2.3 Projeto Piloto

Antes da execução do estudo realizaremos um projeto piloto com uma estrutura idêntica a apresentada. Nesse piloto apenas dois participantes simularão o comportamento de cada grupo. Este piloto não visa extrair qualquer resultado dos participantes, ele será executado apenas para encontrar problemas no material planejado para o estudo, permitindo que este material seja aprimorado antes de sua utilização.

6.2.2.4 Participantes

Os participantes do estudo serão um conjunto de desenvolvedores e analistas de software. O estudo será executado em um ambiente da indústria de desenvolvimento de software. Os participantes serão pessoas com experiência profissional e executarão o estudo em seus ambientes de trabalho ou em suas casas utilizando o *plug-in* Argus e acessando remotamente o serviço Web Hera.

6.2.2.5 Instrumentação

Cada participante deverá atuar como desenvolvedor de um software. Esse software será composto de um uma interface gráfica que contará com uma etiqueta, onde será apresentado o texto "Arquivo" e um campo texto onde um suposto usuário

poderia digitar o caminho físico de um arquivo texto. Ao lado desse campo deverá existir um botão com o texto "Arquivo..." que permita que o suposto usuário abra uma janela gerenciadora de diretórios para buscar o arquivo texto e indicar o seu caminho no campo mencionado anteriormente.

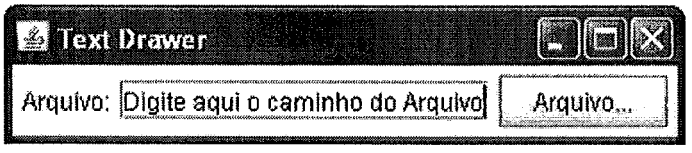


Figura 6.15 – Exemplo da tela principal da tarefa

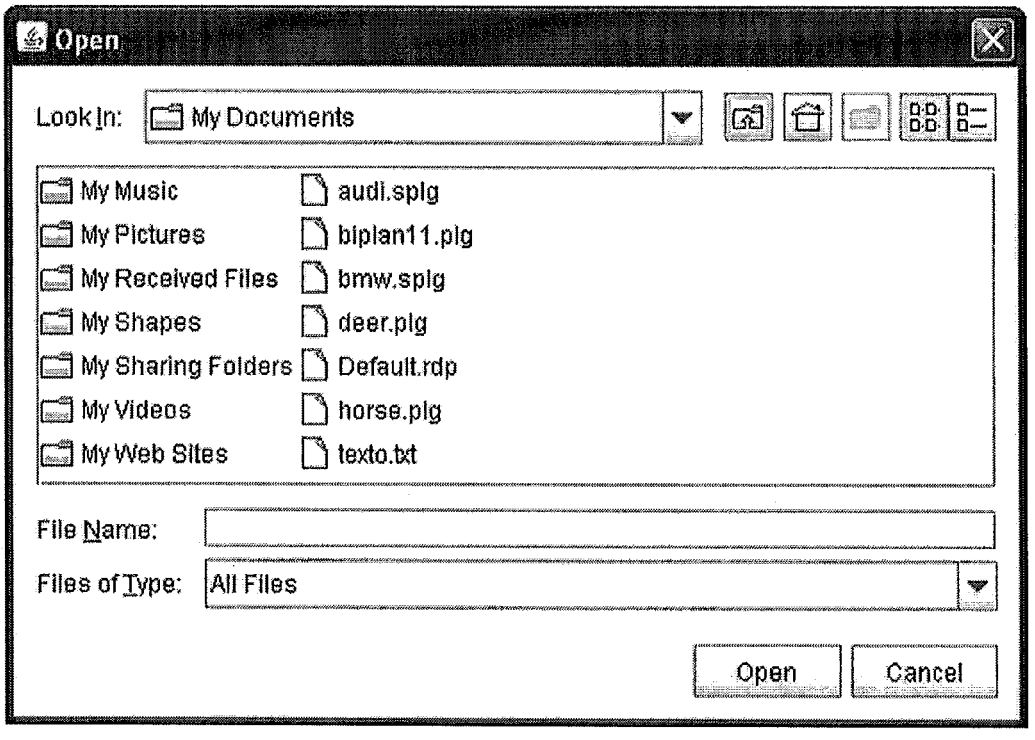


Figura 6.16 - Tela de navegação em diretórios

Uma vez que um arquivo texto for escolhido uma nova janela deve ser exibida apresentando o conteúdo desse arquivo exibido como um texto tridimensional. Essa janela deve ser semelhante à apresentada na Figura 6.17.

Figura 6.17 - Tela de apresentação de texto 3D

6.2.2.6 Critérios

Nesse estudo experimental usaremos os mesmos critérios de avaliação do estudo de observação realizado anteriormente. Reconhecemos anteriormente que o tempo e o número de páginas visitadas podem não ser os aspectos mais relevantes na pesquisa para realização das tarefas. Então, considerando que nesse estudo a tarefa é mais complexa e, portanto, exigirá mais tempo, acrescentamos um limite máximo de tempo para execução da tarefa e acrescentaremos um novo critério: o número de participantes que concluíram a tarefa.

6.2.2.7 Hipótese Nula

Assim como no estudo de observação executado anteriormente, a hipótese nula desse estudo se traduz na inexistência de diferenças significativas no tempo e no número de páginas navegadas para aquisição de conhecimento que permitissem a

conclusão do desenvolvimento de softwares por participantes utilizando o sistema em relação a desenvolvimento de softwares utilizando-se apenas a navegação tradicional pela Web; além de considerar que não há diferença significativa no número de participantes que terrão ajuda de cadeias de navegação e conseguirão concluir a tarefa dentro do limite de tempo estabelecido em relação àqueles e aqueles que não terão ajuda do sistema.

$$H_0 : \mu_{\#páginas\ \ sem\ \ sistema} = \mu_{\#páginas\ \ com\ \ sistema}$$
$$\wedge \mu_{tempo\ \ sem\ \ sistema} = \mu_{tempo\ \ com\ \ sistema}$$
$$\wedge \mu_{\#concluíram\ \ tarefa\ \ sem\ \ sistema} = \mu_{\#concluíram\ \ tarefa\ \ com\ \ sistema}$$

6.2.2.8 Hipótese Alternativa

No estudo atual, a hipótese alternativa determina que os participantes do estudo que utilizarem o sistema Olimpo terão resultados superiores aos participantes que utilizarem apenas a navegação na Web. De acordo com os critérios selecionados, esta hipótese se traduz em menor número de páginas navegadas ou tempo para conclusão das tarefas executadas por participantes utilizando as técnicas propostas em relação a tarefas executadas utilizando-se as a navegação convencional pela Web. Ou ainda, no aumento no número de usuários que conseguirão concluir a tarefa dentro do tempo estabelecido com a ajuda das cadeias de conhecimento em relação àqueles que não terão essa ajuda.

$$H_1 : \mu_{\#páginas\ \ sem\ \ sistema} > \mu_{\#páginas\ \ com\ \ sistema}$$
$$\vee \mu_{tempo\ \ sem\ \ sistema} > \mu_{tempo\ \ com\ \ sistema}$$
$$\vee \mu_{\#concluíram\ \ tarefa\ \ sem\ \ sistema} < \mu_{\#concluíram\ \ tarefa\ \ com\ \ sistema}$$

6.2.2.9 Variáveis Independentes

A principal variável independente do estudo é um indicador que determina se cada participante utilizou ou não o sistema Olimpo (escala nominal). A formação e a experiência dos participantes, medidas em uma escala nominal, também são informações independentes coletadas durante o estudo que poderão ser utilizadas durante a análise para a formação de blocos.

6.2.2.10 Variáveis Dependentes

As variáveis dependentes são o número de páginas visitadas, o tempo de desenvolvimento dos softwares e o número de participantes que concluirão a tarefa

antes do tempo estabelecido. O número de páginas será medido como inteiro, de acordo com o número de páginas que o participante visitar que forem selecionadas como relevante pelo sistema. O tempo de desenvolvimento será medido em segundos (escala inteiro), indicando o tempo necessário para a conclusão do software. O número de participantes que concluirão a tarefa antes do tempo estabelecido será medido como inteiro.

6.2.2.11 Análise Qualitativa

Com o objetivo de avaliar a satisfação dos participantes em relação às cadeias geradas pelo sistema e a qualidade do material utilizado no estudo. A análise qualitativa será realizada através de um questionário Q2. A satisfação dos participantes será avaliada por perguntas envolvendo a classificação de quanto à cadeia de conhecimento gerada representou a navegação do participante e quanto à cadeia recomendada ajudou na execução da tarefa.

6.2.2.12 Capacidade Aleatória

No estudo de observação executado anteriormente a distribuição dos objetos de análise foi feita de forma aleatória pelo sistema. No entanto, nesse estudo, um dos requisitos é garantir a semelhança dos grupos de teste em relação a suas distribuições de pessoas por experiência no objeto de estudo. Dessa forma, os participantes foram selecionados de forma aleatória dentro de um universo de candidatos a participantes, mas a distribuição dos objetos de estudo foi feita de acordo com as classificações em relação à experiência nesses objetos.

6.2.2.13 Classificação em Bloco

Os participantes serão divididos de acordo com a análise do questionário Q1. Esse questionário permitirá a que eles sejam classificados em diferentes níveis de experiência com a linguagem de programação Java e em relação às bibliotecas que serão usadas especificamente nesse estudo.

Uma vez classificados os participantes serão divididos em dois grupos que tenham a mesma distribuição de pessoas por nível de experiência. Dessa forma tentaremos garantir o máximo de semelhança entre os grupos.

6.2.2.14 Balanceamento

Durante a realização do estudo nos limitaremos a distribuir um número similar de participantes nos três grupos. Durante a análise, após a eliminação dos valores extremos, procuraremos um balanceamento, se este for possível.

6.2.2.15 Mecanismos de Análise

Os mecanismos de análise serão os mesmos utilizados no estudo de observação anterior, com exceção das métricas de precisão e completude adaptadas para aquele estudo. Essas métricas tornam-se inviáveis para esse caso uma vez que o número de páginas navegadas será muito grande para que um especialista o classifique manualmente.

6.2.2.16 Validade Interna do Estudo

Nesse estudo esperamos contar com pelo menos 18 participantes, o que garante um bom nível de validação interna do estudo. Outro ponto que pode influenciar o resultado do estudo é a troca de informações entre os participantes que já realizaram o estudo e os que não o realizaram. Nesse estudo os participantes não terão qualquer contato entre eles, já que executarão as tarefas remotamente.

6.2.2.17 Validade Externa do Estudo

Acreditamos que o maior problema em relação à validade externa do estudo é a falta de interesse dos participantes no próprio estudo. Alguns indivíduos podem realizar o estudo de forma desleixada, sem um interesse real no desenvolvimento das tarefas em menor tempo. A validade externa do estudo é considerada suficiente, visto que o presente estudo visa avaliar a viabilidade de aplicação do sistema de recomendação de cadeias de conhecimento. Demonstrada esta viabilidade, novos estudos podem ser planejados para refinar o universo de aplicação das técnicas.

6.2.2.18 Validade de Construção do Estudo

A validade de construção do estudo é similar à do estudo anterior e também não visa avaliar a correção das cadeias de conhecimento, mas a capacidade de utilização dessas cadeias no aprendizado. Assim, as cadeias serão compatíveis com as tarefas propostas aos participantes.

6.2.2.19 Validade de Conclusão do Estudo

Não encontramos grandes dificuldades em relação à capacidade de conclusão do estudo, visto que esta pode ser traçada a partir de um mecanismo de análise estatística amplamente utilizado, como o teste T (as escalas das variáveis dependentes e independente assim o permitem). Além disso, o estudo utiliza medidas objetivas, o que neutraliza a influência humana sobre os dados apurados e analisados.

6.2.3 Execução do Estudo experimental

6.2.3.1 Seleção dos Participantes

Para a presente execução do estudo, selecionamos os participantes dentre profissionais de tecnologia de informação que exercem diferentes tipos de atividades na área (desenvolvedores, analistas de sistema, analistas de processo, líderes de equipe, gerentes de projeto, pesquisadores, etc.), com diferentes graus de escolaridade, e com diferentes níveis de experiência nos objetos de estudo. Estes participantes atendem às restrições indicadas no planejamento, visto que todos são desenvolvedores de software participam do desenvolvimento de softwares.

Os convites foram feitos por e-mail para várias pessoas, que por sua vez convidaram várias pessoas. Os voluntários responderam o e-mail e receberam o material necessário para participar do estudo também por e-mail.

6.2.3.2 Instrumentação

Para esse estudo experimental foi desenvolvido um questionário Q1 em JSP (*JavaServer Pages*) (SUN, 2006) que ficou armazenado no mesmo servidor que o serviço Web Hera. O objetivo desse questionário era classificar a experiência dos participantes em relação à programação, à linguagem de programação Java e aos tópicos específicos de Java que seriam utilizados no estudo. Ele foi baseado no questionário utilizado no estudo comparativo realizado por CAMARGO *et al.* (2006) que tinha o mesmo objetivo.

O questionário Q1 continha as seguintes perguntas:

1) Qual o seu grau de escolaridade?
   a) Ensino Médio.
   b) Graduação incompleta.
   c) Graduação completa.

d) Mestrado incompleto.

e) Mestrado completo.

f) Doutorado incompleto.

g) Doutorado completo.

2) Qual é a alternativa que melhor caracteriza a sua experiência com desenvolvimento de software na prática?

a) Eu desenvolvi software sozinho.

b) Eu desenvolvi software como parte de uma equipe como parte de um curso.

c) Eu desenvolvi software como parte de uma equipe em uma empresa por um período de até cinco anos.

d) Eu desenvolvi software como parte de uma equipe em uma empresa por mais de cinco anos.

3) Quão bem você conhece os conceitos de Orientação a Objetos?

a) Especializado – "Emprego e praticamente nunca consulto a bibliografia. Em geral sou referência para esclarecer dúvidas de outras pessoas".

b) Avançado – "Emprego com freqüência e raramente preciso consultar a bibliografia para esclarecer alguma dúvida".

c) Médio – "Já empreguei os conceitos de Orientação a Objeto, mas ainda preciso consultar a bibliografia para esclarecer algumas dúvidas".

d) Básico – "Conheço algum dos princípios de Orientação a Objeto, mas nunca empreguei".

e) Nada – "Não sei o que é Orientação a Objeto".

4) Quão bem você conhece a linguagem de programação Java?

a) Especializado – "Já trabalhei e conheço as plataformas e *frameworks* Java muito bem. Estou sempre a par das últimas novidades em Java. Sou uma referência para outras pessoas".

b) Avançado – "Conheço e já trabalhei com algumas plataformas e *frameworks* Java".

c) Médio – "Já desenvolvi programas em Java usando teorias mais complexas de Orientação a Objeto como interface, sobreposição ou sobrecarga".

d) Básico – "Já desenvolvi programas muito simples em Java, a maioria para cursos ou cadeiras da faculdade".

e) Nada – "Nunca escrevi uma linha de código em Java".

5) Quão bem você conhece as bibliotecas de criação de interface cliente-servidor em Java?
   a) Especializado – "Conheço muito bem pelo menos uma dessas bibliotecas e praticamente nunca preciso consultar a bibliografia para esclarecer dúvidas. Em geral sou uma referência para outras pessoas".
   b) Avançado – "Já desenvolvi grandes sistemas com estrutura cliente-servidor e conheço muito bem pelo menos uma dessas bibliotecas, mas eventualmente consulto a bibliografia para esclarecer algumas dúvidas".
   c) Médio – "Já desenvolvi pequenos programas com algumas interfaces que se relacionavam entre si".
   d) Básico – "Conheço e já desenvolvi programas com uma única interface com o usuário em pelo menos uma dessas bibliotecas".
   e) Nada – "Não conheço nenhuma biblioteca de criação de interface cliente-servidor em Java, ou já ouvi falar, mas nunca usei".
6) Quão bem você conhece as bibliotecas de leitura e escrita em arquivos em Java?
   a) Especializado – "Conheço muito bem pelo menos uma dessas bibliotecas e praticamente nunca preciso consultar a bibliografia para esclarecer dúvidas. Em geral sou uma referência para outras pessoas".
   b) Avançado – "Já desenvolvi sistemas com leitura e escrita em arquivos em Java e conheço muito bem pelo menos uma dessas bibliotecas, mas eventualmente consulto a bibliografia para esclarecer algumas dúvidas".
   c) Médio – "Já desenvolvi pequenos programas com algumas rotinas de leitura e escrita em arquivos em Java".
   d) Básico – "Conheço e já desenvolvi programas com leitura e escrita em arquivos em Java em pelo menos uma dessas bibliotecas".
   e) Nada – "Não conheço nenhuma biblioteca leitura e escrita em arquivos em Java, ou já ouvi falar, mas nunca usei".
7) Quão bem você conhece as bibliotecas de desenhos 3D em Java?
   a) Especializado – "Conheço muito bem pelo menos uma dessas bibliotecas e praticamente nunca preciso consultar a bibliografia para esclarecer dúvidas. Em geral sou uma referência para outras pessoas".
   b) Avançado – "Já desenvolvi sistemas com desenhos 3D em Java e conheço muito bem pelo menos uma dessas bibliotecas, mas eventualmente consulto a bibliografia para esclarecer algumas dúvidas".

c) Médio – "Já desenvolvi pequenos programas com alguns desenhos 3D em Java".

d) Básico – "Conheço e já desenvolvi programas com desenhos 3D em Java em pelo menos uma dessas bibliotecas".

e) Nada – "Não conheço nenhuma desenhos 3D em Java, ou já ouvi falar, mas nunca usei".

