Universidade Federal do Maranhão
Centro de Ciências Exatas e Tecnologia – CCET
Departamento de Engenharia de Eletricidade
Programa de Pós-Graduação em Engenharia de Eletricidade

UM MODELO DE SISTEMA DE FILTRAGEM HÍBRIDA PARA UM AMBIENTE COLABORATIVO DE ENSINO APRENDIZAGEM

André Luis Silva dos Santos

São Luís
2008

André Luis Silva dos Santos

UM MODELO DE SISTEMA DE FILTRAGEM HÍBRIDA PARA UM AMBIENTE COLABORATIVO DE ENSINO APRENDIZAGEM

Dissertação submetida à Coordenação do Programa de Pós-Graduação em Engenharia de Eletricidade da Universidade Federal do Maranhão como parte dos requisitos para a obtenção do título de Mestre em Engenharia de Eletricidade, área de concentração: Ciência da Computação.

Orientador: Prof. Ph.D. Zair Abdelouahab
Orientador: Prof. Dr. Sofiane Labidi

São Luís
2008

Santos, André Luis Silva dos

Um modelo de sistema de filtragem híbrida para um ambiente colaborativo de ensino aprendizagem / André Luis Silva dos Santos. – São Luís, 2008.

131 f.

Impresso por computador (fotocópia).

Orientadores: Zair Abdelouahab, Sofiane Labidi.

Dissertação (Mestrado) – Universidade Federal do Maranhão, Programa de Pós-Graduação em Engenharia de Eletricidade, 2008.

1. Ensino-Aplicação – NETCLASS 2. Informação – Sistemas de Filtragem 3. Informação – Técnicas de Filtragem 4. Ensino-aprendizagem – Ambientes Colaborativos I. Título

CDU 004.891:37

UM MODELO DE SISTEMA DE FILTRAGEM HIBRIDA PARA UM AMBIENTE COLABORATIVO DE ENSINO APRENDIZAGEM

André Luis Silva dos Santos

Dissertação aprovada em15 de fevereiro de 2008.

Prof. Zair Abdelouahab, Ph.D.
(Orientador)

Prof. Pedro Porfírio Muniz Farias, Dr.
(Membro da Banca Examinadora)

Prof. Sofiane Labidi, Dr.
(Membro da Banca Examinadora)

Prof. Denivaldo Cicero Pavão Lopes, Dr.
(Membro da Banca Examinadora)

A Deus.

Aos meus pais Francisco e Telma Santos.

À minha esposa Simone.

Às minhas filhas Gabryela e Clara.

Aos meus irmãos e sobrinhos.

“O Senhor é meu pastor; nada me faltará...Ainda que eu ande pelo vale da sombra da morte, não temerei mal nenhum, porque tu estás comigo”

Salmo 23

AGRADECIMENTOS

Inicialmente, agradeço a Deus por todas as provisões que tornaram esta dissertação uma realidade.

A minha família pela compreensão nas minhas ausências em nossas reuniões familiares, principalmente minha esposa e filhas que compartilharam comigo de todos os momentos durante esses anos.

Aos meus orientadores, Professor Zair Abdelouahab e Sofiane Labidi, que me acolheram e confiaram em mim, pela orientação e por toda paciência e dedicação despendidas para realização deste trabalho.

Aos meus queridos pais, Francisco e Telma Santos, pelo esforço realizado para me proporcionar toda a infra-estrutura necessária para a realização desta conquista.

Aos meus amigos do Mestrado, por compartilharem comigo a felicidade desta conquista.

Aos meus colegas do CEFET-MA em especial Antonio Luna, Omar Andres, Santiago Filho, Rafael Lopes, peja ajuda e palavras de apoio.

Ao amigo Walker Cleisson, pela ajuda incomensurável, e a sua esposa, pela compreensão.

A todos os professores do curso de Mestrado e de Ciência da Computação, que não mediram esforços para transmitir aos seus alunos o verdadeiro conhecimento.

Enfim, agradeço a todos que de uma alguma forma contribuíram para a realização deste trabalho e me ajudaram no decorrer do curso.

RESUMO

A Web é uma excelente fonte de informação, mas um dos problemas que surgem com a grande disseminação de informações é a dificuldade de se obter informação relevante em tempo hábil e de forma precisa. Mecanismos que auxiliem o usuário na recuperação de informações tais como o Google.com, Altavista e Cadê, muitas das vezes retornam uma grande quantidade de conteúdo, sem garantir uma boa efetividade de recuperação, com excesso de informações recuperadas ou com informações irrelevantes. Os Sistemas de Filtragem de Informação surgem como alternativa de auxílio aos usuários na busca de informações relevantes. Este trabalho propõe a criação de um modelo de sistema de filtragem híbrido de informação baseados nos métodos: Filtragem Baseada em Conteúdo e Filtragem Colaborativa. O modelo proposto é aplicado a um ambiente colaborativo de ensino-aprendizagem, o NetClass, e foi desenvolvido com a metodologia PASSI. Um estudo de caso feito com alunos do CEFET-MA também é descrito.

Palavras-chave: Sistemas de Filtragem de Informação, Técnicas de Filtragem de Informação, Ambientes Colaborativos de Ensino-Aprendizagem

ABSTRACT

Nowadays, the World Wide Web (WWW) is an excellent source of information. However, open issues carry on. It´s difficult obtain relevant information in short time. Moreover, there is no accuracy for retrieving this information. Servers such as Google, Altavista and Cadê, can retrieve a huge amount of information. Nonetheless, the retrieved information could be not relevant. The information filtering systems arise to aim users in the searching for relevant information. This work proposes a hybrid model of filtering information based on content-based filtering and collaborative filtering. This model has been used into a collaborative learning system named NetClass and it was developed using the PASSI methodology. A case study done with CEFET´s students is presented as well.

Keywords: Filtering Information Systems, Techniques of Filtering Information, Collaborative Learning Environment.

LISTA DE ABREVIATURAS DE SÍMBOLOS

ACAC	–	Aprendizagem Colaborativa Apoiada por Computador
ACLMessage	–	Agent Communication Language
CEFET-MA	–	Centro Federal de Educação Tecnológica do Maranhão
FBC	–	Filtragem Baseada em Conteúdo
FC	–	Filtragem Colaborativa
FI	–	Filtragem de Informação
FIPA	–	Foudation for Inetlligent Physical Agents
IDE	–	Integrated Development Environment
IDF	–	Inverse-Document-Frequency
JADE	–	Java Agent Development Framework
KNN	–	K Nearest Neighbors
MAFIS	–	Sistema Multiagente de Filtragem de Informação
MAS	–	Multi-Agent Systems
PASSI	–	Process for Agent Societies Specification and Implementation
PTK	–	PASSI ToolKIt
RI	–	Recuperação de Informação
SFI	–	Sistemas de Filtragem de Informação
SIG	–	Sistema de Informação Geográfica
SMA	–	Sistema Multiagente
SRI	–	Sistemas de Filtragem de Informação
STI	–	Sistemas Tutores Inteligentes
TF	–	Term-Frequency
TF-IDF	–	Term-Fequency-Inverse-Document-Frequency
UFMA	–	Universidade Federal do Maranhão
UML	–	Unified Modeling Language

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

1.1 Contexto e Problemática

A presente dissertação de mestrado enfoca a área de Sistemas Tutores Inteligentes (STI) e Filtragem de Informação utilizando sistemas multiagentes, e faz parte do projeto NetClass [32] da Universidade Federal do Maranhão (UFMA).

O NetClass é um projeto que une as idéias que fundamentam os STIs [30] ao paradigma de Aprendizagem Colaborativa, definindo um Ambiente Colaborativo de Ensino-Aprendizagem. Este ambiente integra alunos, professores e um sistema computacional, favorecendo a aprendizagem do aluno (individualmente ou em grupo) através da resolução de problemas, e com a assistência do sistema tutor e dos professores, adaptada às necessidades do aluno.

A Filtragem de Informação é o nome usado para descrever uma variedade de processos envolvendo o envio de informação para pessoas que necessitam dela. A filtragem é baseada na descrição das preferências de indivíduos ou grupos.

O problema principal abordado neste trabalho diz respeito a grande quantidade de informações contidas em base de dados que dificultam o acesso dos aprendizes à informação que realmente é relevante para o seu aprendizado, aumentando o tempo perdido na busca do mesmo.

1.2 Relevância e Motivação

De maneira geral, Sistemas de Filtragem de Informação merecem ser profundamente pesquisados por conta de sua capacidade promissora no desenvolvimento de aplicações que venham atender de maneira efetiva aos interesses de um usuário em qualquer área de conhecimento.

Tem-se como diferencial neste sistema o reaproveitamento do trabalho anteriormente realizado na Recuperação de Informação desenvolvido por Oliveira [49], do qual faz-se uso de agentes de recuperação para compor o

Sistema de Filtragem e na aplicação de um uma abordagem hibrida de filtragem num ambiente de ensino colaborativo de ensino-aprendizagem.

De maneira particular, pesquisar sobre Filtragem de Informação é interesse do Grupo de Pesquisa do Laboratório de Sistemas Inteligentes (LSI), porque representa a continuidade dos trabalhos que vêm sendo desempenhados no Ambiente NetClass para a entrega de informações relevantes aos seus usuários.

Por fim, o que motiva a realização deste trabalho, tornando-o diferente de tantos outros, é o fato de utilizar uma abordagem hibrida de filtragem de informação aplicada a um ambiente colaborativo de ensino-aprendizagem.

1.3 Objetivos

O objetivo geral deste trabalho é desenvolver uma aplicação que, através do emprego de técnicas de filtragem de informação, seja capaz de filtrar informações aos usuários do NetClass, um ambiente colaborativo de ensino aprendizagem.

No sentido de alcançar tal objetivo geral, tem-se os seguintes objetivos específicos: 1) descrever os conceitos inerentes à Filtragem de Informação; 2) estabelecer direções gerais sobre os algoritmos de Filtragem; 3) especificar a estrutura dos agentes responsáveis, no NetClass, pela filtragem de informações; e 4) implementar um protótipo de Agentes de Filtragem que podem ser utilizados no NetClass.

1.4 Estruturação

Esta dissertação está organizada em oito capítulos.

No Capítulo 2, contém uma revisão bibliográfica sobre Filtragem e Recuperação de Informação, contemplando os principais conceitos, semelhanças e diferenças entre as mesmas.

No Capítulo 3, discorre-se sobre as principais técnicas de filtragem de informação utilizadas em sistemas de filtragem.

No Capítulo 4, apresentam-se os algoritmos mais utilizados na filtragem de informação, mostrando suas especificidades e utilização.

Em seguida, no Capítulo 5, apresenta-se o Ambiente NetClass, destacando-se o conceito de Aprendizagem Colaborativa e mostra-se a arquitetura multiagente do NetClass.

O Capítulo 6 contém a especificação dos Agentes de Filtragem do NetClass encontra-se. Onde apresentam-se a arquitetura do sistema, as classes que compõem esses agentes e as responsabilidades de cada uma delas, face ao processo de filtragem de informações.

A partir dessa especificação, no Capítulo 7, apresenta-se um protótipo de implementação dos Agentes de Filtragem, utilizando-se a metodologia PASSI, com a finalidade de mostrar a viabilidade do uso de agentes na filtragem de informações e sua inserção no NetClass, conforme descrito no decorrer deste trabalho mostrando os testes realizados.

Finalmente, no Capítulo 8, apresentam-se as conclusões deste trabalho e perspectivas de trabalhos futuros.

2 FILTRAGEM DE INFORMAÇÃO

2.1 Considerações Iniciais

A Web é uma excelente fonte de informação, mas um dos problemas que surgem com a grande disseminação de informações é a dificuldade de se obter informação relevante em tempo hábil e de forma precisa. A falta de padronização, classificação e filtragem dos conteúdos da Web contribui para a formação deste cenário [1].

Motores de Busca que auxiliem o usuário na recuperação de informações tais como o Google.com, Altavista e Cadê, muitas vezes retornam uma grande quantidade de conteúdo, sem garantir uma boa efetividade de recuperação, com excesso de informações recuperadas ou com informações irrelevantes.

Os Sistemas de Filtragem de Informação surgem como alternativa de auxílio aos usuários na busca de informações, com filtragem de itens que serão apreciados pelos usuários. Estes sistemas de filtragem, como o próprio nome sugere, fornecem informações personalizadas de acordo com a descrição das preferências (i.e. perfil) de um usuário ou grupos de usuários de forma explícita (e.g. preenchendo formulários) ou implícita (e.g. navegação, *cookies* ou sessões).

Tais sistemas possuem algoritmos que procuram otimizar a filtragem fazendo um "*matching*" entre usuários ou entre usuário e itens para filtrar informações pontualmente e que supõe-se serem relevantes.

Os principais mecanismos utilizados nos sistemas de filtragem integram várias técnicas de recuperação e filtragem de informação. Para [3], habitualmente, os sistemas de filtragem classificam-se em três categorias: Baseada em Conteúdo, Colaborativa e Híbrida. As categorias levam em consideração a maneira com a qual estas filtragens são realizadas.

Conforme Adomavicius [1] e Balabanovic [3], na Filtragem Baseada em Conteúdo o conteúdo de itens selecionados por um usuário no passado e enviados é analisado, dessa forma, itens com conteúdos similares. Em outras

palavras, os itens são basicamente documentos de texto comparados com perfis de usuários que na realidade representam suas preferências.

A Filtragem Colaborativa apóia-se na similaridade entre usuários para gerar a filtragem e não utiliza o conteúdo de itens porque se baseia na troca empírica de avaliações realizadas por outros usuários com interesses comuns [10].

De acordo com Herlocker [23], a Filtragem Híbrida consiste na combinação de diversos métodos utilizados tanto na Filtragem Baseada em Conteúdo quanto na Colaborativa.

De posse de tais abordagens os Sistemas de Filtragem proporcionam melhorias, no que tange o acesso às informações em longo prazo, sugerindo informações personalizadas de acordo com os interesses dos usuários.

Adomavicius e Tuzhilin [1] esclarecem que além de integrarem tradicionalmente técnicas de recuperação e filtragem de informação, outros métodos também podem ser utilizados e dentre eles várias técnicas de aprendizagem de máquina (e.g. agrupamento, árvores de decisão e redes neurais artificiais).

Os sub-tópicos seguintes detalham melhor os conceitos de Recuperação de Informação e Filtragem de Informação, de fundamental importância para este trabalho.

2.2 Recuperação e Filtragem de Informação

Recuperação e Filtragem de informações são processos de busca de informações relevantes em grandes coleções de dados digitais não estruturados ou semi-estruturados como textos, imagens, musicas, dados de modo geral.

Neste subtópico, procura-se fazer uma análise entre estas duas áreas do conhecimento consideradas por alguns como dois lados da mesma moeda, ou seja, complementares. Inicialmente, mostram-se as características

da recuperação de informação e da filtragem de informação para então elucidar suas principais semelhanças e diferenças.

2.2.1 Recuperação de Informação

Recuperação de Informação (RI) é uma subárea da Ciência da Computação que estuda formas de armazenamento e recuperação automática de documentos, geralmente textos [22]. Um Sistema de Recuperação de Informação (SRI) pode ser estruturado conforme a Figura 1.

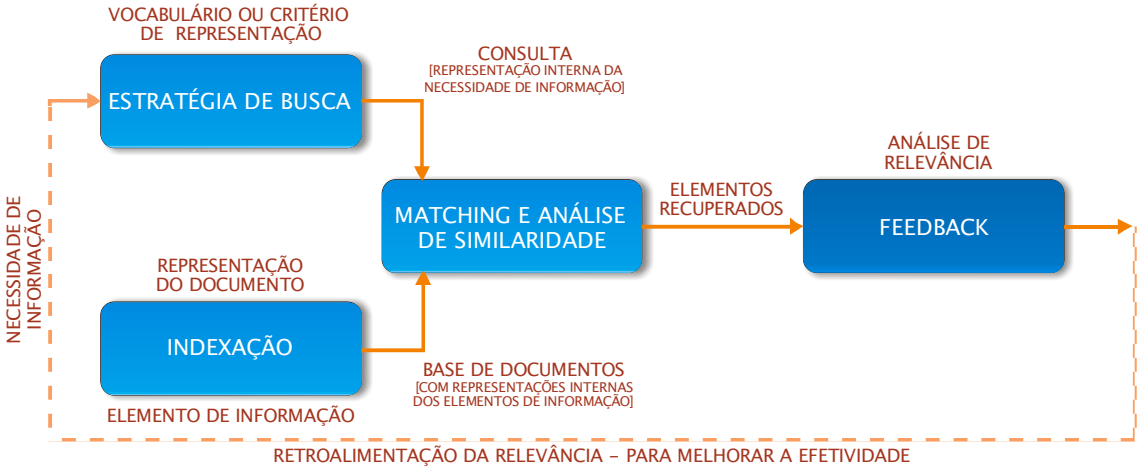

Figura 1 - Modelo Geral de Recuperação de Informação. Adaptado de Belkin [4]

O objetivo principal de um SRI é satisfazer as necessidades de informação do usuário, geralmente expressas em uma consulta ou pedido. O sistema responderá com uma lista (que pode estar vazia) de itens de informação extraídos de uma fonte de informação.

Os componentes do sistema incluem documentos, necessidades do usuário, gera a consulta formulada, e finalmente o processo de recuperação que, a partir das estruturas de dados e da consulta formulada, recupera uma lista de documentos considerados relevantes.

O processo de especificação da consulta geralmente é uma tarefa difícil. Há freqüentemente uma distância semântica entre a real necessidade do usuário e o que ele expressa na consulta formulada. Essa distância é gerada pelo limitado conhecimento do usuário sobre o universo de pesquisa e pelo formalismo da linguagem de consulta.

O processo de recuperação consiste na geração de uma lista de documentos recuperados para responder a consulta formulada pelo usuário. Os índices construídos para uma coleção de documentos são usados para acelerar esta tarefa. Além disso, a lista de documentos recuperados é classificada em ordem decrescente com um grau de similaridade entre o documento e a consulta [47].

Um Sistema de Recuperação de Informação (SRI) classifica os documentos recuperados para cada consulta, de acordo com uma ordem de relevância gerando um vetor resultado. Avalia-se o SRI através da comparação das respostas geradas por este sistema e o conjunto ideal de respostas. Para isso, o vetor resultado é examinado e comparado com o conjunto ideal, obtendo-se dois índices de avaliação: precisão e *recall*.