8) Como você dividiria sua experiência em criação de software?

a) Esforço individual (sozinho): _____ %

b) Esforço cooperativo com outros (equipe): _____ %

c) Supervisão de outros programadores (gerenciamento): _____ %

9) Preencha a tabela abaixo com o número de meses que você teve experiência acadêmica ou profissional nos tópicos listados.

Tabela 6.8 – Meses de experiência dos participantes por assunto.

Quantos meses de experiência você tem...	Acadêmica	Profissional
em gerenciamento de projetos de Software?		
em análise de software?		
em desenvolvimento de software?		
em teste de software?		
utilizando conceitos de Orientação a Objetos?		
utilizando linguagem de programação Java?		
utilizando as bibliotecas de criação de interface cliente-servidor em Java?		
utilizando as bibliotecas de leitura e escrita em arquivos em Java?		
utilizando as bibliotecas de desenhos 3D em Java?		

10) Quantos programas você já escreveu sozinho de acordo com os tamanhos e linguagens dados a seguir? Preencha na tabela abaixo a quantidade de programas de acordo com o seu tamanho.

Tabela 6.9 - Quantidade de programas escritos pelo participante.

Linguagem	Número de linhas de código do programa

de Programação	Menos de 1000	Entre 1000 e 5000	Entre 5000 e 10000	Entre 10000 e 15000	Mais de 15000
C++					
Java					
Delphi					
Outra:_____					

11) Qual o seu nível de proficiência para leitura em língua inglesa?

a) Básico
b) Médio
c) Avançado
d) Especializado

Assim que todos os voluntários responderam o questionário Q1 eles foram divididos em dois grupos de forma que ambos tivessem uma distribuição homogênea de pessoas com experiências semelhantes em linguagem de programação Java.

Os grupos receberam um instalador do *plug-in* Argus, um pacote de arquivos para execução da tarefa, e um roteiro de como executar o experimento. Esse roteiro, além de explicar cada passo que o participante deveria executar, especificava detalhadamente como executar a tarefa descrita na seção 6.2.2.5. Essa especificação não mencionou nada relativo às bibliotecas necessárias para sua execução.

O primeiro grupo executou o estudo primeiro e os recursos do *plug-in* Argus que solicitavam a recomendação de cadeias de conhecimento para o agente Hermes foram desabilitados. O papel do Argus nesse caso foi apenas de armazenar as informações da navegação.

O segundo grupo executou o estudo imediatamente após o último participante do primeiro grupo ter finalizado sua tarefa. No entanto, os participantes desse grupo tiveram a oportunidade de utilizar os recursos de recomendação de cadeias, que foram montadas a partir da navegação do primeiro grupo.

No fim da execução de sua tarefa os participantes do segundo grupo responderam um questionário Q2 com perguntas que permitiram avaliar as cadeias criadas. Essas perguntas foram:

1) A cadeia de conhecimento que o Sistema Olimpo criou a partir dos sites que você navegou:

a) Não representou sua navegação.
b) Representou sua navegação com imprecisão.
c) Representou sua navegação com precisão razoável.
d) Representou sua navegação com boa precisão.
e) Representou sua navegação com exatidão.

2) A cadeia de conhecimento que o Sistema Olimpo recomendou para a execução da tarefa:
   a) Atrapalhou muito a execução da tarefa.
   b) Atrapalhou a execução da tarefa.
   c) Não foi relevante para a execução da tarefa.
   d) Ajudou na execução da tarefa
   e) Ajudou muito na execução da tarefa

6.2.3.3 Procedimento de Participação

Existem dois procedimentos distintos de participação no estudo. O primeiro foi seguido pelos participantes que não utilizaram o Olimpo para recomendação de cadeias de conhecimento, enquanto o segundo procedimento foi utilizado pelos participantes que utilizaram a recomendação de cadeias.

Procedimento de participação **sem a recomendação de cadeias**:

1. O participante se dispõe a participar do estudo experimental.
2. O participante recebe o endereço Web do questionário Q1 para acessá-lo e preenchê-lo.
3. Os experimentadores definem a que grupo o participante fará parte.
4. O participante recebe um roteiro do estudo.
5. O *plug-in* Argus associa um número ao participante.
6. O participante ativa o registro de páginas do Argus e inicia sua pesquisa para conclusão da tarefa.
7. O participante executa a tarefa enquanto pesquisa na Web informações para a execução.
8. O participante conclui a tarefa e desativa o registro de páginas do Argus.
9. O Argus apresenta ao participante a cadeia de conhecimento gerada a partir da sua navegação.
10. O participante envia o programa criado para o responsável pelo estudo.

Procedimento de participação **utilizando a recomendação de cadeias**:

1. O participante se dispõe a participar do estudo experimental.
2. O participante recebe o endereço Web do questionário Q1 para acessá-lo e preenchê-lo.
3. Os experimentadores definem a que grupo o participante fará parte.
4. O participante recebe um roteiro do estudo.
5. O *plug-in* Argus associa um número ao participante.
6. O participante solicita a recomendação de uma cadeia de conhecimento sobre a tarefa que está sendo executada.
7. O Argus recomenda uma cadeia retornada pelo módulo Hermes.
8. O participante ativa o registro de páginas do Argus e inicia sua pesquisa para conclusão da tarefa.
9. O participante executa a tarefa enquanto pesquisa na Web informações para a execução.
10. O participante conclui a tarefa e desativa o registro de páginas do Argus.
11. O Argus apresenta ao participante a cadeia de conhecimento gerada a partir da sua navegação.
12. O Argus apresenta o questionário Q2 para que o participante preencha.
13. O participante preenche o formulário e envia o programa criado para o responsável pelo estudo.

6.2.3.4 Execução

O estudo experimental do sistema Olimpo contou com a participação de dezoito profissionais de informática, todos com formação acadêmica mínima com o ensino superior incompleto. Todos os participantes atuam no mercado de trabalho de tecnologia da informação no presente momento. O estudo aqui descrito foi modelado para durar duas horas, pois consideramos ser esse um tempo médio razoável para a conclusão da tarefa proposta.

Nesse estudo cada voluntário respondeu o questionário Q1, apresentado na seção 6.2.3.2. Uma vez que todos os questionários foram respondidos e analisados os voluntários foram divididos em dois grupos distintos, de forma que ambos possuíssem uma distribuição similar de membros em relação aos seus graus de experiência em

linguagem de programação Java e nas bibliotecas necessárias para a execução do experimento.

A Tabela 6.10 mostra o número de participantes distribuídos em cada grupo de acordo com seus graus de escolaridade e com suas experiências práticas em desenvolvimento de software.

Tabela 6.10 – Numero de participantes X Tema X Grupo.

Tema	Categoria	Nº. Participantes	
		Grupo 1	Grupo 2
Escolaridade	Graduação incompleta	0	1
	Graduação completa	1	0
	Mestrado incompleto	5	3
	Mestrado completo	3	4
	Doutorado incompleto	0	1
Experiência prática no desenvolvimento de software	Desenvolveu software como parte de uma equipe em uma empresa por um período de até cinco anos.	8	3
	Desenvolvi software como parte de uma equipe em uma empresa por mais de cinco anos.	1	6

O número de pessoas distribuído por categoria de experiência em orientação a objeto, linguagem de programação Java e nas bibliotecas necessárias para a execução da tarefa é apresentada na Tabela 6.11.

Tabela 6.11 – Numero de participantes X Experiência X Grupo.

Experiência	Categoria	Nº. Participantes	
		Grupo 1	Grupo 2
Inglês	Média	1	0
	Avançada	7	9
	Especializada	1	0
Orientação a objeto em geral	Média	5	5
	Avançada	2	2

	Especializada	2	2
Linguagem de programação Java em geral	Básica	2	3
	Média	2	2
	Avançada	5	3
Bibliotecas para Criação de Interface Cliente-Servidor em Java	Especializada	0	1
	Nenhuma	4	1
	Básica	0	3
	Media	3	4
	Avançada	2	2
Bibliotecas para leitura e escrita em arquivos em Java	Nenhuma	2	0
	Básica	3	1
	Média	3	4
	Avançada	1	3
Bibliotecas para criação de imagens 3D em Java	Nada	9	9

A Figura 6.18 representa as médias das distribuições da experiências dos grupos entre desenvolvimento individual, desenvolvimento em equipe e gerenciamento de equipes de desenvolvimento.

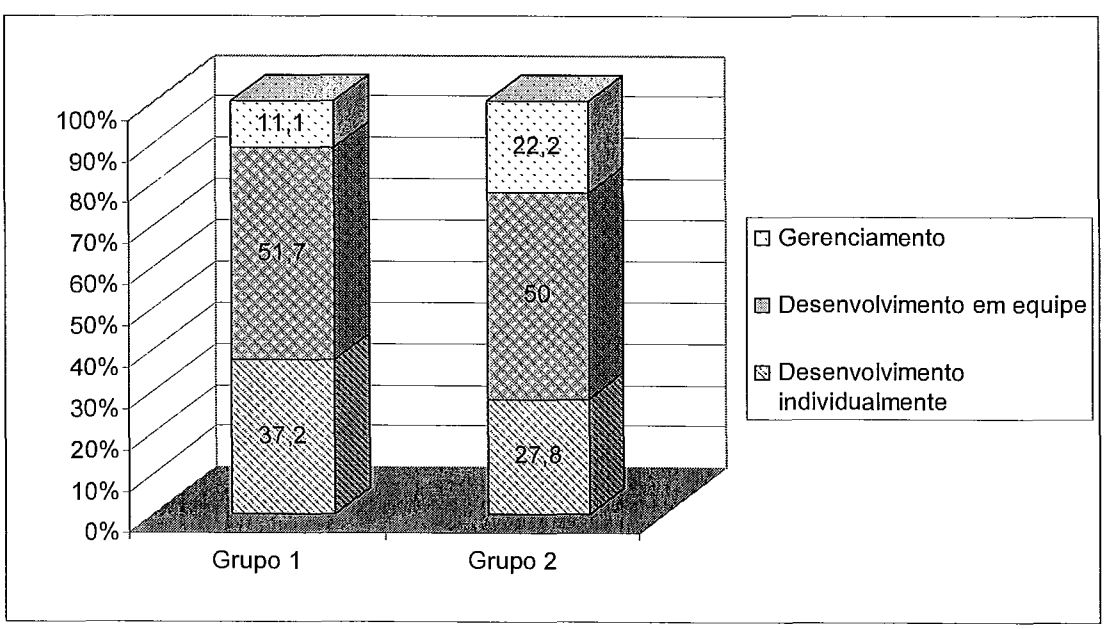


Figura 6.18 – Médias das distribuições da experiência dos participantes

A média do número de meses que os participantes do estudo experimental tiveram de experiência com os assuntos propostos pelo questionário Q1 são apresentadas na Tabela 6.12.

Tabela 6.12 – Médias dos meses de experiência dos participantes por assunto.

Experiência	Acadêmica		Profissional	
	Grupo 1	Grupo 2	Grupo 1	Grupo 2
Gerenciamento de projetos de Software	0,78	2,44	10,67	13,44
Análise de software	7,11	9,33	29,00	24,11
Desenvolvimento de software	57,33	40,00	57,33	50,89
Teste de software	5,78	2,44	7,56	8,89
Utilizando conceitos de Orientação a Objetos	12,44	26,27	27,89	27,44
Utilizando linguagem de programação Java	25,00	23,44	18,33	18,67
Utilizando as bibliotecas de criação de interface cliente-servidor em Java	6,33	9,22	8,44	6,67
Utilizando as bibliotecas de leitura e escrita em arquivos em Java	4,78	11,33	4,11	9,78
Utilizando as bibliotecas de desenhos 3D em Java	0,00	0,00	0,00	0,00

A Tabela 6.13 mostra o número médio de programas criados pelos participantes em relação com seus tamanhos medidos em quantidade de linhas de código.

Tabela 6.13 – Número médio de programas desenvolvidos X Tamanho aproximado dos programas em número de linhas de código

Tamanho dos Programas	Grupo 1	Grupo 2
Menos de 1000 linhas	22,22	28,78
Entre 1000 e 5000 linhas	5,56	11,56
Entre 5000 e 10000 linhas	1,89	7,22

Entre 10000 e 15000 linhas	0,22	1,11
Mais de 15000 linhas	0,44	2,22

A partir dessas informações, os participantes, foram considerados aptos a representarem profissionais da área por mostrarem qualificação razoável.

6.2.4 Análise dos Resultados do Estudo de Observação

6.2.4.1 Avaliação Quantitativa

Tabela 6.14 – Resultados obtidos no estudo experimental

Grupo	ID	Tempo (s)	Nº. de páginas
Grupo 1	1	8416	21
	2	2275	55
	4	12368	107
	6	6618	82
	7	17445	143
	8	4259	73
	12	10981	170
	13	3880	37
	14	5995	99
Grupo 2	3	8366	86
	5	2373	20
	9	3104	13
	10	3316	35
	11	6903	91
	15	4951	48
	16	5281	76
	17	4356	26
	18	4913	51

A análise quantitativa do estudo experimental pode ser separada em análise de número de páginas visitadas e análise de tempo. Cada uma destas análises foi

realizada em duas etapas: eliminação de valores extremos e teste paramétrico de média. Os resultados obtidos no estudo experimental são apresentados na Tabela 6.14.

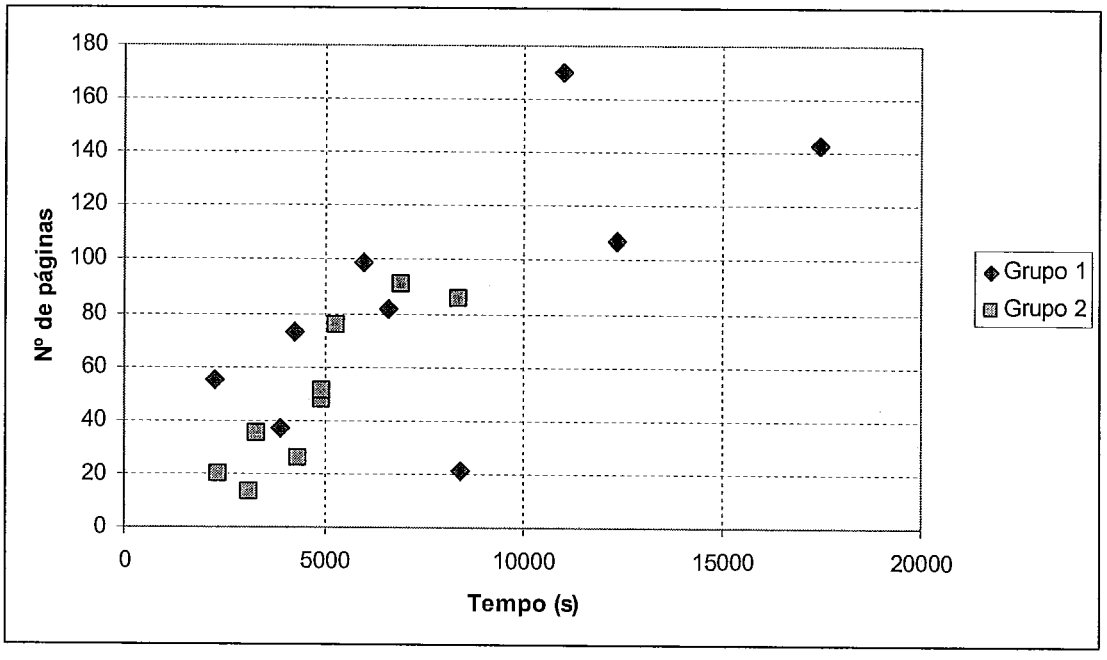


Figura 6.19 - Resultados obtidos no estudo experimental

6.2.4.2 Eliminação de Valores Extremos

Tabela 6.15 - Resultados obtidos no estudo experimental com corte T

Grupo	ID	Tempo (s)	Nº. de páginas
Grupo 1	1	8416	~~21~~
	2	~~2275~~	55
	4	12368	107
	6	6618	82
	7	~~17445~~	~~143~~
	8	4259	73
	12	10981	~~170~~
	13	3880	~~37~~
Grupo 2	14	5995	99
	3	~~8366~~	~~86~~
	5	~~2373~~	~~20~~
	9	3104	~~13~~

	10	3316	35
	11	~~6903~~	~~91~~
	15	4951	48
	16	5281	76
	17	4356	26
	18	4913	51

Tendo coletado as informações de tempo e número de páginas navegadas através do módulo Argus, passamos para a etapa de eliminação de valores extremos. Nesta etapa, aplicamos um corte baseado na distribuição T em 99% (FREUND, PERLES, 1998).

Na análise de tempo de conclusão da tarefa, o corte T eliminou dois participantes do grupo que não utilizou as cadeias de conhecimento recomendadas e três participantes do grupo que utilizou as cadeias criadas. Na análise do número de páginas visitadas, o corte T eliminou quatro participantes do primeiro grupo e quatro participantes do segundo grupo. A Tabela 6.15 apresenta os resultados do estudo de observação, grifando os valores eliminados pelo corte T. A Tabela 6.16 apresenta estatísticas descritivas sobre os resultados do estudo após a eliminação dos valores extremos.

Tabela 6.16 - Estatísticas descritivas sobre os resultados do estudo (após a eliminação de valores extremos)

Estatística		**Nº. de páginas**	**Tempo (s)**
Grupo 1	Média	83,20	7502,43
	Mediana	82	6618
	Mínimo	55	3880
	Máximo	107	12368
	Desvio padrão	20,72	3248,21
Grupo 2	Média	47,20	4320,17
	Mediana	48	4634,50
	Mínimo	26	3104
	Máximo	76	5281

	Desvio padrão	18,99	912,28

6.2.4.3 Teste Paramétrico

Os dados que permaneceram após a eliminação de valores extremos foram submetidos a uma análise paramétrica baseada na distribuição T (WOHLIN *et al.*, 2000). Na análise de tempo, o teste T foi conclusivo no sentido de afirmar que o tempo médio que os participantes que utilizaram a cadeias de conhecimento de outros participantes levaram para concluir as tarefas propostas foi inferior ao tempo médio que os participantes que não utilizaram cadeias de conhecimento levaram para concluir as tarefas. Os resultados intermediários do teste T para o critério tempo são apresentados na Tabela 6.17.

Tabela 6.17 - Resultados intermediários para o Teste T para análise do tempo (90%)

Desvio	T_0	Liberdade	Distribuição T	Resultado ($T_0 < -T$)
3898,86	-1,90	5	2,01	$\mu_{\text{tempo s/ sistema}} > \mu_{\text{tempo c/ sistema}}$

A Tabela 6.18 nos mostra que uma análise semelhante à realizada para o tempo pode ser aplicada ao número de páginas visitadas.

Tabela 6.18 - Resultados intermediários para o Teste T para análise do número de páginas visitadas (90%)

Desvio	T_0	Liberdade	Distribuição T	Resultado ($T_0 < -T$)
13	-5,33	2	2,92	$\mu_{\text{\# páginas s/ sistema}} > \mu_{\text{\# páginas c/ sistema}}$

6.2.4.4 Análise da Execução das Tarefas

Outra métrica utilizada nesse estudo experimental é o número de participantes capazes de concluir a tarefa dentro de um determinado espaço de tempo. Esse estudo experimental foi concebido para que os participantes levassem no máximo duas horas para concluí-lo, ou seja, 7200 segundos. De acordo com os dados mostrados na Tabela 6.14, observamos que cinco dos nove participantes do primeiro grupo conseguiram concluir a tarefa dentro do tempo proposto. Enquanto isso, oito dos nove participantes do segundo grupo foram capazes de concluir a tarefa.

6.3 Conclusão

Neste capítulo foram apresentados os estudos de observação e experimental ao qual o protótipo do Sistema Olimpo foi submetido, detalhando a metodologia que dividiu os participantes em grupos diferentes e permutou tarefas entre eles compartilhando suas experiências.

Embora os testes estatísticos relacionados com o uso das cadeias recomendadas pelo especialista sejam inconclusivos, o sucesso dos testes relacionados com a utilização de alguma cadeia para conclusão das tarefas nos leva a concluir que a geração automática e recomendação de cadeias de conhecimento são viáveis e auxiliam um aprendiz na pesquisa e aquisição de conhecimento.

A análise dos dados obtidos no estudo de observação mostrou que o Olimpo pode agilizar a aquisição de conhecimento para a solução de problemas imediatos, filtrando conteúdo na Internet e selecionando páginas visitadas por outros membros. Isso agregaria valor imediato a uma comunidade de prática, uma vez que seus membros poderiam solucionar seus problemas com mais eficiência.

Por outro lado, o comportamento dos participantes desse estudo demonstrando interesse em adquirir mais conhecimento quando este é disponibilizado pela ferramenta mostra que ela pode também ser uma importante contribuição para facilitar o acesso ao conhecimento. Dessa forma estaria auxiliando a comunidade a suprir alguns dos fatores para o seu sucesso, segundo Mcdermott (2000). Em longo prazo, os membros poderiam adquirir conhecimento com mais facilidade de acesso à informação e de forma mais bem estruturada, uma vez que a cadeia de conhecimento indicaria em que ordem estudar e onde encontrar a informação.

O estudo experimental executado posteriormente simulou a utilização do sistema em um ambiente mais próximo do real, permitindo que as pessoas o utilizasse de forma distribuída em seus próprios ambientes de trabalho. Esse estudo confirmou o resultado obtido no ambiente mais controlado criado para o estudo de observação.

Capítulo 7 - Conclusão

Mcdermott (2002) e Wenger *et al.* (2002) apresentaram fatores críticos para o sucesso de uma comunidade de prática. Esses valores são extensíveis a comunidades de aprendizado, já que essas são uma especialização da anterior. Este trabalho está voltado para complementar o aprendizado dos membros dessa comunidade. Com isso, ele pode assumir um papel importante para ajudar a garantir alguns dos fatores de sucesso, uma vez que permite que os membros das comunidades compartilhem algum conhecimento com o mínimo de esforço, além de auxiliar na criação de um relacionamento interpessoal e estimular iniciantes a obterem informação e experiência com membros mais experientes.

Parte do objetivo do sistema aqui proposto é, portanto, semi-automatizar a colaboração oportunística entre os membros de uma comunidade de aprendizado, permitindo que essa aconteça não só na sua informalidade, mas também de forma mais fácil, transparente e automatizada possível. Ele tenta, de uma maneira simplificada, determinar como cada membro da comunidade se comportou para adquirir conhecimento na Internet. Para isso, ele é dividido em vários módulos de softwares, cada um com a responsabilidade de executar um subconjunto de todas as tarefas necessárias para atingir seu objetivo.

A identificação e análise desse comportamento não seriam possíveis sem a utilização das técnicas de mineração apresentadas nessa dissertação e empregadas na ferramenta implementada. Os dados brutos são extraídos durante a navegação do usuário, mas é através da aplicação dessas técnicas na seqüência de páginas navegadas, bem como no conteúdo de cada uma, que é possível lapidar e analisar as informações escondidas nos dados.

Essa lapidação inicia-se com três etapas básicas: a preparação dos dados, que filtra e limpa os dados coletados, separando as visitas relevantes das não relevantes; a descoberta dos padrões de utilização, que minera os dados em busca de padrões de comportamento dos usuários; e a análise dos padrões encontrados é a etapa onde o usuário final do sistema minerador tem a possibilidade de visualizar e tirar conclusões a partir dos padrões encontrados. Essas técnicas de mineração de utilização são empregadas nas seqüências de URLs navegadas pelo usuário.

A análise é finalizada com a classificação do conteúdo das páginas através de uma adaptação da técnica de mineração de conteúdo proposta por Larocca Neto *et al.* (2000) e conhecida como Sumarização Automática. Durante este processo, o texto de cada página é resumido apenas aos termos com maior relevância. Assim, se uma palavra pode ser significativa em um texto por sua freqüência, ela também pode indicar a importância relativa das sentenças que a contêm. Neste caso específico, o conteúdo das páginas é descrito em meio a etiquetas da linguagem de marcação HTML. Essa marcação, por si só, já é um indicativo importante do grau de importância dos termos encontrados no texto.

Neste trabalho, os dois processos de mineração foram usados de forma complementar para encontrar padrões de navegação entre os assuntos das páginas e não entre cada URL especificamente. A mineração de utilização foi utilizada para descobrir os padrões de navegação através das páginas e a mineração de conteúdo foi utilizada para classificá-las, definindo o seu assunto principal.

Para aprimorar a qualidade do resultado da classificação das páginas navegadas pelos usuários, propomos a utilização de uma técnica de classificação pelos próprios membros da comunidade conhecida como folksonomia. Quanto mais páginas forem classificadas pelos usuários, maior a porcentagem de acerto na classificação automática de páginas que não foram previamente classificadas. Essa técnica, no entanto, só é empregada com segurança porque é utilizada juntamente com as técnicas de mineração e de ontologia apresentadas nessa dissertação.

Após a apresentação dos objetivos do trabalho e das técnicas utilizadas nele, descrevemos a arquitetura modularizada proposta para atingir tais objetivos através das técnicas de mineração. Essa arquitetura foi batizada de Olimpo, uma vez que cada módulo que a compõe recebeu o nome de uma entidade mitológica grega mencionada no conto de Ovidius (14). Essa arquitetura permite a classificação das páginas e a criação de uma cadeia de conhecimento representada por um grafo orientado de estruturas que carregam não só o conceito estudado pelo usuário, mas também as páginas que ele visitou para obtê-lo. Essa arquitetura foi implementada em um protótipo do sistema Olimpo que foi posteriormente submetido a um estudo experimental.

Com o desenvolvimento de um protótipo, tornou-se imediatamente necessário o desenvolvimento de uma ontologia que o apoiasse, já que seu funcionamento depende de uma ontologia durante o processo de classificação empregado pelo

módulo Hermes. Essa ontologia foi desenvolvida para descrever o domínio do paradigma de orientação a objeto e, em seguida, instanciada para a linguagem Java. No entanto, é importante que fique claro que essa ontologia pode ser instanciada para diferentes assuntos.

No processo de construção dessa ontologia, seguimos os passos propostos por Noy & Mcguinness (2001) e criamos uma ontologia de domínio (mais genérica) para só depois especializá-la em uma ontologia de aplicação, de acordo com as recomendações de Guarino (1998). Observamos que, se seguidas essas recomendações, de fato a ontologia, além de poder ser facilmente revertida a uma ontologia de domínio, se necessário, é facilmente extensível a outros domínios.

Observamos também que, como subproduto da utilização de ontologia no processo de classificação, este trabalho contempla as duas camadas mais baixas de uma ontologia de aprendizado colaborativo (SUPNITHI, 1999), uma vez que captura o processo de aprendizado pessoal do aprendiz, criando subsídio pra a camada mais baixa; e permite que os vários processos de aprendizado pessoal sejam trocados e fornece as fontes de consultas, criando as informações necessárias para a camada mais alta de negociação.

Com protótipo da ferramenta e ontologia implementados, ambos puderam ser finalmente submetidos a um estudo experimental. Para isso, onze voluntários foram divididos em três grupos distintos e trocaram tarefas (as tarefas envolviam o desenvolvimento de classes simples na linguagem de programação Java) em três rodadas. Na primeira rodada, cada grupo recebeu uma tarefa e a executou com auxílio apenas de suas próprias pesquisas na Internet. Na segunda rodada, os participantes trocaram tarefas e as cadeias de conhecimento geradas pelo sistema. Dessa vez, os participantes poderiam contar com a ajuda dessas cadeias além de sua própria pesquisa. Na terceira e última rodada, as participantes trocaram novamente suas tarefas e dessa vez receberam uma cadeia de conhecimento estruturada, criada através da navegação de um especialista e com as correções feitas por ele.

As análises dos dados obtidos neste estudo experimental mostraram que, comparando tempo e número de páginas visitadas para conclusão de cada tarefa diminuíam drasticamente na segunda rodada, quando os participantes recebiam ajuda das cadeias. Conforme esperado, isso mostra que o Olimpo pode agilizar a aquisição de conhecimento para a solução de problemas imediatos, filtrando conteúdo na Internet e selecionando páginas visitadas por outros membros. Isso agrega valor

imediato a uma comunidade de prática, uma vez que seus membros poderiam solucionar seus problemas com mais eficiência.

No entanto, o número de páginas visitadas e o tempo dispensado a execução das tarefas elevou-se novamente na terceira rodada do estudo de caso, o que não estava dentro das expectativas. Uma análise mais profunda mostrou que, de posse de uma cadeia mais completa e bem estruturada, os participantes do estudo de caso tiveram seus interesses em adquirir mais conhecimento estimulado. Isso aumentou o tempo de pesquisa, o número de páginas visitadas e os seus conhecimentos no assunto. Essa análise mostrou que o Olimpo pode também ser uma importante contribuição para facilitar o acesso ao conhecimento. Dessa forma, acreditamos que estamos auxiliando a comunidade a suprir alguns dos fatores para o seu sucesso, segundo Mcdermott (2000).

7.1 Trabalhos Futuros

A flexibilidade da arquitetura modularizada aliada à ontologia nos permite vislumbrar várias possibilidades de alterações e adaptações que podem gerar novas contribuições. Muitas já estão descritas ao longo dessa dissertação.

Direcionando nossas atenções para o uso da ontologia, poderíamos incluir no sistema um módulo responsável por identificar ocorrências freqüentes de navegação entre termos que não estão relacionados na ontologia. Com esse tipo de análise, novos relacionamentos poderiam ser sugeridos e negociados pela comunidade com apoio do sistema. Com essa funcionalidade desenvolvida, seria necessário pouco esforço para evoluir a utilização de folksonomia para que essa permitisse a sugestão (e conseqüente negociação) de novos conceitos para a ontologia por parte dos usuários.

Em relação aos membros da comunidade de prática, a ferramenta mostrou-se bastante proativa no que diz respeito à captura e análise das informações. No entanto, a mesma proatividade é desejada no momento em que essa informação é oferecida a outros membros. A proposta contida nessa dissertação é verificar quando uma cadeia que está sendo criada por um usuário é subconjunto de uma cadeia maior que pertença a outro usuário. Quando isso ocorrer, a cadeia mais completa pode ser recomendada ao aprendiz de forma proativa.

A microarquitetura proposta para o módulo Hermes não foi totalmente desenvolvida da forma proposta, no entanto poderia trazer vantagens de desempenho e flexibilizá-lo para inclusão de novas funcionalidades.

A proposta desta dissertação é apenas uma instância do que pode ser realizado com o potencial das idéias propostas pela terceira geração da Internet. Os princípios da *Web* semântica poderão ser utilizados nas mais variadas áreas de aplicação. Munidos de contextualização de informação e uma arquitetura de módulos responsáveis por realizar o trabalho árduo, os membros de comunidades de prática e usuários da Internet ficarão mais livres para trocar e procurar por conhecimento de alto nível que somente humanos sabem trocar, e que talvez, nenhum sistema seja capaz de interpretá-lo corretamente.

Referências Bibliográficas

ANALOG, 2007, "Analog: WWW logfile analysis", URL: http://www.analog.cx, visitado em 13/01/2007.

APACHE, 2006, "Web Services – Axis", Apache Software Foundation, URL: http://ws.apache.org/axis/, visitado em 05/04/2007.

APACHE, 2007, "Jakarta Commons HttpClient", Apache Software Foundation, URL: http://jakarta.apache.org/commons/httpclient/, visitado em 03/06/2007.

APPLE Inc., 2007, "Apple – Mac OS X – Leopard Sneak Peek", URL: http://www.apple.com/macosx/leopard/, visitado em 05/04/2007.

AZEVEDO, H., SCALABRIN, E. A., 2003, "Human Collaborative Online Environment Using Intelligent Agents". Não publicado.

BARCLAY, R., MURRAY, P., 1997, "What is Knowledge Management". Disponível em: http://www.media-access.com/whatis.html, Acessado em 06/10/2006.

BEER S., 2007, "IE7 being caught by Firefox despite 100 million installations", IT wire, 18 de janeiro, URL: http://www.itwire.com.au/content/view/8664/53/, visitado em 05/04/2007.

BERNERS-LEE, T., HENDLER, J., LASSILA, O., 2001, "The Semantic Web", Scientific American, Maio.

BOURDREAU, A., COUILLARD G., 1999, "Systems Integration and Knowledge Management", Information Systems Management, vol. 16, issue 4, 24-32.

BRADSHAW, J. M., 1997, "An introduction to software agents". In: BRADSHAW, J. M. Ed. Software Agents. Massachusetts: MIT Press.

BRNA, P., BURTON, M., 1997, "The computer modeling the students collaborating in learning about energy". Journal of computer assisted learning, 13:193-204.

BROWN, J. S., DUGUID, P., 1998, "Organizing knowledge". California Management Review, 40 (3), 90-111.

BULL, S., BRNA, P., 1997, "What does Susan know that Paul doesn't (and vice-versa)? Contributing to each other's student model". In du Boulay, B. and Mizoguchi, R. (eds.), Artificial Intelligence in Education: Knowledge and Media in Learning Systems, pages 568-570. IOS, Amsterdam.

BURTON, M., BRNA, P., THREASURE-JONES, T., 1997, "Splitting the Collaborative Atom: How to support learning about collaboration". In du Boulay, B. and Mizoguchi, R. (eds.), Artificial Intelligence in Education: Knowledge and Media in Learning Systems, pages 135-142. IOS, Amsterdam.

CAMARGO, V.V., NÏNA, E., MALDONADO, J.C., 2006, "Um Estudo Comparativo do Tempo de Composição de um Framework Orientado a Aspectos de Persistência e de um Framework Orientado a Objetos de Persistência". In: Anais do 20º Simpósio Brasileiro de Engenharia de Software (SBES), Florianópolis, Brasil, outubro. URL: http://www.icmc.usp.br/~valter/Portuguese/Publicacoes.html, visitado em 04/08/2007.

CATLEDGE, L., PITKOW, J., 1995, "Characterizing Browsing Strategies on the World Wide Web", In: Proceedings of the 3rd International WWW Conference, Darmstad, Germany, Apr.

CHEN, M., PARK, J. S., YU, P. S., 1996, "Data Mining for Path Traversal Patterns in a Web Environment", In: Proceedings of the 16th Conference on Distributed Computing Systems, pp. 385-392, Baltimore, Maryland, USA, May.

CLANCEY. W. J., SOLOWAY. E., 1990, "Artificial Intelligence and learning environments". Artificial Intelligence. Amsterdam. v.1. n.42. p.1.

COOLEY, R. MOBASHER, B., SRIVASTAVA, J., 1999, "Data Preparation for Mining World Wide Web Browsing Patterns", Knowledge and Information Systems v.1, n.1 (Jan), pp. 5-32.

COOLEY, R., 2000, "Web Usage Mining: Discovery and Application of Interesting Patterns from Web Data", Ph.D. thesis, University of Minnesota, Minneapolis, USA.

COOLEY, R., MOBASHER, B., SRIVASTAVA, J., 1997, "Web Mining: Information and Pattern Discovery on the World Wide Web", In: Proceedings of the 9th IEEE International Conference on Tools with Artificial Intelligence (ICTAI97), Newport Beach, CA, USA, Nov.