Precisão é a fração dos documentos já examinados que são relevantes, e *recall* é a fração dos documentos relevantes observada dentre os documentos examinados. A avaliação do modelo de um SRI pode ser observada por um gráfico com as médias precisão x *recall*.

Ao contrário dos Sistemas de Gerenciamento de Banco de Dados (SGDB), nos SRIs [59] [63] é freqüentemente difícil formular requisições de informação precisas. Além disso, a informação recuperada pode incluir itens que podem ou não coincidir exatamente com a informação requisitada.

A consulta, que deve ser expressa em uma linguagem entendida pelo sistema, é uma representação da necessidade de informação. Devido a dificuldade de especificar esta necessidade, a consulta em um SRI é sempre considerada como aproximada e imperfeita [4].

Outro ponto importante são os itens de informação que o usuário do SRI eventualmente acessará. Têm-se os produtores ou autores dos textos; os agrupamentos de textos dentro de coleções; a representação de textos; e, a classificação destas representações em um índice. O processo de representação de textos em uma forma mais adequada para ser processado pelo computador é chamado indexação. Uma representação típica consistiria, por exemplo, de um vetor de termos ou palavras chaves.

A comparação de uma consulta e o índice de itens de informação conduz para a seleção e recuperação de, possivelmente, um conjunto de textos considerados relevantes. Estes textos recuperados são então usados, avaliados, sendo que ao final do processo, o usuário deixará o SRI, ou a avaliação conduzirá a alguma modificação na consulta, ou na necessidade de informação, ou nos índices.

2.2.2 Filtragem de Informação

A Filtragem de Informação [4] é o nome usado para descrever uma variedade de processos envolvendo o envio de informação para pessoas que necessitam dela. A filtragem pode ser baseada na descrição das preferências de indivíduos ou grupos, denominado de perfil.

Os perfis podem ser adquiridos de forma explícita, fornecidos pelos usuários, ou implícita, através da captura automática do comportamento do usuário, utilizando, por exemplo, mineração de uso [38], aprendendo o perfil ou o comportamento do usuário.

Sistemas de Filtragem de Informação (SFI) lidam com grandes seqüências de documentos, normalmente distribuídos a partir de fontes remotas. Algumas vezes os SFIs podem ser vistos como roteadores de documentos (*document routing*), onde, de acordo com o perfil dos usuários são encaminhados itens.

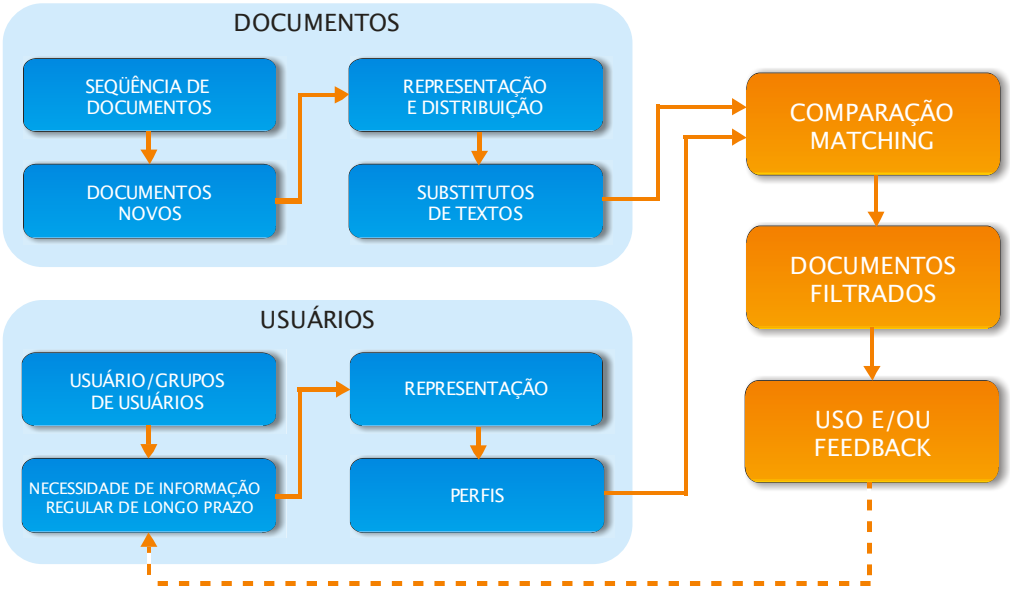

Figura 2 - Modelo Geral de Filtragem de Informação. Adaptado de Motta [46]

O sistema mantém os perfis de usuários que descrevem seus interesses a longo prazo. O perfil deve descrever o que o usuário gosta ou não. Os documentos que não se adequem ao perfil do usuário são removidos das seqüências que chegam. Como resultado, o usuário só vê o que é deixado na seqüência depois que os documentos inadequados foram removidos – um filtro de e-mails, por exemplo, remove “lixo” dos e-mails. A Figura 2 mostra um modelo geral de como é feita a Filtragem de Informação (FI).

O primeiro passo no uso de um SFI é criar um perfil ou *profile*. Um perfil representa as necessidades de informação de um usuário ou grupo de usuários, que se percebe estável por um longo período de tempo, ou seja, um perfil representa uma necessidade de informação que não se altera continuamente. Sempre que um novo documento é recebido na seqüência de dados, o sistema gera um modelo deste, representa-o e o compara com cada perfil armazenado no sistema. Se o documento coincide com o perfil, este documento será roteado para o usuário correspondente. O usuário pode então usar os documentos recebidos e/ou fornecer um retorno (*feedback*). O retorno fornecido pode levar a modificações no perfil e/ou na necessidade de informação.

Os métodos descritos para Filtragem de Informação são também utilizados para Recuperação de Informação (RI). A maior parte desses métodos foi inicialmente desenvolvida visando RI. Com o advento da FI, os métodos de RI foram adaptados para se adequar às necessidades da FI. Além disso, mais pesquisas surgem nesta área e isto resultou no desenvolvimento de novos métodos, os quais são correntemente usados em ambas as áreas [1].

Neste modelo, a filtragem de informação começa com pessoas (os usuários do sistema de filtragem) que têm metas ou desejos periódicos, a longo-prazo, relativamente estáveis (por exemplo, realizar uma tarefa de trabalho). Grupos, bem como, indivíduos, podem ser caracterizados por tais metas. Estas, então, conduzem a interesses regulares de informação (por exemplo, manter-se atualizado sobre um determinado assunto) que pode mudar lentamente toda vez que condições, metas e conhecimento mudam. Tais interesses de informações conduzem as pessoas a envolver-se com formas relativamente passivas de comportamento de buscar informação, na

medida em que textos são recuperados e trazidos para sua atenção. Isto é realizado pela representação de interesses de informação como perfis. Tais perfis são geralmente construídos com boas especificações de necessidades de informação mais ou menos permanentes [46].

No lado superior da Figura 2, o foco são os documentos. De cada novo documento distribuído uma representação interna é criada, e então, é comparada com todos os perfis de usuários existentes no sistema. A partir da comparação, o texto é trazido para a atenção do usuário cujo perfil apresenta algum grau de similaridade com a representação interna do documento. Estes textos recuperados são selecionados (ou não) e são avaliados em termos de como eles respondem aos interesses de informação e metas do usuário. A avaliação pode ser utilizada para modificação dos perfis e interesses de informação. Os perfis modificados são usados em processos subseqüentes de comparação.

2.2.3 Diferenças entre Recuperação e Filtragem de Informação

Recuperação de Informação (RI) e Filtragem de Informação (FI) [1][4][46] são duas técnicas que têm objetivos subentendidos equivalentes. Elas lidam com o processo de busca de informação. Dada a necessidade de informação apresentada pelo usuário tentam devolver um conjunto de documentos que satisfaçam a esta necessidade. A necessidade de informação é representada via consultas ou *queries* em SRI e via perfis ou *profiles* em SFI.

Os Sistemas de Filtragem da Informação (SFI), projetados para lidar com dados não-estruturados e semi-estruturados, compartilham muitas técnicas com os antigos Sistemas de Recuperação da Informação (SRI), mas diferenciam-se destes em três pontos básicos, conforme elucidado por Balabanovic [3] e Belkin [4]:

(a) Nos SRI, as consultas representam uma meta a curto prazo que pode ser satisfeita recuperando um conjunto particular de documentos; Nos SFI o perfil do usuário representa um interesse a longo prazo.;

(b) Nos SRI, aplicações assumem que os documentos não mudam constantemente com o tempo; Nos SFI, pressupõe-se um fluxo constante de documentos dependente do tempo;

(c) Os SRI atuam encontrando itens relevantes nos Bancos de Dados; Os SFI atuam removendo do fluxo itens irrelevantes.

Em Motta [46], encontra-se uma descrição mais detalhadas dessas diferenças.