COOLEY, R., MOBASHER, B., SRIVASTAVA, J., 1997, "Web Mining: Information and Pattern Discovery on the World Wide Web", In: Proceedings of the 9th IEEE International Conference on Tools with Artificial Intelligence (ICTAI97), Newport Beach, CA, USA.

COOLEY, R., MOBASHER, B., SRIVASTAVA, J., 1997a, "Grouping Web Page References into Transactions for Mining World Wide Web Browsing Patterns", In: Proceedings of the 1997 IEEE Knowledge and Data Engineering Exchange Workshop (KDEX-97), Newport Beach, CA, USA, Nov.

COSTA, M. T. C., 1999, "Uma Arquitetura Baseada em Agentes para Suporte ao Ensino à Distância". Tese de Doutorado. Programa de Pós-Graduação em Engenharia de Produção – UFSC. Florianópolis, 1999. Disponível em http://www.eps.ufsc.br/teses99/thiry/index.htm. Visitado em 06/10/2006.

CUSTOMERCENTRICS, 2002, "CustomerCentrics Home-page", URL: http://www.netgen.com.

DELICIOUS, 2007, "del.icio.us social bookmarking", URL: http://del.icio.us/. Visitado em 30/06/2007.

DEMAZEU, Y., 1993, "Distributed Artificial Intelligence and multiagents systems". In: Simpósio Brasileiro de Inteligência Artificial, 10. Porto Alegre. Anais. Porto Alegre: SBC.

DESMONTILS, E., JACQUIN, C., 2001, "Indexing a web site with a terminology oriented ontology". SWWS 549-565.

DILLENBOURG, P., BAKER, M., BLAYE, A., O'MALLEY, C., 1994, "The evolution of research on collaborative learning". In Spada, H. and Reimann, P. (eds.) Learning in Humans and Machines.

DIXON, N., 1937, "Common Knowledge – How Companies Thrive by Sharing What They Know". Harvard Business School Press – Boston, Massachusetts.

DOMINGUE, J., 1998, "Tadzebao and WebOnto: Discussing, Browsing, and Editing Ontologies on the Web". 11th Knowledge Acquisition for Knowledge-Based Systems Workshop, April 18th-23rd. Banff, Canada.

DOORENBOS, R. B., ETZIONI, O., WELD. D. S., 1997, "A scalable comparison shopping agent for the World Wide Web", In: Proceedings of the 1st International Conference on Autonomous Agents (AGENTS'97), pp. 39-48, New York, NY, USA.

DRUCKER, P., 1998, "Sociedade Pós-Capitalista". 7ª Edição, São Paulo, Editora Pioneira.

DYRESON, C., 1997, "Using an Incomplete Data Cube as a Summary Data Sieve", Bulletin of the IEEE Technical Committee on Data Engineering, Mar, pp. 19 - 26.

FENSEL, D. et al., 2000, "OIL in a nutshell In: Knowledge Acquisition, Modeling, and Management", Proceedings of the European Knowledge Acquisition Conference (EKAW-2000), R. Dieng et al. (eds.), Lecture Notes in Artificial Intelligence, LNAI, Springer-Verlag, October.

FERBER, J., 1999, "Multi-Agents Systems. An Introduction to Distributed Artificial Intelligence". Addison-Wesley.

FERREIRA, A., 1999, "Dicionário Aurélio eletrônico. V. 2.0". Rio de Janeiro: Nova Fronteira.

FIELDING, R., IRVINE, U. C., GETTYS, J. et al., 1997, "Hypertext Transfer Protocol - HTTP/1.1", Request for Comments 2616, URL: http://www.w3.org/Protocols/History.html

FIPA, Foundation for Intelligent Physical Agents, 1997, "Agent communication Language", Geneva, FIPA97 Specification, part2. Disponível em http://www.fipa.org. Visitado em 06/10/2006.

FRAPPAOLO, C., 2000, "Ushering in the Knowledge-Base Economy". Disponível em: http://www.delphigroup.com. Visitado em 30/08/2000.

FREUND, J.E., PERLES, B.M., 1998, "Statistics: a First Course", Seventh Edition, Prentice-Hall, Englewood Cliffs, NJ.

GOLDBERG, D. et al., 1992, "Using collaborative filtering to weave an information Tapestry", Communications of the ACM, New York, v.35, n.12, p. 61-70, Dec.

GONSALVES A., 2006, "Firefox Gains Market Share Against Microsoft Internet Explorer", Information Week, 12 de maio, URL: http://www.informationweek.com/software/showArticle.jhtml?articleID=187202805, visitado em 05/04/2007.

GRICE, H.P., 1975, "Logic of conversation". In Davidson, D. and Harman, G., (eds.), The Logic of Grammar. Dickenson.

GUARINO, N., 1998, "Formal Ontology in Information Systems", Proc. of FOIS'98, Trento, Italy, IOS Press, June.

HALLAM-BAKER, P.M., BEHLENDORF, B., 1996, Extended Log File Format, W3C Working Draft WD-session-id-960323, URL: http://www.w3.org/pub/WWW/TR/WDlogfile.html.

HALLAM-BAKER, P.M., CONNOLLY, D., 1996a, Session Identification URI, W3C Working Draft WD-session-id-960221, URL: http://www.w3.org/pub/WWW/TR/WDsession-id.html

HEFLIN, J., 2001, "Towards the Semantic Web: Knowledge Representation in a Dynamic Distributed Environment", by. Ph.D. Thesis, University of Maryland, College Park.

HORROCKS, I., PATEL-SCHNEIDER, P. F., VAN HARMELEN, F., 2002, "Reviewing the Design of {DAML+OIL}: An Ontology Language for the Semantic Web". Presented at 18th Nat. Conf. on Artificial Intelligence (AAAI~2002).

HU J., 2005, "Firefox continues gains against IE", CNET News.com, 21 de janeiro, URL: http://news.com.com/Firefox+continues+gains+against+IE/2100-1032_3-5545930.html, visitado em 05/04/2007.

JADE, 2007, "Java Agent DEvelopment Framework", URL: http://jade.tilab.com/, visitado em 11/01/2007

JOACHIMS, T., FREITAG, D., MITCHELL, T., 1997, "Webwatcher: A tour guide for the world wide web", Proc. of the 15th IJCAI, Nagoya, Japan, 770-775.

KARP, P. D., CHAUDHRI, V. K., THOMERE, J., 1999, "XOL: An XML-Based Ontology Exchange Language", August 31

KATO, H., NAKAYAMA, T., YAMANE, Y., 2000, "Navigation analysis tool based on the correlation between contents distribution and access patterns", In: Worskhop on Web Mining for E-Commerce - Challenges and Opportunities, in WEBKDD'2000, Boston, MA, USA, Aug, URL: http://robotics.stanford.edu/~ronnyk/WEBKDD2000/papers.

KENT, R. E., 1999, "Conceptual Knowledge Markup Language: The Central Core". In the Electronic Proceedings of the Twelfth Workshop on Knowledge Acquisition, Modeling and Management (KAW'99). Banff, Alberta, Canada, 16–21 October.

KIMBALL, R., MERZ, R., 2000, The Data Webhouse Toolkit: Building the Web-Enabled Data Warehouse, 1 ed., New York, USA, John Wiley & Sons.

KLENSIN, J., 2001, "RFC 2821 Simple Mail Transfer Protocol", The Internet Engineering Task Force, URL: http://tools.ietf.org/html/rfc2821, visitado em 24/05/2007.

KOBAYASHI, M., TAKEDA, K.M, 2000, "Information retrieval on the Web", ACM Computing Surveys, v. 32, n. 2 (Jun), pp. 144-173.

KUUSI, O., 1999, "Growing and learning entrepreneurial networks ad the focus of the national innovation strategy", in Schienstock and Kuusi, eds.

LAMPREIA, C., 1993, "Os Limites das Propostas Anti-Mentalistas Behavioristas", Informativo nº. 2 da ABPMC, Brasília, DF, Brasil.

LAROCCA NETO, J.; SANTOS, A. D.; KAESTNER, C. A. A. & FREITAS, A. A., 2000 "Document clustering and text summarization". In Proceedings of the 4th Int. Conf. Practical Applications of Knowledge Discovery and Data Mining (Padd-2000), 41-55. London: The Practical Application Company.

LAVE, J., WENGER, E., 1991, "Situated Learning: Legitimate Peripheral Participation". Cambridge University, New York, NY.

LEITCH, M., 1986, "Human Knowledge Design by undergraduate project", URL: http://www.learningideas.me.uk/HKDesign/HKDdissertation.html, visitado em 04/06/2007.

LI, T., 2001, Web-Document Prediction And Presending Using Association Rule Sequential Classifiers, M.Sc. dissertation, School of Computing Science, Simon Fraser University, Vancouver, BC, Canada.

LINUX Online Inc., 2007, "The Linux Home Page at Linux Online", URL: http://www.linux.org/, visitado em 05/04/2007.

LUOTONEN, A., NIELSEN, H. F., BERNERS-LEE, T., 1995, The Common Logfile Format. http://www.w3.org/Daemon/User/Config/Logging.html, 1995.

MACEDO, H. T., RAMALHO, G. L., 2001, "Mobilidade, Autonomia e Distribuição em Agentes para gerenciamento Corporativo de Sistemas". Dissertação de Mestrado do Curso de Ciência da Computação. Recife : Universidade Federal de Pernambuco.

MAEDCHE, A., STAAB, S., STOJANOVIC, N., et al., 2001, "SEmantic PortAL – The SEAL approach", In: Fensel, D. , Hendler, J., Lieberman, H., Wahlster, W. (eds.), Creating the Semantic Web, MIT Press, Cambridge, MA, USA.

MAGALHÃES, J. A. P., SARDINHA, J. A. R. P., 2001, "A Critical Look Upon Agent UML". Nos anais do 1o. Seminário sobre Sistemas Multi-Agentes realizado na PUC-Rio, páginas 69-82.

MAGALHÃES, J. A., 2002, "Um Framework Multi-Agentes para Busca e Flexibilização de Algoritmos de Classificação de Documentos", Tese de Mestrado, Departamento de Informática, PUC-RJ

MANNILA, H., TOIVONEN, H., 1996, "Discovering Generalized Episodes Using Minimal Ocurrences", In: Proceedings of 2nd ACM SIGKDD International Conference On Knowledge Discovery and Data Mining (KDD96), pp. 146-151, Portland, OR, USA, Aug.

MATHES, A., 2004, "Folksonomies - Cooperative Classification and Communication Through Shared Metadata". Illinois, December. Disponível em http://www.adammathes.com/academic/computer-mediated-communication/folksonomies.html, visitado em 22/01/2007.

MATOS, M. A, 1993, "Behaviorismo metodológico e behaviorismo radical", II Encontro Brasileiro de Psicoterapia e Medicina Comportamental, Campinas, SP, Brasil.

MCDERMOTT, R., 2000, "Knowing in Community: 10 Critical Success Factors in Building Communities of Practice". IHRIM Journal .

MCDERMOTT, R., O'DELL, C., 2000, "Overcoming cultural barriers to sharing knowledge." Under review.

MICROSOFT Corporation , 2007b, "Windows Home Page", URL: http://www.microsoft.com/windows/default.mspx, visitado em 05/04/2007.

MICROSOFT Corporation, 2007, "Home : The Official Microsoft ASP.NET 2.0 Site : English", URL: http://www.asp.net/, visitado em 03/04/2007.

MIRANDA, M., XEXÉO, G., SOUZA, J. M., 2006, "Building Tools for Emergent Design with COPPEER". In: The 10th International Conference on CSCW in Design, 2006, Nanjing, China. The 10th International Conference on CSCW in Design.

MLADENIC, D., 1999, "Text learning and related intelligent agents: a survey", IEEE Intelligent Systems, v. 14, n. 4 (Jul), pp. 44-54.

MOREY, D., FRANGIOSO, T., 1997, "Knowledge Management Systems". MITRE Organization, Massachusetts.

MOZILLA Foundation, 2007, "Firefox – Rediscover the web", URL: http://www.mozilla.com/en-US/firefox/, visitado em 05/04/2007.

MOZILLA Foundation, 2007b, "XML User Interface Language (XUL)", URL: http://www.mozilla.org/projects/xul/, visitado em 05/04/2007.

MOZILLA Foundation, 2007c, "JavaScript – MDC", URL: http://developer.mozilla.org/en/docs/JavaScript, visitado em 05/04/2007.

MYSQL, 2007, "MySQL", URL: http://www.mysql.com/, visitado em 10/04/2007

NONAKA, I., TAKEUCHI, H., 1995, "The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation". Oxford Univ. Press.

NONAKA, I., TAKEUCHI, H., 1995, "The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation". Oxford Univ. Press.

NOY, N. F., MCGUINNESS, D. L., 2001, "Ontology Development 101: A Guide to Creating Your First Ontology". Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880, March 2001.

NWANA, H. S., NDUMU, D. T., 1999, "A Perspective on Software Agents Research". The Knowledge Engineering Review. Vol 14, No 2, pp 1-18.

ODELL, J., 1999, "Objects and Agents: How do they differ?". Draft.

ODELL, J., PARUNAK, H. V. D., BAUER, B., 2000, "Extending UML for Agents". Em Proceedings of the Agent-Oriented Information Systems Workshop at the 17th National conference on Artificial Intelligence.

OMG, Object Management Group, 2000, "Agent Technology – Green Paper". Version 1.0. Agent Platform Special Interest Group.

OVIDIUS, P. N., 14, "Metamorphoseon", v. 1.

PARUNAK, H. V. D., 1997, "'Go to the Ant': Engineering Principles from Natural Agent Systems", Em Annals of Operations Research, 75, páginas 69-101.

PAWLOWSKI, S., ROBEY, D., RAVEN, A., 2000, "Supporting shared information systems: boundary objects, communities, and brokering", Proc. of the 21st Int. Conf. on Information Systems. Brisbane, Australia, pp. 329-338.

PEREIRA, V. B., REZENDE, J. L., XEXEO, G. B., SOUZA, J. M., 2006, "Building a Personal Knowledge Recommendation System using Agents, Learning Ontologies and Web Mining". In: 10th International Conference o Computer Supported Cooperatice Work in Design, 2006, Nanjing. Proceedings of 10th International Conference o Computer Supported Cooperative Work in Design, v. I. p. 147-142.

PEREIRA, V. B., REZENDE, J. L., XEXEO, G. B., SOUZA, J. M., 2007, "Olympus: Personal Knowledge Recommendation using Agents, Ontologies and Web Mining.", Lecture Notes in Computer Science.

PERKOWITZ, M., ETZIONI, O., 1997, "Adaptive web sites: an AI challenge", In: Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI97), pp. 16-23, Nagoya, Japan, Aug.

PERKOWITZ, M., ETZIONI, O., 1998, "Adaptive web sites: Automatically synthesizing web pages", In: Proceedings of the 15th National Conference on Artificial Intelligence, pp. 727-732, Madison, WI, USA, Jul.

PERKOWITZ, M., ETZIONI, O., 1999, "Adaptive web sites: Conceptual cluster mining", In: Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI99), pp. 264-269, Stockholm, Sweden, Aug.

PIROLLI, P., PITKOW, J., RAO, R., 1996, "Silk from a Sow's Ear: Extracting Usable Structures from the Web", In: Proceedings of the ACM SIGCHI'96 Conference on Human Factors in Computing Systems, pp. 118-125, Vancouver, BC, Canada, Apr.

PITKOW J., 1997, "In Search of Reliable Usage Data on the WWW", In: Proceedings of the 6th International WWW Conference, pp. 451-463, Santa Clara, CA, USA.

PITKOW, J., BHARAT, K., 1994, "Webviz: a Tool for World-Wide-Web Access Log Analysis", In: Proceedings of the 1st International WWW Conference, Geneve, Switzerland, May.

POLANYI, M., 1983, "The Tacit Dimension". Peter Smith Pub, London.

PROTÉGÉ, 2007, "The Protégé Ontology Editor and Knowledge Acquisition System", Stanford Medical Informatics, URL: http://protege.stanford.edu/, visitado em 03/06/2007.

RESNICK, P. et al., 1994, "GroupLens: An Open Architecture for Collaborative Filtering of Netnews". In Proc. Proceedings of ACM Conference on Computer Supported Cooperative Work. Chapel Hill, North Carolina, p.175-186.

RESNICK, P.; VARIAN, H. R., 1997, "Recommender Systems", Communications of the ACM, New York, v.40, n.3, pp. 55-58, Mar.

ROSCHELLE, J., TEASLEY, S., 1995, "The construction of shared knowledge in collaborative problem solving". In O'Malley, C. E., (ed.), Computer Supported Collaborative learning. Pages 69-97. Springer-Verlag, Heildelberg.

SALTON, G., BUCKLEY, C., 1988, "Term-weighting Approaches in Automatic Text Retrieval. Information Processing and Management", 24, pp. 513-523.

SANE, 2002, "Analyzing web site traffic, Sane Solutions white paper", URL: http://www.sane.com/products/NetTracker/whitepapers.html

SCALABRIN, E. E., 1996, "Conception et Réalization d'environnement de développement de systèmes d'agents cognitifs". Tese de doutorado, 1996. Université de Technologie de Compiègne, France.

SCHAFER, J.B., KONSTAN, J.A., RIEDL, J., 2001, "E-Commerce Recommendation Applications". Data Mining and Knowledge Discovery, Vol. 5. (2001) 115-153.

SHAHABI, C., BANAEI-KASHANI, F., FARUQUE, J., 2001, A Reliable, Efficient, and Scalable System for Web Usage Data Acquisition, In: Proceedings of the WebKDD'2001 Workshop in conjunction with the ACM-SIGKDD 2001, San Francisco, CA, Aug.