Na literatura, RI é vista como antecessora da FI. A razão para isto é que RI é mais antiga e FI baseia vários de seus fundamentos em RI. Apesar do fato destas duas técnicas serem muito similares do ponto de vista de fundamentos, pode-se ainda destacar várias diferenças entre os dois que os tornam distintos. Uma comparação entre estes tópicos é delineada a seguir:

- Sistemas de RI são desenvolvidos para tolerar algumas inadequações na representação da consulta da necessidade da informação enquanto sistemas de FI assumem que perfis são acurados;
- Sistemas de RI são normalmente usados por um único usuário de uma só consulta [49], enquanto sistemas de FI são repetidamente usados pelo mesmo usuário com algum perfil;
- Sistemas de RI são desenvolvidos para servir aos usuários com necessidades de curto prazo, enquanto sistemas FI servem a usuários cujas necessidades são relativamente estáticas por um longo período de tempo;
- Sistemas de RI normalmente operam em coleções estáticas de documentos enquanto sistemas FI lidam com dados dinâmicos de seqüências de documentos;
- O primeiro objetivo da RI é coletar e organizar (ordenar de acordo com a importância) um conjunto de documentos que coincide com uma dada consulta, enquanto o objetivo primário da FI é

distribuir os novos documentos recebidos aos usuários com perfis coincidentes.

A comparação exposta entre FI e RI realça as principais diferenças entre os dois em relação a objetivos, uso, usuários e os tipos de dados que eles operam (estático vs. dinâmico).

De forma a prover os usuários com a informação requisitada, tanto sistemas RI quanto FI, devem representar a necessidade de informação (consulta e perfil respectivamente) e o conjunto de documentos de uma maneira adequada à comparação e combinação. Algumas representações correntemente usadas são: a representação vetorial e redes semânticas [43]. O mecanismo de comparação usado para combinar a necessidade de informação e de documentos depende da técnica de representação utilizada. Para melhorar a acurácia dos sistemas RI e FI, um mecanismo de *feedback* de relevância é utilizado. Uma vez que é apresentado ao usuário um conjunto de documentos retornado por um sistema RI ou FI, as medidas de precisão ou *recall* são calculadas. A grosso modo, estas duas medidas têm uma relação inversa, ou seja, conforme o número de resultados retornados aumenta, a probabilidade de retornar respostas erradas também aumenta.

Considerando a relação entre filtragem e recuperação de informação, após um estudo dos fundamentos de cada um destes processos, pode-se verificar que existem, relativamente, diferenças importantes entre os dois, em um certo nível de abstração. Todavia, suas metas básicas são essencialmente equivalentes. Isto é, ambos estão interessados em adquirir informação para pessoas com necessidades dela, e ambos estão interessados em mais ou menos o mesmo tipo de informação, e o mesmo tipo de contexto. Além disso, a maioria das questões que parecem, no princípio, ser iguais a filtragem da informação, são realmente especializações de problemas de RI [4].

Desta maneira, Recuperação de Informação e Filtragem de Informação são realmente dois lados de uma mesma moeda [4]. Elas trabalham juntas para ajudar pessoas a adquirirem a informação necessária para a realização de suas tarefas.

Uma vez vistos os conceitos de recuperação e filtragem de informação, algumas consideração sobre as Técnicas de Filtragem serão abordadas no próximo capítulo.

2.3 Considerações finais

Este capítulo apresentou diversos aspectos referentes a recuperação e filtragem de informação. Assim, em primeiro lugar, foi discutido o conceito de recuperação de informação, com seus principais aspectos. Em seguida, foram abordadas as características gerais da filtragem de informação. Por fim, foram expostas as principais diferenças entre ambas.

O próximo capítulo trata das técnicas de filtragem de informação, importante referencial teórico para o desenvolvimento do trabalho.

3 TÉCNICAS DE FILTRAGEM PARA SISTEMAS DE FILTRAGEM DE INFORMAÇÃO

A Web é uma excelente fonte de informação, mas um dos problemas que surge com a grande disseminação de informações é a dificuldade de se obter informação relevante em tempo hábil e de forma precisa. A falta de padronização, classificação e filtragem dos conteúdos da Web contribui para a formação deste cenário [1].

Nesse contexto, o usuário necessita utilizar mecanismos que auxiliem na recuperação de informações tais como os motores de busca de informação Google.com [76], Altavista [77] e Cadê [75]. Entretanto, às vezes, o usuário não consegue se expressar através de palavras-chave, e dessa maneira, os mecanismos de busca retornam uma grande quantidade de conteúdo, sem garantir uma boa efetividade de recuperação, com excesso de informações recuperadas ou com informações irrelevantes.

Os Sistemas de Filtragem de Informação surgem como alternativa de auxílio aos usuários na busca de informações, com filtragem de itens que serão apreciadas por estes. Estes sistemas possuem algoritmos que procuram otimizar a filtragem, *matching* entre usuários e filtrar pontualmente informações, que supõe-se serem relevantes.

Neste capítulo, procura-se fazer uma análise entre os diversos algoritmos de filtragem usados em sistemas de filtragem. Inicialmente, mostra-se o estado da arte de sistemas de filtragem, suas principais abordagens, de acordo com a técnica usada, suas limitações e são citados alguns exemplos de sistemas de filtragem usados nas mais diversas áreas.

3.1 Sistemas de Filtragem

Um Sistema de Filtragem de informação tem o objetivo de fornecer informações baseadas em registros sobre as preferências dos usuários. As preferências dizem respeito a quaisquer itens de interesse do usuário como um livro, uma música, um restaurante ou um filme [5].

As preferências dos usuários são geralmente armazenadas através de perfis, representados por *avaliações* (explicitas ou implícitas) ou *palavras-chave* extraídos de textos lidos no passado [69], por exemplo.

Dessa forma, os sistemas que utilizam técnicas de Filtragem de Informação para processar informações provêem ao usuário itens potencialmente mais relevantes do que sistema sem filtragem.

A Filtragem de Informação é usada para atender necessidades de informação em *longo prazo* a partir de fontes de informação dinâmica e não estruturada [8], apresentando basicamente três abordagens: Filtragem Baseada em Conteúdo, Filtragem Colaborativa ou social e Filtragem Híbrida.

Devido à abundância de aplicações práticas, esta área de pesquisa tem tido elevado interesse e constitui uma área rica de pesquisa, pois ajudam os usuários a tratar a informação sem sobrecarregar e fornecem documentos, itens, índices e serviços personalizados aos usuários [1].

A utilidade de um item em um sistema de filtragem é representada por uma *avaliação*, que indica quanto um usuário *gosta* de um *item* em particular. Um dos problemas com que os sistemas de filtragem lidam é com a quantidade possível de *itens* avaliados e de *usuários* avaliadores, que podem variar de milhares a milhões, além de outros problemas tais como: novos itens não avaliados, novos usuários e poucos usuários.

A área dos sistemas de filtragem de informação, por ser relativamente nova ainda não possui uma classificação plenamente aceita pela comunidade de usuários, profissionais e pesquisadores [68], dessa maneira classificar-se-á, para efeito didático, com a mesma classificação utilizada em filtragem de informação, existindo abordagens baseadas em conteúdos, colaborativas ou sociais e a junção destas duas abordagens, resultando em uma técnica híbrida. Também são denominados de Sistemas de Recomendação [1].

3.2 Técnicas de Filtragem de Informação

As principais técnicas de filtragem de informação são a Filtragem Baseada em Conteúdo e a Filtragem Colaborativa. Como ambas as técnicas possuem limitações, as abordagens Híbridas surgem combinando vantagens de ambas as técnicas para obter melhor performance. Estas técnicas fazem comparações entre documentos e perfis de usuários calculando a similaridade entre eles para então calcular a predição e fazer a filtragem.

A predição calcula o valor que o usuário *supostamente* dará a um item, enquanto que a filtragem é uma *sugestão* de um item, com base nos maiores valores calculados na predição.

3.3 Perfil do usuário

O perfil (ou modelo) do usuário procura retornar informações baseadas nos dados armazenados sobre o usuário de modo que elas se adeqüem cada vez mais aos anseios do mesmo. Nesse contexto, padrões são extraídos através de observações do comportamento do usuário a fim de predizer quais itens serão selecionados ou descartados [60].

Segundo Rocha [60], existem dois tipos de perfis de usuário:

a. Baseado em conteúdo, onde a necessidade de informação do usuário é expressa como uma consulta, lista de termos ou palavras-chave;

b. Colaborativo, que visa a avaliação dos padrões similares de usuários.

Muito embora os perfis sejam úteis para identificar o comportamento do usuário, alguns problemas podem ser encontrados, sendo que os problemas relacionados aos perfis de *usuários* são basicamente:

- a geração de perfis iniciais para usuários novos, pois começam com seu perfil sem informação alguma e é preciso preenchê-lo;
- quanto a atualização dos perfis ao longo do tempo.

3.4 Filtragem Baseado em Conteúdo

A Filtragem Baseada em Conteúdo [1] baseia-se em informações obtidas através da análise do conteúdo dos itens de informação pelos quais o usuário demonstrou preferência no passado, ou seja, um item será filtrado se este item é similar a outro que o usuário preferiu no passado. Nesse âmbito, pode se observar que a Filtragem Baseada em Conteúdo pode ser utilizada na filtragem de textos, como artigos, notícias, etc.

Para fazer a filtragem, os sistemas que utilizam essa abordagem inicialmente tentam classificar padrões de preferências existentes nos itens previamente avaliados pelo próprio usuário, criando assim um perfil.

Este perfil é composto por termos ou palavras-chave e pesos associados. O peso indica a maior ou menor “importância” de uma palavra na filtragem. De maneira semelhante, para cada item de informação são configurados perfis de representação, contendo os termos considerados mais importantes destes itens. Faz-se então uma comparação entre o perfil do usuário e os documentos, ou seja, através da análise de similaridade entre os documentos com o perfil do usuário e então os mais similares são filtrados. As seguintes atividades são geralmente realizadas pelas técnicas de Filtragem Baseada no Conteúdo:

I. Representação do perfil do usuário;

II. Representação dos documentos;

III. Comparação entre perfil e documentos;

IV. Análise de similaridade;

V. Filtragem.

A Figura 3 mostra o processo descrito acima

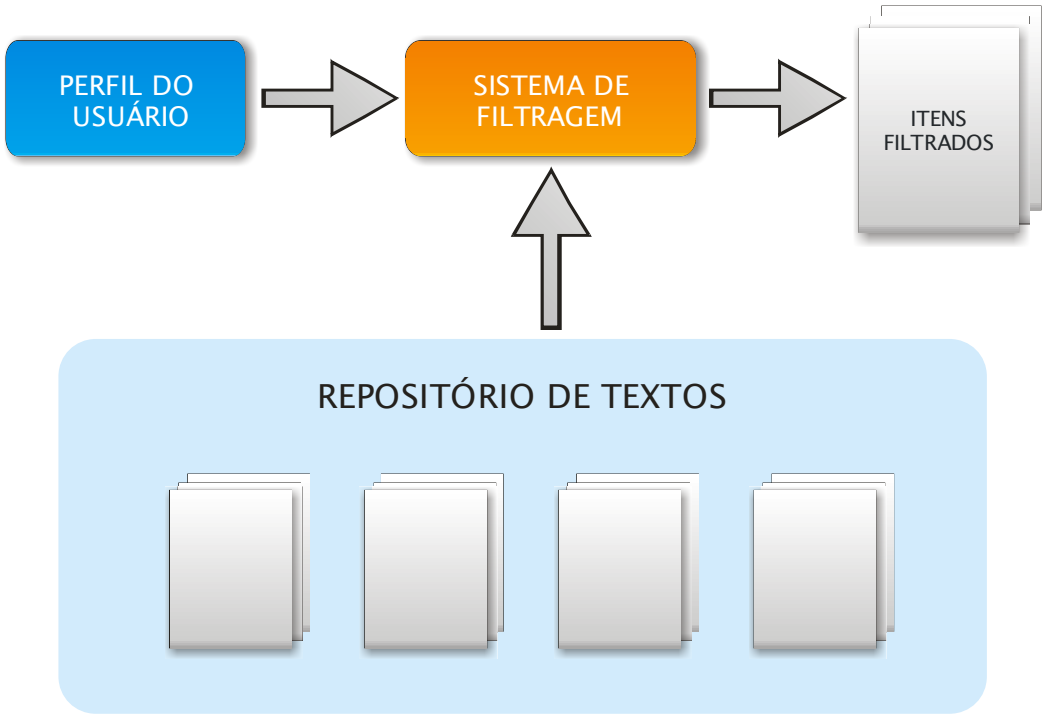

Figura 3 - Filtragem Baseada em Conteúdo. Adaptado de Torres [68].

3.4.1 Similaridade em Filtragem Baseada em Conteúdo

Similaridade é a medida que define a distância entre dois itens sejam eles dois usuários, um usuário e um documento ou entre dois documentos. A similaridade, entre dois documentos, por exemplo, é feita comparando as palavras comuns a ambos os documentos.

A técnica mais usada para medir a similaridade é o TF-IDF [1] (*Term-Frequency-Inverse-Document-Frequency*), que é a Freqüência de Termos e a Freqüência Inversa de Documentos, baseado no modelo vetorial, onde os documentos são representados como vetores de palavras dentro de um espaço vetorial, muito usado na técnica de recuperação de informação. O Cálculo é feito como mostrado na equação 1, onde $f_{i,j}$ é o numero de vezes que a palavra-chave ki aparece no documento dj. Então $TF_{i,j}$, ou seja, a freqüência do termo (freqüência normalizada) é definida em (1) onde o máximo é computado sobre a freqüência $f_{z,j}$ de todos os termos ou palavras-chave k_z que aparecem no documento d_j.

$$TF_{i,j} = \frac{f_{i,j}}{\max_z f_{z,j}} \qquad (1)$$

O IDF, apresentado na equação 2, é a freqüência inversa de um documento para uma palavra-chave, onde N é o número de documentos de uma coleção e n_i é o número de documentos da coleção em que a palavra-chave apareceu. Assim quanto menos uma palavra ocorre em um conjunto de documentos maior será seu IDF.

$$IDF_i = \log \frac{N}{n_i} \qquad (2)$$

Um problema que surge com a técnica da freqüência de termos é que uma palavra-chave que aparece muitas vezes em um documento muitas vezes não é útil para determinar se um documento é relevante ou irrelevante. Dessa maneira, utiliza-se uma combinação entre o TF e o IDF para atingir melhores resultados. Sendo assim, o peso TF-IDF para palavras chaves em um documento é definido pela equação 3, onde TF representa a freqüência de termos e IDF a freqüência inversa de termos:

$$w_{i,j} = TF_{i,j} x IDF_i \qquad (3)$$

A principal vantagem dos sistemas que utilizam Filtragem Baseada em Conteúdo e a independência do número de usuários. No entanto, a mesma apresenta algumas limitações:

- Impossibilidade em encontrar itens que poderiam interessar ao usuário e que não possuem similaridade de conteúdo com outros itens avaliados pelo usuário (*Overspecialization*);
- Análise de conteúdo limitado:
    - Análise de conteúdo não textual, como por exemplo, dados multimídia como imagens, gráficos, áudio, vídeo;

- Avaliação da qualidade do texto, pois não há como diferenciar se um documento está bem escrito ou mal escrito;

- Problemas com novos usuários – Os novos usuários têm que avaliar um número suficiente de artigos antes que o sistema possa realmente compreender suas preferências e apresentar recomendações de confiança. Conseqüentemente, um usuário novo tendo poucas avaliações, pode não ter uma filtragem eficiente.

3.5 Filtragem Colaborativa

A Filtragem Colaborativa [1] é uma técnica utilizada na filtragem de itens que utiliza a similaridade entre os *usuários* para sugerir itens, não exigindo a análise do conteúdo dos itens e trabalha na troca de experiência de avaliações de interesses comuns [10]. Assim, permite aumentar a precisão na busca de informações importantes para um grupo de usuários através de análise entre avaliações realizadas por outros usuários com interesses comuns e que atendam aos seus interesses.

As seguintes atividades são geralmente realizadas pelas técnicas de filtragem colaborativa e ilustrada na Figura 4:

1. Obter avaliações dos usuários e construir modelos de usuários;
2. Encontrar usuários vizinhos a um usuário alvo, ou seja, com interesses comuns através da análise de similaridade;
3. Através de um algoritmo específico determinar a vizinhança;
4. Realizar a predição;
5. Filtrar a Informação.

Para que possa ocorrer a filtragem, é necessário criar *modelos de usuários* e dessa forma modelar e classificar grupos de usuários similares através da análise dos *modelos de usuários* definidos por um certo critério de classificação, produzindo assim grupos de usuários de perfis similares usados como base para a filtragem.

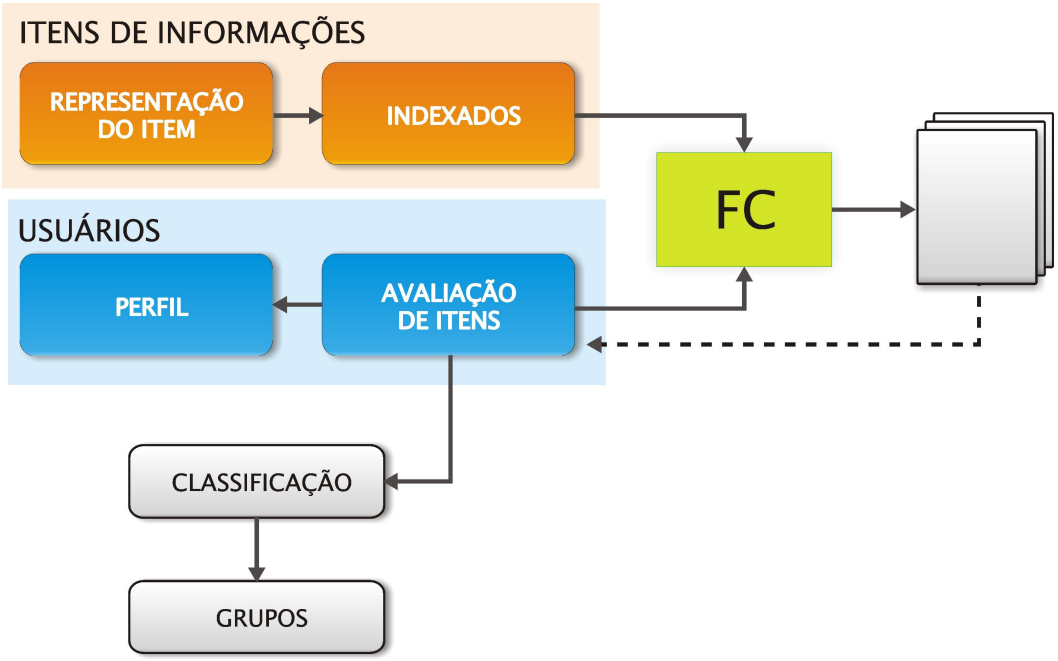

Figura 4 – Filtragem Colaborativa.