SHAHABI, C., ZARKESH, A. M., ADIBI, J., SHAH, V., 1997, "Knowledge discovery from users web page navigation", In: Proceedings of the International

Workshop on Research Issues in Data Engineering IEEE (RIDE'97), pp. 20 - 31, Birmingham, UK, Apr.

SHOHAM, Y., 1993, "Agent-Oriented Programming". Artificial Intelligence, 60: 24-29.

SICHMAN, J. S., DEMAZEU, Y., BOISSIER, O., 1992, "How can Knowledge-based systems be called agents?", In: Simpósio Brasileiro de Inteligência Artificial, 9, Rio de Janeiro. Rio de Janeiro: Pontifícia Universidade Católica do Rio de Janeiro, p.173-185.

SILVA, R. L. S., REZENDE, J. L., SOUZA, J. M., RAMIREZ, M. R., 2006 "An experiment in exchanging knowledge chains to build personal knowledge". International Journal of Web Based Communities, v. 2, p. 413-427.

SILVA, R.L.S, REZENDE, J.L., DE SOUZA, J.M., RAMIREZ, M., 2005, "Building Personal Knowledge through Exchanging Knowledge Chains", Proc. of IADIS Int. Conf. on WBC. Algarve, Portugal, pp. 87-94.

SOUZA, J.F., REZENDE, J.L., DE SOUZA, J.M., 2005, "Peer - to - Peer Collaborative Integration of Dynamic Ontologies", 9th Int. Conf. on CSCWD, Coventry, UK.

SPILIOPOULOU, M., FAULSTICH, L. C., 1998, "WUM: A Web Utilization Miner", In: Proceedings of the EDBT Workshop, WebDB98, LNCS - Lecture Notes in Computer Science, v. 1590, Springer-Verlag, Valencia, Spain.

SPILIOPOULOU, M., FAULSTICH, L.C., WINKLER, K., 1999, "A Data Miner analyzing the Navigational Behaviour of Web Users", In: Proceedings of Workshop on Machine Learning in User Modeling of the ACAI'99, Creta, Grécia, Jul.

SRIKANT, R., AGRAWAL, R., 1996, "Mining sequential patterns: generalizations and performance improvements", In: Proceedings of the 5th International Conference on Extending Database Technology (EDBT'96) , pp. 3-17, Avignon, France, Mar.

SU, Z., YANG, Q., ZHANG, H., et al., 2001, "Correlation-based Document Clustering using Web Logs", In: Proceedings of the 34th IEEE Hawaii's International Conference on System Sciences (HICSS-34), Hawaii, USA, Jan.

SUN Microsystems, 2006, "JavaServer Pages Technology", URL: http://java.sun.com/products/jsp/, visitado em 03/04/2007.

SUN Microsystems, 2007, "Java Technology Concept Map", URL: http://java.sun.com/developer/onlineTraining/new2java/javamap/intro.html, visitado em 03/06/2007.

SUN Microsystems, 2007b, "JavaMail API", URL: http://java.sun.com/products/javamail/, visitado em 03/06/2007.

SUPNITHI, T., et al., 1999, "Learning Goal Ontology Supported by Learning Theories for Opportunistic Group Formation", In Artificial Intelligence in Education S.P.Lajoie and M.Vivet (Eds.), IOS Press.

SVEIBY, K. E,;LLOYD, T., 1987, "Managing Knowhow: Add value... By valuing Creativity". London: Bloomsbury Publishing Limited.

TECHNORATI, 2007, "Technorati: Home", URL: http://technorati.com/, Visitado em 30/06/2007.

TORNAGHI, A., VIVACQUA, A., SOUZA, J.M., 2005, "Creating Educator Communities", Int. Journal Web Based Communities. Grã-Bretanha, 001 – 015.

TRAVASSOS, M.O.B.C.M.L.W.G.H., 2005, "Um Estudo Experimental sobre a Utilização de Modelagem e Simulação no Apoio à Gerência de Projetos de Software". In: XIX Simpósio Brasileiro de Engenharia de Software (SBES), Uberlândia, MG, Brasil, Outubro de 2005.

VAN SOLINGEN, R., BERGHOUT, E., 1999, "The Goal / Question / Metric Method: A Practical Guide for Quality Improvement of Software Development", McGrawn Hill. ISBN 0077095537

VANDERWAL, T., 2005, "Off the Top: Folksonomy Entries.", http://www.vanderwal.net/random/category.php?cat=153, visitado em 22/01/2007.

VARELLA, A., 2007, "COOPRACTICE – Comunidades de prática virtuais apoiadas por ontologias", Dissertação de Mestrado, Programa de Engenharia de Sistemas e Computação, Universidade Federal do Rio de Janeiro, Rio de Janeiro.

VICCARI, R. M, 1989. "Um tutor inteligente para a programação em lógica: idealização, projetos e desenvolvimento". Coimbra: Universidade de Coimbra. Tese de doutorado.

W3C, 1999, "HyperText Markup Language (HTML) Home Page", World Wide Web Consortium, URL: http://www.w3.org/TR/html401/, visitado em 31/03/2007.

W3C, 2001, "URIs, URLs, and URNs: Clarifications and Recommendations 1.0", Report from the joint W3C/IETF URI Planning Interest Group, URL: http://www.w3.org/TR/uri-clarification/.

W3C, 2004, "RDF Vocabulary Description Language 1.0: RDF Schema", World Wide Web Consortium, URL: http://www.w3.org/TR/rdf-schema/, visitado em 22/01/2007.

W3C, 2006, "Extensible Markup Language (XML)", World Wide Web Consortium, URL: http://www.w3.org/XML/, visitado em 05/04/2007.

W3C, 2007, "Web Services Activity", World Wide Web Consortium, URL: http://www.w3.org/2002/ws/, visitado em 05/04/2007.

W3C, 2007b, "W3C XML Protocol Working Group", World Wide Web Consortium, URL: http://www.w3.org/2000/xp/Group/, visitado em 05/04/2007.

W3C, 2007c, "Web Services Description Working Group", World Wide Web Consortium, URL: http://www.w3.org/2002/ws/desc/, visitado em 05/04/2007.

WEBTRENDS, 2007, "WebTrends Marketing Web Analytics and Web Statistics", URL: http://www.webtrends.com, visitado em 13/01/2006.

WENGER, E., 1998, "Communities of Practice: Learning, Meaning, and Identity". Cambridge University Press.

WENGER, E., MCDERMOTT, R., SNYDER, W.M., 2002, "Seven Principles for Cultivating Communities of Practice". Cultivating Communities of Practice: A Guide to Managing Knowledge, Harvard Business School Press.

WOHLIN, C., RUNESON, P., HÖST, M., OHLSSON, M.C., REGNELL, B., WESSLÉN, A., 2000, "Experimentation in Software Engineering: an Introduction", Kluver Academic Publishers, Norwell, MA

XEXÉO, G., et al, 2004, "COE: A Collaborative Ontology Editor Based on a Peer-to-Peer Framework", Int. Journal of Advanced Engineering Informatics, Germany.

YAN, T. W., JACOBSEN, M., GARCIA-MOLINA, H. et al., 1996, "From User Access Patterns to Dynamic Hypertext Linking", In: Proceedings of the 5th International WWW Conference, pp. 1007-1014, Paris, France, May.

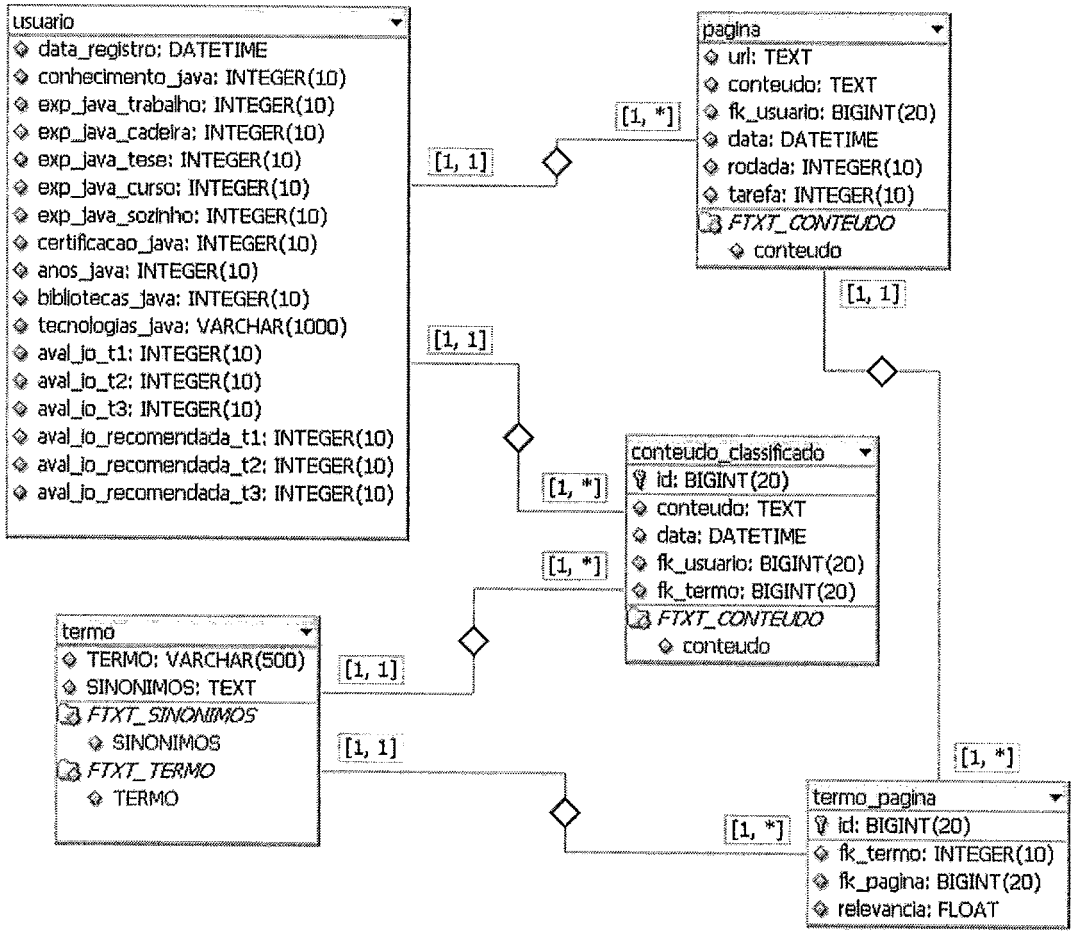
ZAÏANE, O. R., 1999, Resource and Knowledge Discovery from the Internet and Multimedia Repositories, Ph.D. thesis, School of Computing Science, Simon Fraser University, Vancouver, BC, Canada.

ZAÏANE, O. R., 2000, "Web Mining: Concepts, Practices and Research", Conference Tutorial Notes, XIV Brazilian Symposium on Databases (SBBD'2000), pp. 410-474, João Pessoa, Paraíba, Brasil.

ZAÏANE, O. R., XIN, M., HAN, J., 1998, "Discovering Web Access Patterns and Trends by Applying OLAP and Data Mining Technology on Web Logs", In: Proceedings of Advances in Digital Libraries Conference (ADL'98), pp. 19-29, Santa Barbara, CA, USA, Apr.

ZAKI, M. J., LESH, N., OGIHARA, M., 1998, "PLANMINE: Sequence Mining for Plan Failures", In: Proceedings of 4th ACM SIGKDD International Conference On Knowledge Discovery and Data Mining (KDD98), pp. 369-374, New York, USA, Aug.

Apêndice A - Modelo de Dados



Apêndice B - Tabela de média de avaliações

Tabela B.1 - Número de páginas visitadas por usuário x tarefa x rodada

	Tarefa 1			Tarefa 2			Tarefa 3		
Usuário	**Rodada 1**	**Rodada 2**	**Rodada 3**	**Rodada 1**	**Rodada 2**	**Rodada 3**	**Rodada 1**	**Rodada 2**	**Rodada 3**
1			1	6				2	
4	15				10				5
6	13				1				13
7		10				7	36		
10			1	5				4	
11		5				16	22		
12		2				2	26		
14	8					5		9	
15	1				3				13
Média	9,3	5,7	1,0	5,5	4,7	7,5	28,0	5,0	10,3

Tabela B.2 - Tempo utilizado para pesquisa em segundos por usuário x tarefa x rodada

	Tarefa 1			Tarefa 2			Tarefa 3		
Usuário	**Rodada 1**	**Rodada 2**	**Rodada 3**	**Rodada 1**	**Rodada 2**	**Rodada 3**	**Rodada 1**	**Rodada 2**	**Rodada 3**
1			120	1598				217	
4	2731				682				199
6	708				120				1196
7		643				119	1947		
10			120	272				929	
11		807				1712	2369		
12		305				305	4213		
14	413					848		945	
15	120				65				628
Média	993,0	585,0	120,0	935,0	289,0	746,0	2843,0	697,0	674,3

Tabela B.3 - Número de conceitos existentes nas cadeias geradas a partir da navegação por usuário x tarefa

Usuário	Tarefa 1	Tarefa 2	Tarefa 3
0	20	9	21
1	1	3	1
4	8	4	3
5		7	
6	6	1	6
7	2	3	5
9	6	6	4
10	1	3	1
11	4	8	4
12	2		8
14	1	4	3
15	1	3	3
Média	3,2	4,2	3,8

Tabela B.4 - Avaliação das cadeias criadas pelo sistema por usuário x tarefa e por usuário x rodada

	Cadeias do usuário			Cadeias recomendadas			Cadeias recomendadas	
Usuário	**Tarefa 1**	**Tarefa 2**	**Tarefa 3**	**Tarefa 1**	**Tarefa 2**	**Tarefa 3**	**Rodada 2**	**Rodada 3**
1	3	3	3	3		3	3	3
4	2		1		1	1	1	1
5	3	2	2	3		1	1	3
6	1	2	2		2	3	2	3
7	2	2	2	1	2		1	2
9	1	2	3		3	2	3	2
10	3	2	2	3		3	3	3
11	2	2	2	1	3		1	3
12	2	3	2	3	2		3	2
14	3	2	1		3	1	3	1
15	2	2	2		2	2	2	2
Média	2,2	2,2	2,0	2,3	2,3	2,0	2,1	2,3

Apêndice C – Conceitos Navegados

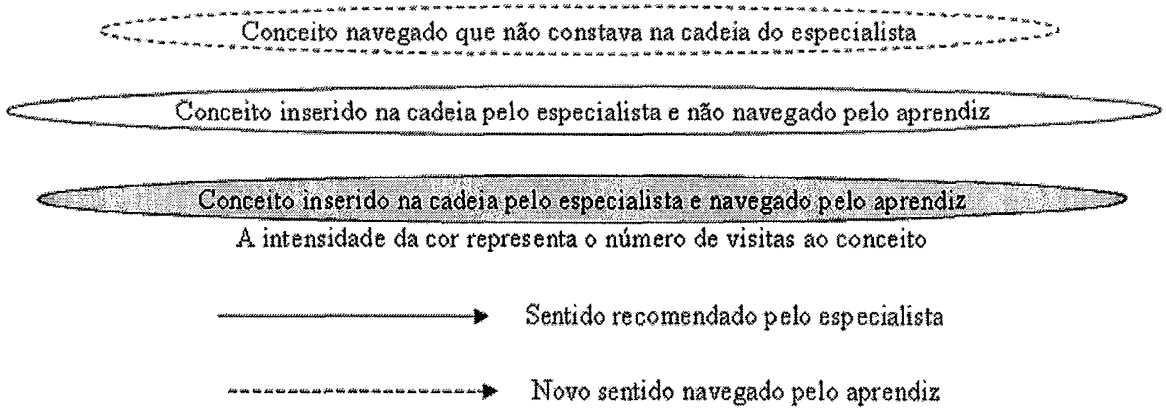


Figura C.1 - Legenda das figuras de conceitos navegados

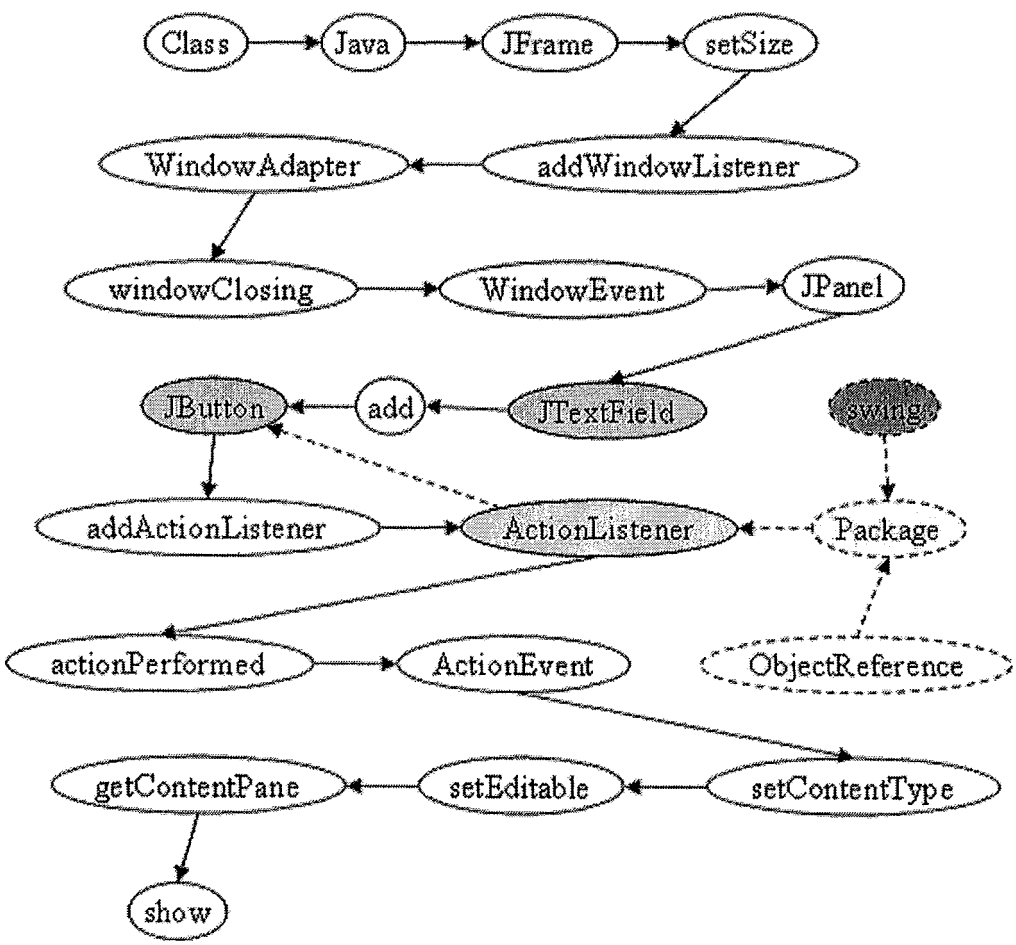


Figura C.2 - Conceitos navegados para cumprir a Tarefa 1 na rodada 1

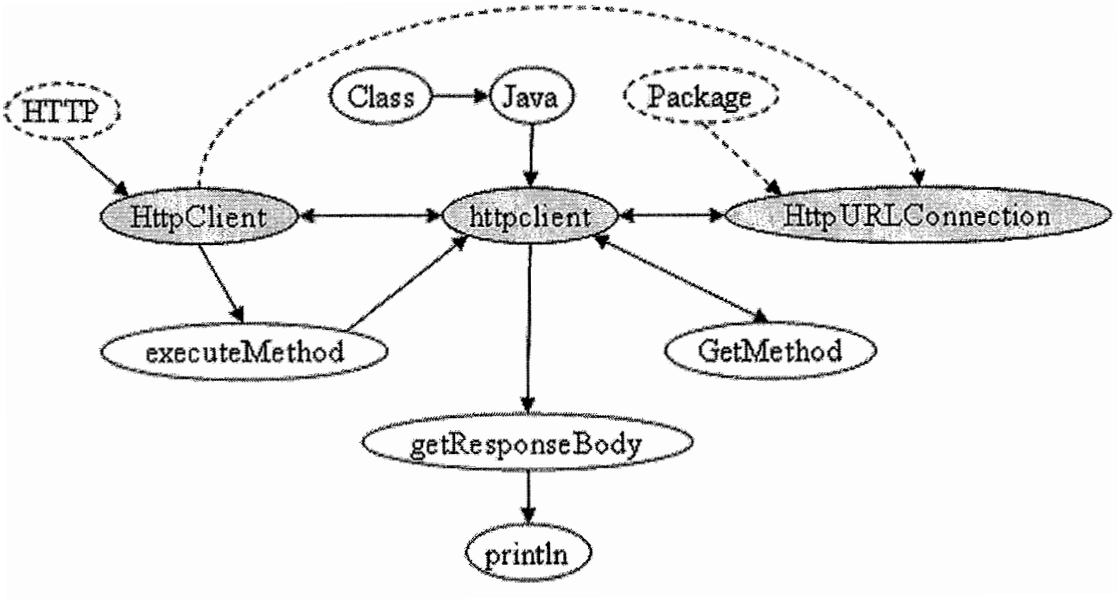


Figura C.3 - Conceitos navegados para cumprir a Tarefa 2 na rodada 1

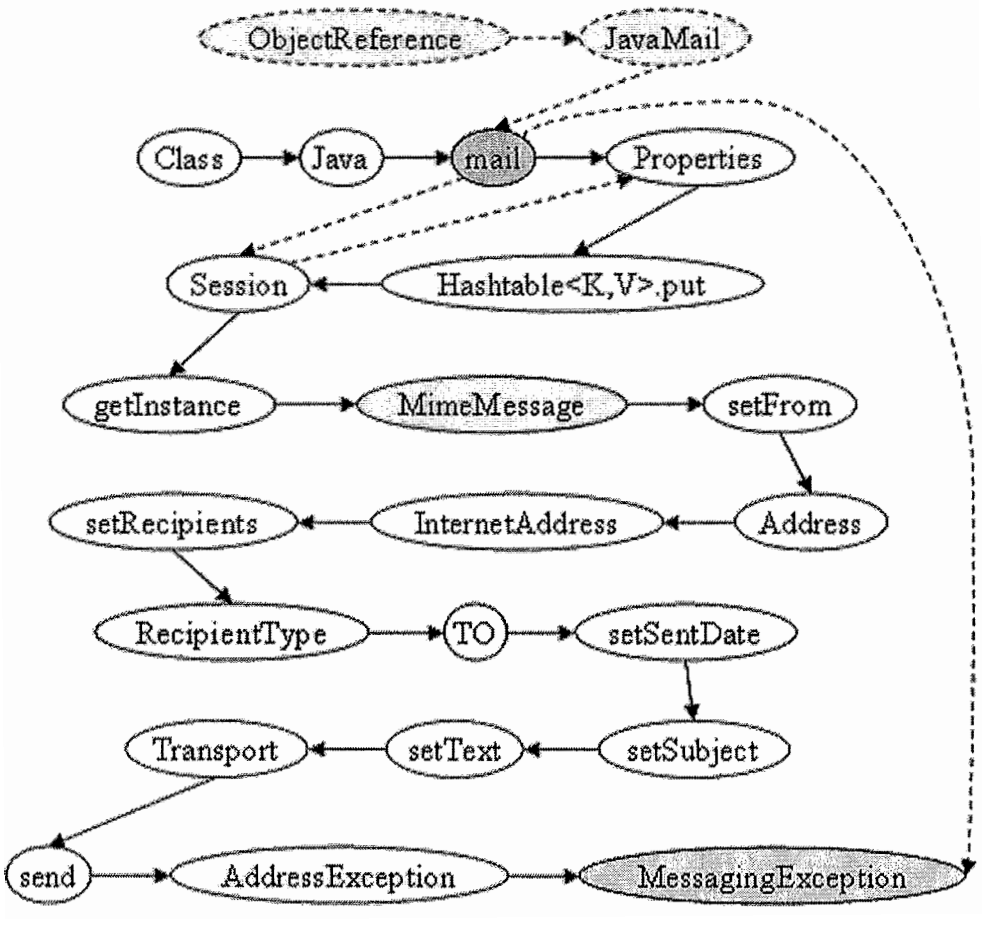


Figura C.4 - Conceitos navegados para cumprir a Tarefa 3 na rodada 1

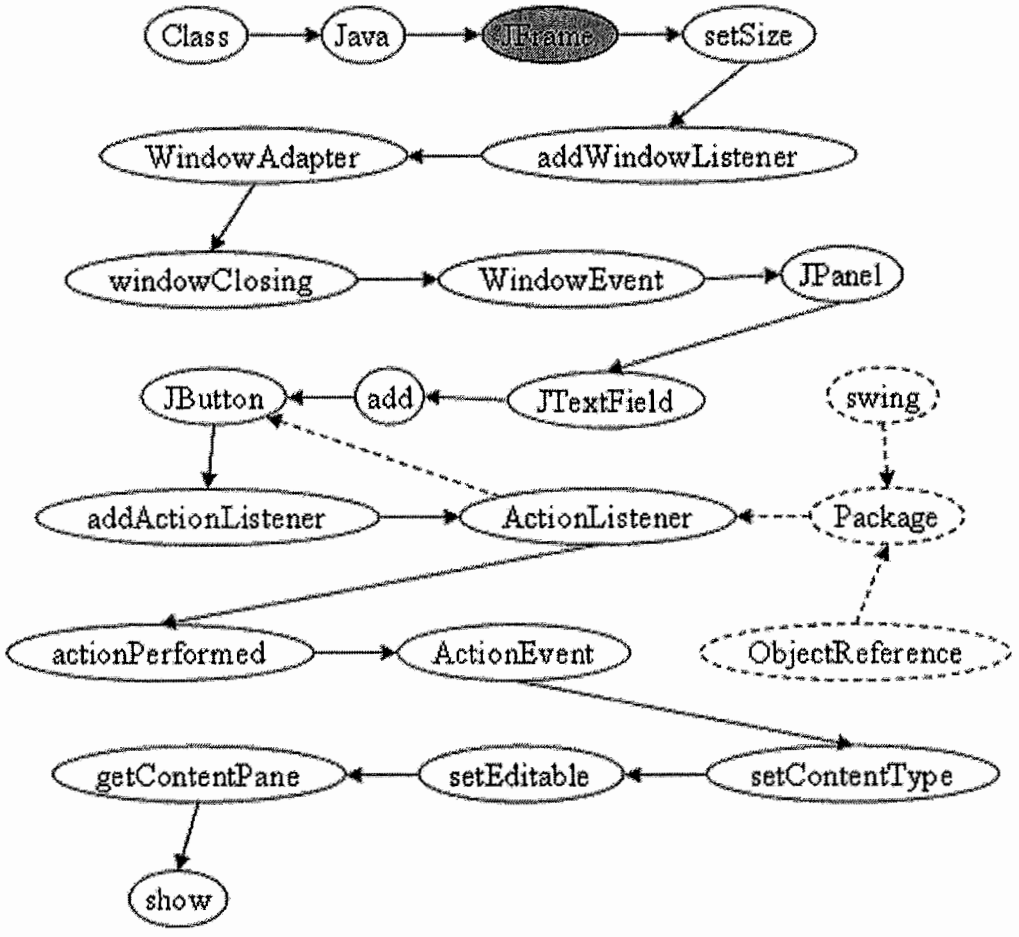


Figura C.5 - Conceitos navegados para cumprir a Tarefa 1 na rodada 3

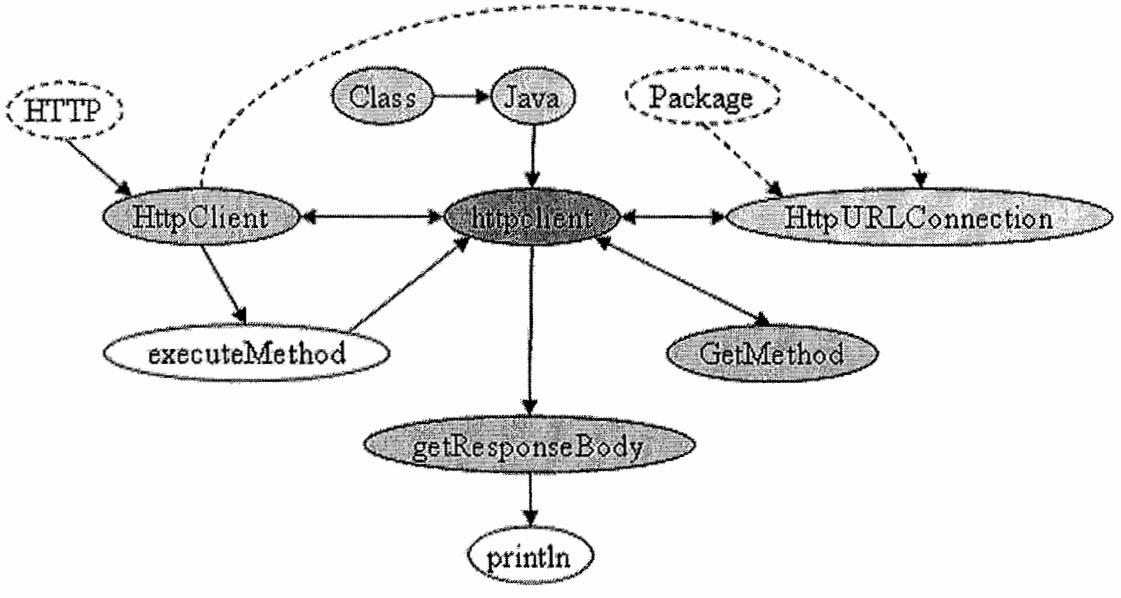


Figura C.6 - Conceitos navegados para cumprir a Tarefa 2 na rodada 3

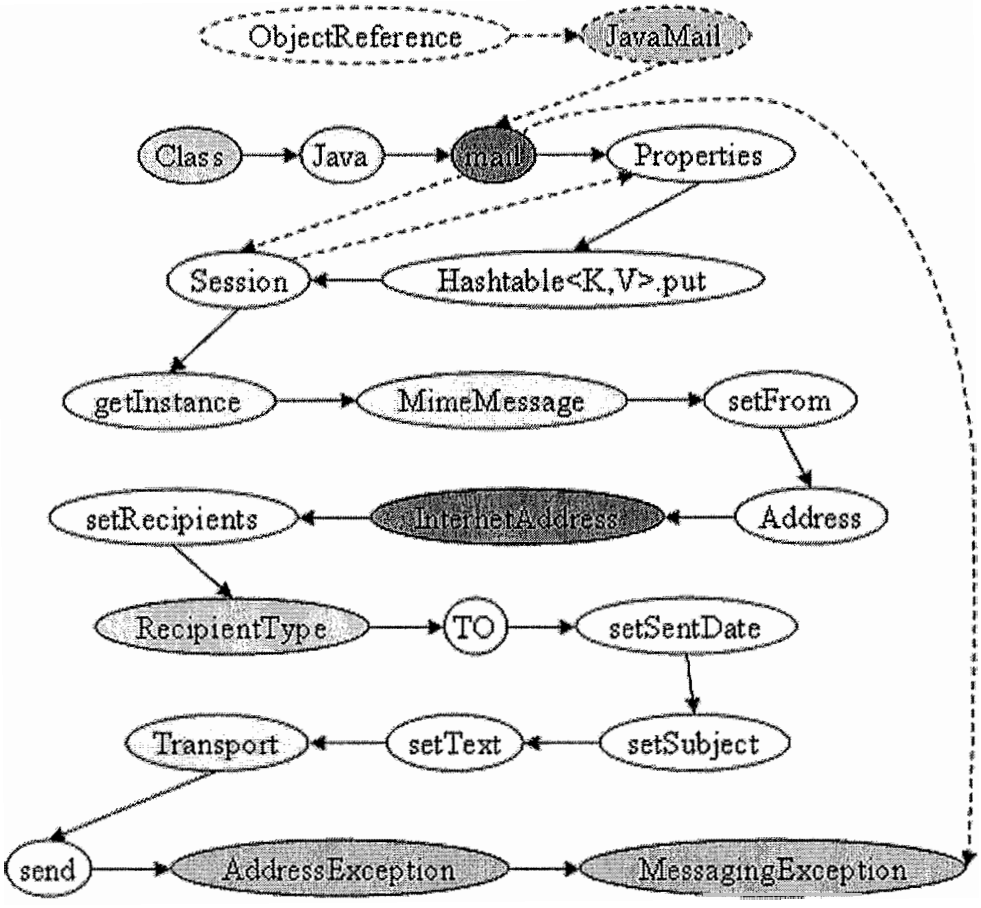


Figura C.7 - Conceitos navegados para cumprir a Tarefa 3 na rodada 3

Apêndice D – Diagramas de Classes

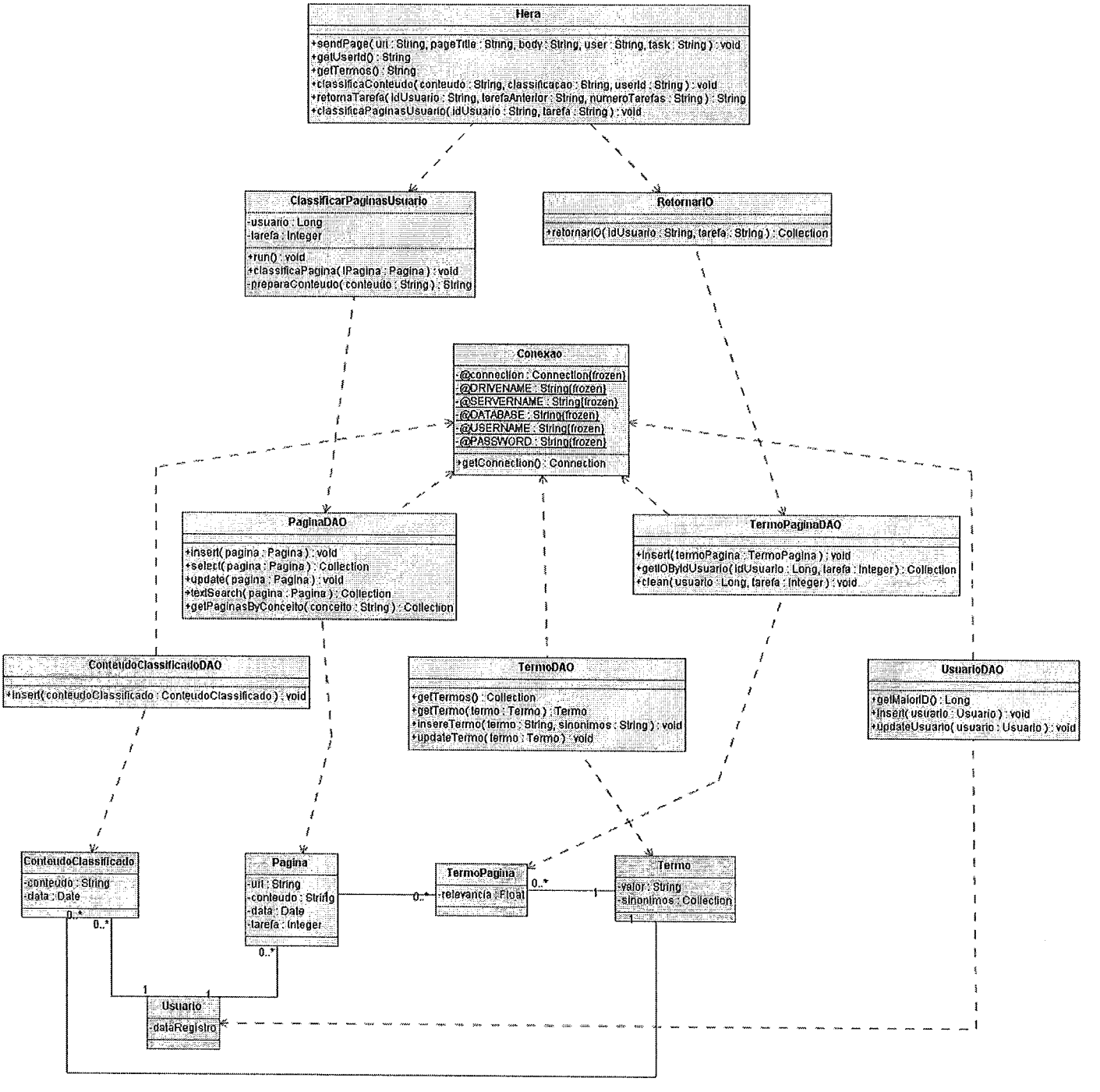


Figura D.1 - Diagrama de classes do serviço Web Hera

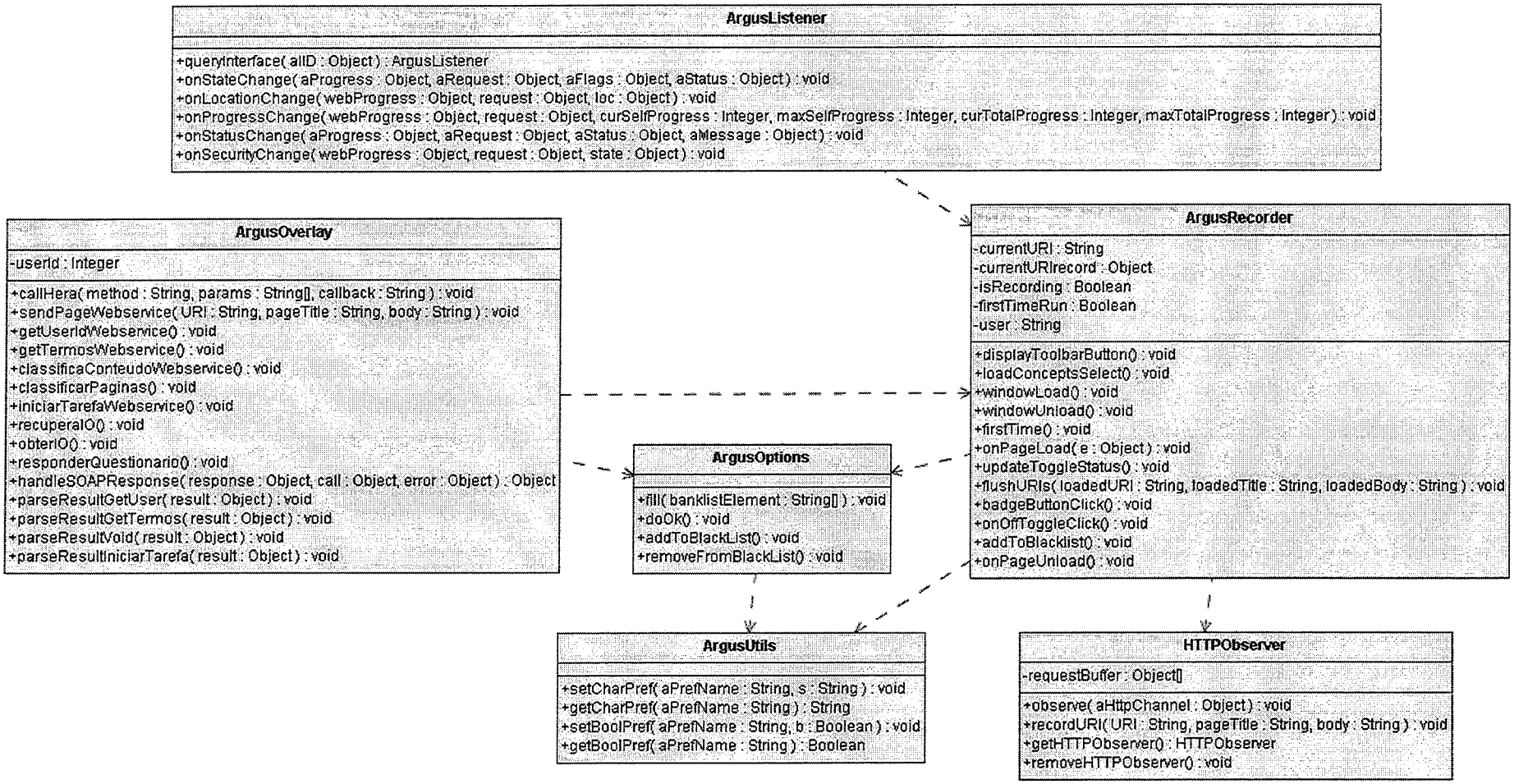


Figura D.2 - Diagrama de classes do *plug-in* Argus

Apêndice E – Ontologia

```
<?xml version="1.0"?>
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xmlns:owl="http://www.w3.org/2002/07/owl#"
   xmlns="http://www.owl-ontologies.com/unnamed.owl#"
  xml:base="http://www.owl-ontologies.com/unnamed.owl">
  <owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="OOLanguage">
   <rdfs:subClassOf>
    <owl:Class rdf:ID="Concept"/>
   </rdfs:subClassOf>
   <rdfs:subClassOf>
    <owl:Restriction>
     <owl:someValuesFrom>
      <owl:Restriction>
       <owl:allValuesFrom>
        <owl:Class rdf:ID="Heritage"/>
       </owl:allValuesFrom>
       <owl:onProperty>
        <owl:TransitiveProperty rdf:ID="contém"/>
       </owl:onProperty>
      </owl:Restriction>
     </owl:someValuesFrom>
     <owl:onProperty>
      <owl:TransitiveProperty rdf:about="#contém"/>
     </owl:onProperty>
    </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >oo language</rdfs:label>
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >linguagem orientada a objeto</rdfs:label>
  </owl:Class>
  <owl:Class rdf:ID="Public">
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >públicos</rdfs:label>
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >pública</rdfs:label>
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >público</rdfs:label>
   <rdfs:subClassOf rdf:resource="#Concept"/>
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >public</rdfs:label>
  </owl:Class>
```

```
<owl:Class rdf:ID="Boolean">
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >boolean</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >booleano</rdfs:label>
  <rdfs:subClassOf>
    <owl:Class rdf:ID="Primitive"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="#Primitive">
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >primitivo</rdfs:label>
  <rdfs:subClassOf>
    <owl:Class rdf:ID="DataType"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="Declaration">
  <rdfs:subClassOf rdf:resource="#Concept"/>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >statement</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >declaration</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >declaraÃ§Ã£o</rdfs:label>
</owl:Class>
<owl:Class rdf:ID="Constructor">
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >constructor</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >construtor</rdfs:label>
  <rdfs:subClassOf>
    <owl:Class rdf:ID="Method"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="TypeCast">
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >cast</rdfs:label>
  <rdfs;subClassOf rdf:resource="#Concept"/>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >type cast</rdfs:label>
</owl:Class>
<owl:Class rdf:ID="LogicalOperator">
  <rdfs:subClassOf>
    <owl:Class rdf:ID="Operator"/>
  </rdfs:subClassOf>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >logical operator</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >operador lÃ³gico</rdfs:label>
</owl:Class>
```

```
<owl:Class rdf:ID="Classpath">
 <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
 >classpath</rdfs:label>
 <rdfs:subClassOf rdf:resource="#Concept"/>
</owl:Class>
<owl:Class rdf:about="#Method">
 <rdfs:subClassOf>
  <owl:Restriction>
   <owl:onProperty>
    <owl:TransitiveProperty rdf:about="#contém"/>
   </owl:onProperty>
   <owl:allValuesFrom>
    <owl:Class rdf:ID="Statement"/>
   </owl:allValuesFrom>
  </owl:Restriction>
 </rdfs:subClassOf>
 <rdfs:subClassOf>
  <owl:Restriction>
   <owl:allValuesFrom>
    <owl:Class rdf:ID="Variable"/>
   </owl:allValuesFrom>
   <owl:onProperty>
    <owl:ObjectProperty rdf:ID="manipula"/>
   </owl:onProperty>
  </owl:Restriction>
 </rdfs:subClassOf>
 <rdfs:subClassOf>
  <owl:Restriction>
   <owl:allValuesFrom>
    <owl:Class rdf:ID="Deprecated"/>
   </owl:allValuesFrom>
   <owl:onProperty>
    <owl:ObjectProperty rdf:ID="podeSer"/>
   </owl:onProperty>
  </owl:Restriction>
 </rdfs:subClassOf>
 <rdfs:subClassOf>
  <owl:Restriction>
   <owl:onProperty>
    <owl:ObjectProperty rdf:about="#podeSer"/>
   </owl:onProperty>
   <owl:allValuesFrom>
    <owl:Class rdf:ID="Private"/>
   </owl:allValuesFrom>
  </owl:Restriction>
 </rdfs:subClassOf>
 <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
 >funÃ§Ã£o</rdfs:label>
 <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
 >método</rdfs:label>
```

```
<rdfs:subClassOf>
 <owl:Restriction>
  <owl:onProperty>
   <owl:ObjectProperty rdf:about="#podeSer"/>
  </owl:onProperty>
  <owl:allValuesFrom>
   <owl:Class rdf:ID="Final"/>
  </owl:allValuesFrom>
 </owl:Restriction>
</rdfs:subClassOf>
<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>method</rdfs:label>
<rdfs:subClassOf>
 <owl:Restriction>
  <owl:allValuesFrom rdf:resource="#Public"/>
  <owl:onProperty>
   <owl:ObjectProperty rdf:about="#podeSer"/>
  </owl:onProperty>
 </owl:Restriction>
</rdfs:subClassOf>
<rdfs:subClassOf>
 <owl:Restriction>
  <owl:allValuesFrom>
   <owl:Class rdf:ID="Comment"/>
  </owl:allValuesFrom>
  <owl:onProperty>
   <owl:TransitiveProperty rdf:about="#contém"/>
  </owl:onProperty>
 </owl:Restriction>
</rdfs:subClassOf>
<rdfs:subClassOf>
 <owl:Class rdf:ID="ObjectModel"/>
</rdfs:subClassOf>
<rdfs:subClassOf>
 <owl:Restriction>
  <owl:allValuesFrom>
   <owl:Class rdf:ID="Data"/>
  </owl:allValuesFrom>
  <owl:onProperty>
   <owl:ObjectProperty rdf:about="#manipula"/>
  </owl:onProperty>
 </owl:Restriction>
</rdfs:subClassOf>
<rdfs:subClassOf>
 <owl:Restriction>
  <owl:allValuesFrom>
   <owl:Class rdf:ID="Protected"/>
  </owl:allValuesFrom>
  <owl:onProperty>
   <owl:ObjectProperty rdf:about="#podeSer"/>
```

```
    </owl:onProperty>
   </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
   <owl:Restriction>
    <owl:allValuesFrom rdf:resource="#Declaration"/>
    <owl:onProperty>
     <owl:TransitiveProperty rdf:about="#contém"/>
    </owl:onProperty>
   </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
   <owl:Restriction>
    <owl:onProperty>
     <owl:TransitiveProperty rdf:about="#contém"/>
    </owl:onProperty>
    <owl:allValuesFrom>
     <owl:Class rdf:about="#DataType"/>
    </owl:allValuesFrom>
   </owl:Restriction>
  </rdfs:subClassOf>
 </owl:Class>
 <owl:Class rdf:ID="Inner_Class">
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >inner class</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >inner</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >innerclass</rdfs:label>
  <rdfs:subClassOf>
   <owl:Class rdf:ID="Class"/>
  </rdfs:subClassOf>
 </owl:Class>
 <owl:Class rdf:ID="Iteration">
  <rdfs:subClassOf>
   <owl:Class rdf:about="#Statement"/>
  </rdfs:subClassOf>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >interaÃ§Ã£o</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >loop</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >iteration</rdfs:label>
 </owl:Class>
 <owl:Class rdf:about="#Operator">
  <rdfs:subClassOf rdf:resource="#Concept"/>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >operador</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >operator</rdfs:label>
```

```
</owl:Class>
<owl:Class rdf:about="#DataType">
 <rdfs:subClassOf rdf:resource="#Concept"/>
 <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
 >tipo de dado</rdfs:label>
 <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
 >data type</rdfs:label>
</owl:Class>
<owl:Class rdf:ID="ArithmeticExpression">
 <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
 >aritmethic expression</rdfs:label>
 <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
 >expressÃ£o aritmética</rdfs:label>
 <rdfs:subClassOf rdf:resource="#Concept"/>
</owl:Class>
<owl:Class rdf:ID="Jump">
 <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
 >Para colocar: break, return...</rdfs:comment>
 <rdfs:subClassOf>
  <owl:Class rdf:about="#Statement"/>
 </rdfs:subClassOf>
 <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
 >jump</rdfs:label>
 <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
 >salto</rdfs:label>
</owl:Class>
<owl:Class rdf:ID="Superclass">
 <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
 >super classe</rdfs:label>
 <rdfs:subClassOf>
  <owl:Restriction>
   <owl:allValuesFrom>
    <owl:Class rdf:ID="Subclass"/>
   </owl:allValuesFrom>
   <owl:onProperty>
    <owl:TransitiveProperty rdf:about="#contém"/>
   </owl:onProperty>
  </owl:Restriction>
 </rdfs:subClassOf>
 <rdfs:subClassOf>
  <owl:Class rdf:about="#Class"/>
 </rdfs:subClassOf>
 <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
 >superclass</rdfs:label>
 <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
 >super class</rdfs:label>
 <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
 >superclasse</rdfs:label>
</owl:Class>
<owl:Class rdf:ID="Object">
```

```
<rdfs:subClassOf>
 <owl:Restriction>
  <owl:onProperty>
   <owl:TransitiveProperty rdf:about="#contém"/>
  </owl:onProperty>
  <owl:allValuesFrom>
   <owl:Class rdf:ID="Field"/>
  </owl:allValuesFrom>
 </owl:Restriction>
</rdfs:subClassOf>
<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>instÃ¢ncia</rdfs:label>
<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>object</rdfs:label>
<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>instance</rdfs:label>
<rdfs:subClassOf>
 <owl:Restriction>
  <owl:allValuesFrom>
   <owl:Class rdf:ID="Interface"/>
  </owl:allValuesFrom>
  <owl:onProperty>
   <owl:ObjectProperty rdf:ID="implementa"/>
  </owl:onProperty>
 </owl:Restriction>
</rdfs:subClassOf>
<rdfs:subClassOf rdf:resource="#Concept"/>
<rdfs:subClassOf>
 <owl:Restriction>
  <owl:allValuesFrom>
   <owl:Class rdf:about="#Class"/>
  </owl:allValuesFrom>
  <owl:onProperty>
   <owl:ObjectProperty rdf:ID="eInstanciaDe"/>
  </owl:onProperty>
 </owl:Restriction>
</rdfs:subClassOf>
<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>objeto</rdfs:label>
<rdfs:subClassOf>
 <owl:Restriction>
  <owl:onProperty>
   <owl:ObjectProperty rdf:ID="eCriadoPor"/>
  </owl:onProperty>
  <owl:allValuesFrom rdf:resource="#Constructor"/>
 </owl:Restriction>
</rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="#Statement">
 <rdfs:subClassOf rdf:resource="#Concept"/>
```

```
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >statement</rdfs:label>
 </owl:Class>
 <owl:Class rdf:ID="DataStructure">
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >estrutura_de_dados</rdfs:label>
  <rdfs:subClassOf rdf:resource="#Concept"/>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >data_structure</rdfs:label>
 </owl:Class>
 <owl:Class rdf:ID="RegularExpression">
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >regular</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >expressÃ£o_regular</rdfs:label>