A Tabela 1 exemplifica como é feita a filtragem colaborativa, caso se queira filtrar um item a um usuário. Caso seja necessário se fazer uma sugestão ao usuário Francisco, procura-se os usuários que tenham hábitos similares a ele: neste caso João e Gabriel têm um perfil semelhante ao de Francisco e estes têm "itens" diferentes ao de Francisco que podem ser sugeridos: os itens 1 e 3.

Usuário	Item 1	Item 2	Item 3	Item 4
João	x	x		
Maria	x		x	
Gabriel		x	x	
Francisco		x		
Lucas				
Alisson				
Heider				
Eduardo				
Simone				
Raimundo				

Tabela 1 – Filtragem Colaborativa. Adaptado de Torres [68].

Existem vários tipos de algoritmos usados na Filtragem Colaborativa, sendo que os mais comumente citados e usados são: o método de *vizinhos mais próximos* [13]; as abordagens de usuários-usuários [1] , item-item [64], onde ocorre o cálculo da similaridade entre produtos (itens) e não com os usuários.

3.5.1 Similaridade na Filtragem Colaborativa

Medidas de similaridade são medidas de distância e são geralmente usadas como “pesos” para encontrar os vizinhos mais próximos, ou seja, quão similares são as avaliações dos usuários, Tabela 2, e dessa maneira melhora a estimação de predição.

Usuário	Item A	Item B	Item C	Item D
João	1		5	
Maria	5	1	2	
Gabriel	6	5	2	
Francisco	2	3	4	
Lucas				
Alisson				
Heider				
Eduardo				
Simone				
Raimundo				

Tabela 2 – Matriz de Avaliações de Itens. Adaptado de Torres [68].

A similaridade em filtragem colaborativa se dá na comparação das avaliações dos usuários para itens que este comprou, utilizou ou que de certa forma teve alguma preferência, usada no modelo usuário-usuário, que pode ser calculada através do *Coeficiente de Pearson* ou através do *cosseno* (com os vetores que representam os usuários), sendo usado como um valor intermediário para calcular o vizinhos mais próximos de um usuário *u*.

3.5.1.1 Coeficiente de Pearson

Também chamado de Coeficiente de *correlação* (*correlation-based*) [1], mede a "força" de relação entre dois perfis de usuários x e y, conforme mostra a Equação 4 [1]:

$$sim\,(x,y) = \frac{\sum_{i=1}^{n}(r_{x,i} - \bar{r}_x) * (r_{y,i} - \bar{r}_y)}{\sqrt{\sum_{i=1}^{n}(r_{x,i} - \bar{r}_x)^2}\sqrt{\sum_{i=1}^{n}(r_{y,i} - \bar{r}_y)^2}} \quad (4)$$

Onde:

- Sim(x,y) - é a similaridade entre o usuário atual *x* com um determinado usuário *y*;
- $r_{x,i}$ - é a avaliação que o usuário *x* deu ao item *i*;
- $\bar{r}_x$ - é a média de todas as avaliações do usuário ativo *x*.

Como resultado obtém-se um número inteiro no intervalo [-1;1], onde *-1* indica uma fraca relação e *1* indica uma forte relação entre os usuários.

O cálculo da similaridade leva em conta, para a soma e média, somente as avaliações de itens em que ambos os usuários fizeram avaliações, ou seja, as avaliações em comum.

O perfil do usuário é construído com base nas avaliações que este fez a um determinado item, geralmente representado por um vetor. Por exemplo, o perfil do usuário Gabriel = [6,5,2] é um vetor de 3 dimensões.

3.5.1.2 Cosseno

O cálculo da similaridade usando o cosseno [1] é feito calculando-se o vetor (*cosine-based*) entre os usuários, que são tratados como vetores num espaço n-dimensional, onde *n* é o número de itens dos vetores, descrito no algoritmo abaixo:

$$sim(x,y) = \cos(\vec{x},\vec{y}) = \frac{\vec{x} * \vec{y}}{\left\|\vec{x}\right\|_2 * \left\|\vec{y}\right\|_2} = \frac{\sum_{i=1}^{n}\left(r_{x,i} * r_{y,i}\right)}{\sqrt{\sum_{i=1}^{n}(r_i)^2}\sqrt{\sum_{i=1}^{n}(r_{y,i})^2}} \quad (5)$$

Onde:

- $\vec{x}$ e $\vec{y}$ - são os vetores associados aos usuários *x* e *y*; respectivamente;
- $\left\|\vec{x}\right\|$ - é a norma do vetor $\vec{x}$;

Como resultado obtém-se o cosseno do ângulo que é um número inteiro no intervalo [0,1]. Então quanto mais próximo de 1, mais similares são os usuários.

Existem propostas de extensão destas abordagens que visam melhorar a performance de seus algoritmos, como por exemplo: votação *default*, freqüência inversa de usuários e amplificação de casos [1], entretanto essas abordagens fogem ao escopo deste trabalho e , portanto não serão, abordadas.

3.5.2 Criando a Vizinhança

Após a utilização de um coeficiente para medir a similaridade entre os usuários, seja pelo coeficiente de Pearson ou pelo cosseno, é preciso agrupá-lo em vizinhanças o que pode ser feito de duas formas: através da *similaridade* ou através do *número de vizinhos*.

A **similaridade** é feita determinando-se um critério de comparação de valores chamado de *limiar de similaridade* [68] . Dessa maneira, os usuários que tiverem um valor superior ao limiar serão definidos como vizinhos do usuário alvo. O problema existente nesta abordagem é a possível geração de uma vizinhança pequena ou nenhuma vizinhança, gerando assim um usuário “ovelha negra”.

Usuário	Cosseno
João	0,88
Maria	0,51
Gabriel	0,84
Francisco	0,79
Lucas	0,55
Alisson	0,98
Heider	0,70
Eduardo	0,78
Simone	
Raimundo	

Tabela 3 – Cálculo da similaridade. Adaptado de Torres [68].

Para exemplificar melhor observemos a Tabela 3, pode-se determinar um limiar de similaridade para encontrar os vizinhos do usuário Eduardo utilizando o cálculo do cosseno para um limiar de 80%. Assim temos como vizinhos mais similares deste os usuários João, Gabriel e Alisson, este último com 98%. Tem-se então a determinação da vizinhança através da similaridade.

A determinação do **Número de vizinhos** é feita determinando-se um critério de parada, ou seja, um número máximo de usuários que serão classificados como vizinhos do usuário alvo independente de um limiar. Esta abordagem apresenta como principal problema a filtragem de baixa qualidade.

Para se determinar a vizinhança através do número de vizinhos delimita-se uma quantidade máxima de vizinhos similares que um usuário pode ter. Pode-se exemplificar tomando como base o exemplo da Tabela 3, determinado que o usuário Eduardo terá no máximo 5 vizinhos. Assim temos como vizinhos mais similares os usuários Simone, Francisco, João, Gabriel e Alisson. Observa-se neste exemplo, Tabela 3, que o usuário Simone possui uma similaridade de 78%, sendo razoavelmente similar a Eduardo, mas caso o número de vizinhos fosse maior seria possível criar uma vizinhança com aqueles que têm um índice ainda menor, gerando assim baixa qualidade de filtragem.

3.5.3 Geração de Sugestão ou Filtragem

A Filtragem para um usuário é feita com base na predição calculada. A predição calcula o valor que o usuário *supostamente* dará a um item, enquanto que a filtragem é a *sugestão* de um item, com base nos maiores valores calculados na predição. A predição pode ser calculada da seguinte forma, como mostrado na Equação 6 :

$$P_{x,i} = \bar{r}_x + \frac{\sum_{y=1}^{n} (r_{y,i} - \bar{r}_y) * sim_{x,y}}{\sum_{y=1}^{n} \left| sim_{x,y} \right|} \qquad (6)$$

onde:

- $P_{x,i}$ - é a predição para o usuário x do item *i*;
- $\bar{r}_x$ - é a média de todas as avaliações do usuário *x*;
- $sim_{x,y}$ - é a similaridade entre o usuário *x* com um usuário *y*.

Assim, calcula-se a avaliação dada por todos os vizinhos $\sum_{y=1}^{n}$ e este valor é ponderado utilizando-se a similaridade com o usuário alvo ($sim_{x,y}$). O valor encontrado é um número dentro da escala de avaliação determinado no sistema. A filtragem usa então o maior valor da predição para gerar a sugestão.

Os sistemas baseados em filtragem colaborativa possuem algumas vantagens em relação à Filtragem Baseada no Conteúdo:

- Independente de conteúdo;
- Possibilidade de filtrar itens baseado na qualidade e gostos;
- Providencia recomendações dinâmicas.

Torres [69][67] ressalta que as vizinhanças não são pré-computadas, sendo que cada nova requisição gera uma nova vizinhança, ressaltando assim a qualidade da Filtragem Colaborativa.

Não obstante, os sistemas baseados em filtragem colaborativa apresentam algumas limitações:

- **Recomendação de novos itens**: um item só poderá ser recomendado após ser avaliado por um usuário;
- **Número de usuários insuficiente (esparsidade)**: neste caso, o sistema apresenta baixa performance, pois o número baixo de usuários em relação ao volume de informação impossibilita a criação da vizinhança, já que não haverá vizinhos próximos o suficiente para cada usuário;
- **Usuário "ovelha negra"**: neste caso há falta de sobreposição de gostos quando não há vizinhos próximos o suficiente para que a filtragem alcance os interesses do usuário alvo, uma vez que estas são baseadas em usuários que apresentam interesses em comum.

Por conta das limitações particulares de ambas técnicas, é comum encontrar sistemas que utilizam abordagens Híbridas, combinando as vantagens das técnicas Colaborativa e Baseada em Conteúdo, com o objetivo de vencer as limitações de cada uma das referidas abordagens.

3.6 Filtragem Híbrida

A Filtragem Híbrida [23] consiste em combinar diversos recursos utilizados na Filtragem Baseada no Conteúdo e na Filtragem Colaborativa. O objetivo principal é suprir limitações existentes nessas duas abordagens. Existem inúmeras técnicas para realizar essa combinação, as quais basicamente fundamentam-se nos seguintes princípios [1]:

1. Cálculo, separadamente, de resultados e combinação posterior das predições;
2. Introdução de características da Filtragem Baseada em Conteúdo na Filtragem Colaborativa;
3. Introdução de algumas características da Filtragem Colaborativa na Filtragem Baseada no Conteúdo e;
4. Construção de *modelos genéricos* envolvendo as duas técnicas.

A técnicas de combinação para a Filtragem Híbrida podem ser visualizadas na Figura 5:

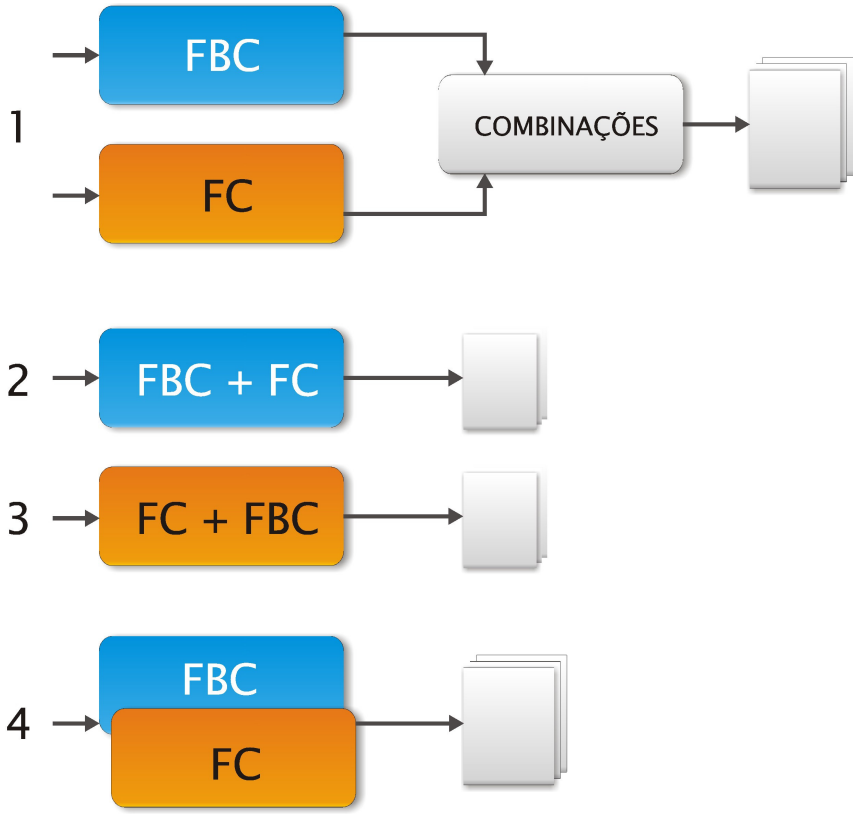

Figura 5 – Filtragem Híbrida. Adaptado de Adomavicius [1].

Para a combinação de filtragem separadamente há basicamente dois cenários de combinações. O primeiro refere-se a calcular separadamente as possíveis avaliações e depois combinar estas avaliações em uma só avaliação final. Outra forma é o uso temporário de cada método dependendo do domínio de aplicação, e alternar o uso deles a fim de melhorar a qualidade da filtragem.

Na adição de características de filtragem baseada em conteúdo na filtragem colaborativa, há a construção de perfis de usuários através da filtragem baseada em conteúdo para que este perfil seja considerado além do cálculo de similaridade entre usuários, que é própria da filtragem colaborativa. Dessa maneira, os problemas relacionados à esparsidade são eliminados ou minimizados.

Ao adicionar características de filtragem colaborativa na filtragem baseado em conteúdo, tem-se como objetivo dimensionalidade num grupo de

perfis para criar uma “visão” colaborativa de uma coleção de perfis de usuários, que são representados por um vetor de termos e assim obter uma melhor performance comparado ao uso da abordagem puramente baseada em conteúdo.

Na unificação das técnicas, desenvolvidas recentemente, propõe o uso de classificadores baseados em regras [9], a unificação dos métodos probabilísticos baseado na análise de semântica latente [46] ou o uso misto de modelos Bayesianos com o método de Markov [1] para construir atributos de usuário a partir de um perfil de usuário e um atributo de item a partir de um perfil de item. Assim através de suas interações estima-se a avaliação de um determinado item.

As vantagens das técnicas híbridas derivam das associações da filtragem colaborativa e da baseada no conteúdo [10]:

a. Baseado no Conteúdo
    i. Bons resultados para usuários incomuns;
    ii. Precisão independente do número de usuários.
b. Colaborativa
    i. Descoberta de novos relacionamentos entre usuários;
    ii. Filtragem de itens diretamente relacionados ao histórico

Apesar das vantagens obtidas pela combinação das técnicas colaborativa e baseada em conteúdo, a filtragem híbrida ainda possui limitações quanto a itens novos não textuais, pois não foram avaliados.

3.7 Análise entre Técnicas de Filtragem baseada em Conteúdo e Colaborativa

A partir do estudo das principais técnicas de filtragem de informação podemos montar uma tabela comparativa, Tabela 4, que mostra de forma sucinta as vantagens e limitações de cada técnica:

Técnica de Filtragem	Vantagens	Limitações
Baseada no Conteúdo	Independência do numero de usuários	Dependência de Conteúdo
		Não usa qualidade e gostos
	Possibilidade de filtrar todos os itens	Superespecialização (“Overspecialization”)
		Novos usuários
Colaborativa	Independência de conteúdo	Avaliação de novos itens
	Baseado na qualidade e gostos	Insuficiência de usuários
	Filtragens inesperadas	“Ovelha negra”
Híbrida	Suprir certas limitações existentes na Filtragem Colaborativa e na baseada em Conteúdo	Não resolve o problema do *start-up*: • Itens novos não textuais;

Tabela 4 - Vantagens e Limitações das Técnicas de Filtragem. Fonte: Adomavicius [1].

Adomavicius [1] cita algumas possíveis extensões para melhorar a capacidade dos sistemas de filtragem como: compreensão de usuários e itens, incorporação de informação contextual, suporte a multicritérios de avaliação, multidimensão da filtragem, criação de técnicas menos intrusivas, mais flexíveis e efetivas para então obter melhor performance.

3.8 Considerações finais

Neste capítulo foram vistos as principais técnicas de filtragem de informação utilizadas em Sistemas de Filtragem.

Iniciou-se expondo o conceito geral sobre sistemas de filtragem, essencial para o entendimento das técnicas explanadas subsequentemente.

Em seguida, explanou-se sobre a Filtragem Baseada em Conteúdo, com suas características, vantagens e limitações. Logo em seguida a técnica de Filtragem Colaborativa com suas abordagens além das vantagens e limitações.

Finalmente, foi exposto as características da técnica de Filtragem Híbrida, foco deste trabalho, como também uma tabela comparativa entre as técnicas apresentadas.

O próximo capítulo trata dos algoritmos mais utilizados nas técnicas de filtragem abordadas neste capitulo, e essências para delineação do sistemas proposto.

4 ALGORITMOS DE FILTRAGEM

Os modelos clássicos de recuperação de informação tais como booleano, vetorial e probabilístico apresentam estratégias de busca de documentos relevantes para uma consulta (*query*) e são também utilizados no processo de Filtragem de Informação.

Estes modelos consideram que cada documento é descrito por um conjunto de palavras chaves, chamadas termos de indexação. Associa-se a cada termo de indexação t_i em um documento d_j um peso $w_{ij} \geq 0$, que quantifica a correlação entre os termos e o documento.

Além destes, existem outros modelos ou algoritmos associados à Filtragem de Informação com estratégias relacionadas com cada etapa da filtragem.

4.1 Modelo Booleano

O Modelo Booleano [4][47] é uma técnica utilizada tanto na filtragem como na recuperação de informação. Dada uma consulta *Q* e um conjunto de documentos considerados relevantes para *Q*, o índice atribuído aos documentos deve indicar qual documento é mais relevante que o outro, estabelecendo uma ordem de relevância. Esses índices são calculados com base na comparação entre a consulta e os documentos.

Neste modelo, os documentos recuperados/filtrados são aqueles que contêm os termos que satisfazem a expressão lógica da consulta. Uma consulta é considerada como uma expressão booleana convencional formada com os conectivos lógicos AND, OR e NOT.