  <rdfs:subClassOf rdf:resource="#Concept"/>
 </owl:Class>
 <owl:Class rdf:ID="Byte">
  <rdfs:subClassOf rdf:resource="#Primitive"/>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >byte</rdfs:label>
 </owl:Class>
 <owl:Class rdf:about="#Data">
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >data</rdfs:label>
  <rdfs:subClassOf rdf:resource="#DataType"/>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >dado</rdfs:label>
 </owl:Class>
 <owl:Class rdf:ID="File">
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >arquivo</rdfs:label>
  <rdfs:subClassOf rdf:resource="#Concept"/>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >file</rdfs:label>
 </owl:Class>
 <owl:Class rdf:ID="Polimorfism">
  <rdfs:subClassOf rdf:resource="#Concept"/>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >polimorfismo</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >polimorfism</rdfs:label>
 </owl:Class>
 <owl:Class rdf:about="#Subclass">
  <rdfs:subClassOf>
   <owl:Class rdf:about="#Class"/>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
   <owl:Restriction>
    <owl:allValuesFrom rdf:resource="#Superclass"/>
```

```
    <owl:onProperty>
      <owl:ObjectProperty rdf:ID="herdaDe"/>
    </owl:onProperty>
  </owl:Restriction>
</rdfs:subClassOf>
<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>sub classe</rdfs:label>
<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>subclass</rdfs:label>
<rdfs:subClassOf>
  <owl:Restriction>
    <owl:onProperty>
      <owl:ObjectProperty rdf:ID="define"/>
    </owl:onProperty>
    <owl:allValuesFrom>
      <owl:Class rdf:about="#Field"/>
    </owl:allValuesFrom>
  </owl:Restriction>
</rdfs:subClassOf>
<rdfs:subClassOf>
  <owl:Restriction>
    <owl:onProperty>
      <owl:ObjectProperty rdf:ID="herda"/>
    </owl:onProperty>
    <owl:allValuesFrom rdf:resource="#Method"/>
  </owl:Restriction>
</rdfs:subClassOf>
<rdfs:subClassOf>
  <owl:Restriction>
    <owl:allValuesFrom>
      <owl:Class rdf:about="#Field"/>
    </owl:allValuesFrom>
    <owl:onProperty>
      <owl:ObjectProperty rdf:about="#herda"/>
    </owl:onProperty>
  </owl:Restriction>
</rdfs:subClassOf>
<rdfs:subClassOf>
  <owl:Restriction>
    <owl:allValuesFrom rdf:resource="#Method"/>
    <owl:onProperty>
      <owl:ObjectProperty rdf:ID="redefinir"/>
    </owl:onProperty>
  </owl:Restriction>
</rdfs:subClassOf>
<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>sub class</rdfs:label>
<rdfs:subClassOf>
  <owl:Restriction>
    <owl:onProperty>
```

```
      <owl:ObjectProperty rdf:about="#define"/>
     </owl:onProperty>
     <owl:allValuesFrom rdf:resource="#Method"/>
    </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:subClassOf>
    <owl:Restriction>
     <owl:onProperty>
      <owl:ObjectProperty rdf:about="#podeSer"/>
     </owl:onProperty>
     <owl:allValuesFrom rdf:resource="#Superclass"/>
    </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >subclasse</rdfs:label>
  </owl:Class>
  <owl:Class rdf:ID="Exception">
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >except</rdfs:label>
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >exceÃ§Ã£o</rdfs:label>
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >exception</rdfs:label>
   <rdfs:subClassOf rdf:resource="#Concept"/>
  </owl:Class>
  <owl:Class rdf:ID="ClassLibrary">
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >library</rdfs:label>
   <rdfs:subClassOf>
    <owl:Restriction>
     <owl:onProperty>
      <owl:TransitiveProperty rdf:about="#contém"/>
     </owl:onProperty>
     <owl:allValuesFrom>
      <owl:Class rdf:ID="Package"/>
     </owl:allValuesFrom>
    </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:subClassOf>
    <owl:Restriction>
     <owl:allValuesFrom>
      <owl:Class rdf:ID="API"/>
     </owl:allValuesFrom>
     <owl:onProperty>
      <owl:ObjectProperty rdf:about="#implementa"/>
     </owl:onProperty>
    </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >biblioteca</rdfs:label>
```

```
  <rdfs:subClassOf rdf:resource="#Concept"/>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >class library</rdfs:label>
 </owl:Class>
 <owl:Class rdf:ID="Integer">
  <rdfs:subClassOf rdf:resource="#Primitive"/>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >integer</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >inteiro</rdfs:label>
 </owl:Class>
 <owl:Class rdf:ID="String">
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >text</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >texto</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >string</rdfs:label>
  <rdfs:subClassOf>
   <owl:Class rdf:ID="ObjectReference"/>
  </rdfs:subClassOf>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >strings</rdfs:label>
 </owl:Class>
 <owl:Class rdf:about="#Comment">
  <rdfs:subClassOf rdf:resource="#Concept"/>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >comment</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >comentÃ¡rio</rdfs:label>
 </owl:Class>
 <owl:Class rdf:about="#Field">
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >campo</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >property</rdfs:label>
  <rdfs:subClassOf>
   <owl:Restriction>
    <owl:allValuesFrom rdf:resource="#DataType"/>
    <owl:onProperty>
     <owl:TransitiveProperty rdf:about="#contém"/>
    </owl:onProperty>
   </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >atributo</rdfs:label>
  <rdfs:subClassOf rdf:resource="#Concept"/>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >propriedade</rdfs:label>
  <rdfs:subClassOf>
```

```
    <owl:Restriction>
      <owl:allValuesFrom rdf:resource="#Data"/>
      <owl:onProperty>
        <owl:TransitiveProperty rdf:about="#contém"/>
      </owl:onProperty>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="#Class">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:allValuesFrom rdf:resource="#Field"/>
      <owl:onProperty>
        <owl:TransitiveProperty rdf:about="#contém"/>
      </owl:onProperty>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >class</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >classe</rdfs:label>
  <rdfs:subClassOf>
    <owl:Class rdf:about="#ObjectReference"/>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:TransitiveProperty rdf:about="#contém"/>
      </owl:onProperty>
      <owl:allValuesFrom rdf:resource="#Method"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:allValuesFrom rdf:resource="#Inner_Class"/>
      <owl:onProperty>
        <owl:TransitiveProperty rdf:about="#contém"/>
      </owl:onProperty>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:allValuesFrom rdf:resource="#Object"/>
      <owl:onProperty>
        <owl:InverseFunctionalProperty rdf:ID="instancia"/>
      </owl:onProperty>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
```

```
      <owl:onProperty>
        <owl:ObjectProperty rdf:about="#implementa"/>
      </owl:onProperty>
      <owl:allValuesFrom>
        <owl:Class rdf:about="#Interface"/>
      </owl:allValuesFrom>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="#API">
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >api</rdfs:label>
  <rdfs:subClassOf rdf:resource="#Concept"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:TransitiveProperty rdf:about="#contém"/>
      </owl:onProperty>
      <owl:allValuesFrom>
        <owl:Class rdf:about="#Interface"/>
      </owl:allValuesFrom>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >application program interface</rdfs:label>
</owl:Class>
<owl:Class rdf:about="#Final">
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >final</rdfs:label>
  <rdfs:subClassOf rdf:resource="#Concept"/>
</owl:Class>
<owl:Class rdf:ID="Qualifier">
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >qualifier</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >qualificador</rdfs:label>
  <rdfs:subClassOf rdf:resource="#Concept"/>
  <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >pra colocar static por exemplo</rdfs:comment>
</owl:Class>
<owl:Class rdf:ID="Double">
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >double</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >decimais</rdfs:label>
  <rdfs:subClassOf rdf:resource="#Primitive"/>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >decimal</rdfs:label>
</owl:Class>
<owl:Class rdf:ID="Pattern">
```

```
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >padrÃ£o</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >pattern</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >padrÃµes</rdfs:label>
  <rdfs:subClassOf rdf:resource="#Concept"/>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >patterns</rdfs:label>
 </owl:Class>
 <owl:Class rdf:ID="Socket">
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >socket</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >sockets</rdfs:label>
  <rdfs:subClassOf rdf:resource="#Concept"/>
 </owl:Class>
 <owl:Class rdf:ID="AbstractClass">
  <rdfs:subClassOf rdf:resource="#Superclass"/>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >classe abstrata</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >abstract class</rdfs:label>
 </owl:Class>
 <owl:Class rdf:ID="ExceptionTreatment">
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >exception treatment</rdfs:label>
  <rdfs:subClassOf rdf:resource="#Concept"/>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >tratamento de exceÃ§Ã£o</rdfs:label>
 </owl:Class>
 <owl:Class rdf:ID="RelationalOperator">
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >operador relacional</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >relational operator</rdfs:label>
  <rdfs:subClassOf rdf:resource="#Operator"/>
 </owl:Class>
 <owl:Class rdf:ID="Character">
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >character</rdfs:label>
  <rdfs:subClassOf rdf:resource="#Primitive"/>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >caractere</rdfs:label>
 </owl:Class>
 <owl:Class rdf:ID="Constant">
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >constante</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >constant</rdfs:label>
```

```
  <rdfs:subClassOf rdf:resource="#Concept"/>
</owl:Class>
<owl:Class rdf:ID="Thread">
  <rdfs:subClassOf rdf:resource="#Concept"/>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >thread</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >threads</rdfs:label>
</owl:Class>
<owl:Class rdf:about="#Variable">
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >variable</rdfs:label>
  <rdfs:subClassOf rdf:resource="#Concept"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:TransitiveProperty rdf:about="#contém"/>
      </owl:onProperty>
      <owl:allValuesFrom rdf:resource="#DataType"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >variÃ¡vel</rdfs:label>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:allValuesFrom rdf:resource="#Data"/>
      <owl:onProperty>
        <owl:TransitiveProperty rdf:about="#contém"/>
      </owl:onProperty>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="#ObjectReference">
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >object reference</rdfs:label>
  <rdfs:subClassOf rdf:resource="#DataType"/>
</owl:Class>
<owl:Class rdf:ID="Scope">
  <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >Pode ser instanciado em Público, protegido e privado</rdfs:comment>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >scope</rdfs:label>
  <rdfs:subClassOf rdf:resource="#Concept"/>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >escopo</rdfs:label>
</owl:Class>
<owl:Class rdf:ID="DateTime">
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >date</rdfs:label>
  <rdfs:subClassOf rdf:resource="#ObjectReference"/>
```

<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>datetime</rdfs:label>
<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>timestamp</rdfs:label>
<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>data</rdfs:label>
<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>hora</rdfs:label>
<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>tempo</rdfs:label>
</owl:Class>
<owl:Class rdf:ID="Array">
<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>array</rdfs:label>
<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>matriz</rdfs:label>
<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>matrix</rdfs:label>
<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>vector</rdfs:label>
<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>vetor</rdfs:label>
<rdfs:subClassOf rdf:resource="#Primitive"/>
</owl:Class>
<owl:Class rdf:about="#Heritage">
<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>heritage</rdfs:label>
<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>heranÃ§a</rdfs:label>
<rdfs:subClassOf rdf:resource="#Concept"/>
<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>inherit</rdfs:label>
</owl:Class>
<owl:Class rdf:about="#Private">
<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>privada</rdfs:label>
<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>privados</rdfs:label>
<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>private</rdfs:label>
<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>privado</rdfs:label>
<rdfs:subClassOf rdf:resource="#Concept"/>
<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>privadas</rdfs:label>
</owl:Class>
<owl:Class rdf:about="#ObjectModel">
<rdfs:subClassOf rdf:resource="#Concept"/>
<rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
>object model</rdfs:label>

```
</owl:Class>
<owl:Class rdf:about="#Deprecated">
 <rdfs:subClassOf rdf:resource="#Concept"/>
 <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
 >deprecated</rdfs:label>
</owl:Class>
<owl:Class rdf:about="#Interface">
 <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
 >interface</rdfs:label>
 <rdfs:subClassOf rdf:resource="#AbstractClass"/>
</owl:Class>
<owl:Class rdf:ID="Collection">
 <rdfs:subClassOf rdf:resource="#DataStructure"/>
 <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
 >coleÃ§Ã£o</rdfs:label>
 <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
 >collections</rdfs:label>
 <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
 >collection</rdfs:label>
 <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
 >coleÃ§Ãµes</rdfs:label>
</owl:Class>
<owl:Class rdf:about="#Package">
 <rdfs:subClassOf>
  <owl:Restriction>
   <owl:allValuesFrom rdf:resource="#Interface"/>
   <owl:onProperty>
    <owl:TransitiveProperty rdf:about="#contém"/>
   </owl:onProperty>
  </owl:Restriction>
 </rdfs:subClassOf>
 <rdfs:subClassOf>
  <owl:Restriction>
   <owl:allValuesFrom rdf:resource="#API"/>
   <owl:onProperty>
    <owl:ObjectProperty rdf:about="#implementa"/>
   </owl:onProperty>
  </owl:Restriction>
 </rdfs:subClassOf>
 <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
 >pacote</rdfs:label>
 <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
 >package</rdfs:label>
 <rdfs:subClassOf rdf:resource="#Concept"/>
 <rdfs:subClassOf>
  <owl:Restriction>
   <owl:onProperty>
    <owl:TransitiveProperty rdf:about="#contém"/>
   </owl:onProperty>
   <owl:allValuesFrom rdf:resource="#Class"/>
```

```
    </owl:Restriction>
   </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Conditional">
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >conditional</rdfs:label>
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >condicional</rdfs:label>
   <rdfs:subClassOf rdf:resource="#Statement"/>
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >desvio_condicional</rdfs:label>
  </owl:Class>
  <owl:Class rdf:about="#Protected">
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >protected</rdfs:label>
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >protegido</rdfs:label>
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >protegidas</rdfs:label>
   <rdfs:subClassOf rdf:resource="#Concept"/>
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >protegidos</rdfs:label>
  </owl:Class>
  <owl:ObjectProperty rdf:about="#herda">
   <rdfs:domain rdf:resource="#Subclass"/>
   <rdfs:range>
    <owl:Class>
     <owl:unionOf rdf:parseType="Collection">
      <owl:Class rdf:about="#Method"/>
      <owl:Class rdf:about="#Field"/>
     </owl:unionOf>
    </owl:Class>
   </rdfs:range>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:about="#eInstanciaDe">
   <rdfs:range rdf:resource="#Class"/>
   <rdfs:domain rdf:resource="#Object"/>
   <owl:equivalentProperty>
    <owl:InverseFunctionalProperty rdf:about="#instancia"/>
   </owl:equivalentProperty>
   <owl:inverseOf>
    <owl:InverseFunctionalProperty rdf:about="#instancia"/>
   </owl:inverseOf>
   <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#InverseFunctionalProperty"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:about="#define">
   <rdfs:range>
    <owl:Class>
     <owl:unionOf rdf:parseType="Collection">
```

```
      <owl:Class rdf:about="#Method"/>
      <owl:Class rdf:about="#Field"/>
     </owl:unionOf>
    </owl:Class>
   </rdfs:range>
   <rdfs:domain rdf:resource="#Subclass"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#eCriadoPor">
   <rdfs:domain rdf:resource="#Object"/>
   <rdfs:range rdf:resource="#Constructor"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#implementa">
   <rdfs:domain>
    <owl:Class>
     <owl:unionOf rdf:parseType="Collection">
      <owl:Class rdf:about="#ClassLibrary"/>
      <owl:Class rdf:about="#Package"/>
      <owl:Class rdf:about="#Class"/>
      <owl:Class rdf:about="#Object"/>
     </owl:unionOf>
    </owl:Class>
   </rdfs:domain>
   <rdfs:range>
    <owl:Class>
     <owl:unionOf rdf:parseType="Collection">
      <owl:Class rdf:about="#API"/>
      <owl:Class rdf:about="#Interface"/>
     </owl:unionOf>
    </owl:Class>
   </rdfs:range>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#herdaDe">
   <rdfs:domain rdf:resource="#Subclass"/>
   <rdfs:range rdf:resource="#Superclass"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#manipula">
   <rdfs:range>
    <owl:Class>
     <owl:unionOf rdf:parseType="Collection">
      <owl:Class rdf:about="#Data"/>
      <owl:Class rdf:about="#Variable"/>
     </owl:unionOf>
    </owl:Class>
   </rdfs:range>
   <rdfs:domain rdf:resource="#Method"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#redefinir">
   <rdfs:domain rdf:resource="#Subclass"/>
   <rdfs:range rdf:resource="#Method"/>
</owl:ObjectProperty>
```

```
<owl:ObjectProperty rdf:about="#podeSer">
 <rdfs:range>
  <owl:Class>
   <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#Superclass"/>
    <owl:Class rdf:about="#Deprecated"/>
    <owl:Class rdf:about="#Final"/>
    <owl:Class rdf:about="#Public"/>
    <owl:Class rdf:about="#Private"/>
    <owl:Class rdf:about="#Protected"/>
   </owl:unionOf>
  </owl:Class>
 </rdfs:range>
 <rdfs:domain>
  <owl:Class>
   <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#Subclass"/>
    <owl:Class rdf:about="#Method"/>
   </owl:unionOf>
  </owl:Class>
 </rdfs:domain>
</owl:ObjectProperty>
<owl:DatatypeProperty rdf:ID="eSerializavel">
 <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#boolean"/>
 <rdfs:domain rdf:resource="#Object"/>
</owl:DatatypeProperty>
<owl:TransitiveProperty rdf:about="#contém">
 <rdfs:domain>
  <owl:Class>
   <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#Class"/>
    <owl:Class rdf:about="#OOLanguage"/>
    <owl:Class rdf:about="#ClassLibrary"/>
    <owl:Class rdf:about="#Package"/>
    <owl:Class rdf:about="#API"/>
    <owl:Class rdf:about="#Object"/>
    <owl:Class rdf:about="#Method"/>
    <owl:Class rdf:about="#Variable"/>
    <owl:Class rdf:about="#Field"/>
   </owl:unionOf>
  </owl:Class>
 </rdfs:domain>
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
 <rdfs:range>
  <owl:Class>
   <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#Field"/>
    <owl:Class rdf:about="#Heritage"/>
    <owl:Class rdf:about="#Method"/>
    <owl:Class rdf:about="#Package"/>
```

```
        <owl:Class rdf:about="#Statement"/>
        <owl:Class rdf:about="#Declaration"/>
        <owl:Class rdf:about="#Comment"/>
        <owl:Class rdf:about="#DataType"/>
      </owl:unionOf>
    </owl:Class>
  </rdfs:range>
</owl:TransitiveProperty>
<owl:InverseFunctionalProperty rdf:about="#instancia">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
  <owl:equivalentProperty rdf:resource="#eInstanciaDe"/>
  <rdfs:range rdf:resource="#Object"/>
  <owl:inverseOf rdf:resource="#eInstanciaDe"/>
  <owl:sameAs rdf:resource="#eInstanciaDe"/>
  <rdfs:domain rdf:resource="#Class"/>
</owl:InverseFunctionalProperty>
```