Uma maneira direta de implementar o modelo booleano seria [62]: que assuma a existência de uma lista invertida na qual cada entrada corresponde a um termo de indexação, ademais, a entrada t_i aponta para uma lista de documentos nos quais o termo t_i ocorre. O conjunto de documentos recuperados pode ser obtido pela interseção das listas invertidas de documentos, dos termos que aparecem na consulta. Assim, somente

documentos cujos termos de indexação satisfazem a consulta booleana são recuperados.

Os principais problemas do modelo booleano são a ausência de ordem na resposta, e as respostas podem ser nulas ou muito grandes. As vantagens desse modelo são a facilidade de implementação, e a expressividade completa das expressões.

4.2 Modelo Vetorial

O modelo de espaço vetorial [4][47], ou simplesmente modelo vetorial, criado por Gerald Salton para ser utilizado num Sistema de Recuperação de Informação chamado SMART, representa documentos e consultas como vetores de termos. Termos são ocorrências únicas nos documentos. Os documentos devolvidos como resultado para uma consulta são representados similarmente, ou seja, o vetor resultado para uma consulta é montado através de um cálculo de similaridade.

Aos termos das consultas e documentos são atribuídos pesos que especificam o tamanho e a direção de seu vetor de representação. Ao ângulo formado por estes vetores dá-se o nome de θ. O $cos\theta$ determina a proximidade da ocorrência. O cálculo da similaridade é baseado neste ângulo entre os vetores que representam o documento e a consulta, através da equação 7 [62].

$$sim(d,q) = \frac{\sum_{i=1}^{t} w_{id} \times w_{iq}}{\sqrt{\sum_{i=1}^{t} w_{id}^2} \times \sqrt{\sum_{i=1}^{t} w_{iq}^2}} \qquad (7)$$

Cada documento possui um vetor associado que é constituído por pares de elementos na forma {(palavra_1, peso_1), (palavra_2, peso_2),..., (palavra_n, peso_n)}.

Os pesos quantificam a relevância de cada termo para as consultas (W_{iq}) e para os documentos (W_{id}) no espaço vetorial. Para o cálculo dos pesos W_{iq} e W_{id}, utiliza-se uma técnica que faz o balanceamento entre as

características do documento, utilizando o conceito de freqüência de um termo num documento. Se uma coleção possui *N* documentos e *n*ti é a quantidade de documentos que possuem o termo t_i, então o inverso da freqüência do termo na coleção, ou *idf* (*Inverse Document Frequency*) é dado por:

$$IDF_i = \log \frac{N}{n_i} \qquad (8)$$

Este valor é usado para calcular o peso, utilizando a seguinte fórmula: $W_{id} = freq(t_i,d) \times idf_i$, ou seja, é o produto da freqüência do termo no documento pelo inverso da freqüência do termo na coleção, visto na seção 3.4.1.

As principais vantagens do modelo vetorial são a sua simplicidade, a facilidade que ele provê de se computar similaridades com eficiência e o fato de que o modelo se comporta bem com coleções genéricas. Um problema característico seria que um documento relevante poder não conter termos da consulta.

4.3 Modelo Probabilístico

O modelo probabilístico [55] descreve documentos considerando pesos binários que representam a presença ou ausência de termos. O vetor resultante, gerado pelo modelo, tem como base o cálculo da probabilidade de que um documento seja relevante para uma consulta. A principal ferramenta matemática do modelo probabilístico é o teorema de Bayes [70].

Este modelo é baseado no princípio de ordenação probabilístico (*Probability Ranking Principle*). Neste modelo, busca-se saber a probabilidade de um documento *D* ser ou não ser relevante para uma determinada consulta *Q*. Tal informação pode ser obtida assumindo-se que a distribuição de termos na coleção seja capaz de informar a relevância provável para um documento qualquer da coleção.

Devem ser calculados: $P(+R_q|d)$ a probabilidade de que um documento *d* seja relevante para uma consulta *q* e $P(- R_q|d)$ a probabilidade de que um documento *d* não seja relevante para uma consulta *q*. O documento *d* é

considerado relevante para a consulta q se $P(+R_q|d) > P(-R_q|d)$, e o vetor resultado é decidido com base num fator $W_{d|q}$, definido por:

$$W_{d|q} = \frac{P(+R_q \mid d)}{P(-R_q \mid d)} \qquad (9)$$

Sendo que este fator minimiza a média do erro probabilístico.

Através do teorema de Bayes e estimativas de relevância baseadas nos termos da consulta, pode-se chegar a seguinte equação:

$$sim(d,q) = W_{d|q} = \sum_{i=1}^{t} x_i \times W_{qi} \qquad (10)$$

O modelo probabilístico tem como vantagem, além do bom desempenho, o princípio probabilístico de ordenação, que uma vez garantido, resulta em um comportamento ótimo do método. Entretanto, a desvantagem é que este comportamento depende da precisão das estimativas de probabilidade. Além disso, o método não explora a freqüência do termo no documento e ignora o problema de filtragem de informação.

4.4 K vizinhos mais próximos - KNN

O método K-vizinhos mais próximos (*K nearest neighbors*) *ou KNN*, proposto originalmente por Cover [13], calcula a distância, através de uma métrica, dos vizinhos mais próximos de um novo usuário *x* numa base de referencia *k* ou de um usuário com um documento.

Este método é muito usado em aplicações envolvendo a tarefa de classificação. O funcionamento do K-NN é dado da seguinte maneira: considerando uma base de dados, uma base de referência, de um problema envolvendo a tarefa de classificação e cada novo registro a ser classificado são executados os seguintes passos:

1. Cálculo da distância do novo usuário a cada um dos usuários já existentes, utilizando alguma métrica de distância (Euclidiana, Hamming, Minkowski) [55];
2. Identificação dos *k* usuários que apresentam menor distância em relação ao novo usuário;
3. Apuração da classe mais freqüente entre os *k* usuários identificados no passo anterior;
4. Comparação da classe apurada com a classe real, computando erro ou acerto do algoritmo, quando as classes dos novos registros são conhecidos e deseja-se avaliar o desempenho do método K-NN.

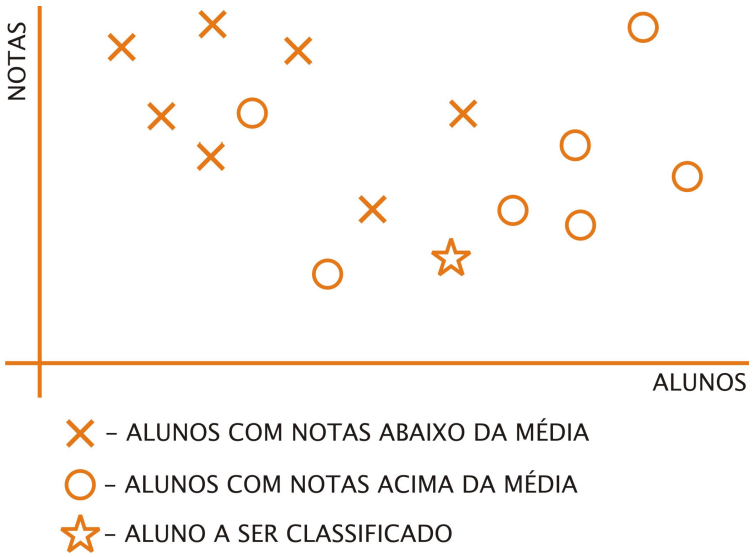

Figura 6 – Alunos com notas positivas e negativas. Adaptado de Passos [55].

Considerando o exemplo da Figura 6, onde temos este conjunto dividido em duas classes: alunos com notas positivas, acima da média, representados com um “x” e alunos com notas negativas, abaixo da média, representados com um “○”.

Apresentando-se um novo registro, representado por “☆”, calcula-se a distância entre o novo registro e todos os registros existentes na base de dados de referência. Assumindo que o número de k de vizinhos mais próximos

seja 3, ou seja, apenas os 3 registros com menor distância ao novo registro são considerados.

Dessa forma, avaliando os resultados da Figura 7, observa-se que a classe com maior ocorrência dentro da área delimitada pelo algoritmo K-NN foi “aluno com nota positiva”

Figura 7 – Resultado K-NN. Adaptado de Passos [55].

O objetivo é filtrar itens baseados na predominância dos *k* vizinhos mais próximos. K é igual ao número de vizinhos mais próximos e os vizinhos mais próximos são os usuários que possuem maior valor de similaridade, onde é feito uma Generalização/Classificação.

4.5 Árvore de Decisão

As árvores de decisão [56] são formas simples para representar a classificação de um repositório de dados. Este repositório são estruturas de dados que caracterizam uma relação entre os dados que a compõem. Esta relação existente entre os dados (denominados nós) representa uma relação hierarquia, onde um conjunto de dados é hierarquicamente subordinado ao outro.

Tem como função descobrir uma relação entre vários atributos preditivos e um atributo objetivo, com a intenção de gerar um novo

conhecimento que ajude a prever o resultado da ocorrência de um conjunto de atributos, ou seja, ela tenta gerar uma ou mais regras de classificação que definam as combinações de atributos que predizem algo em relação ao atributo objetivo. Uma árvore de decisão é definida pela estrutura de nodos, galhos e folhas, como mostra a Figura 8.

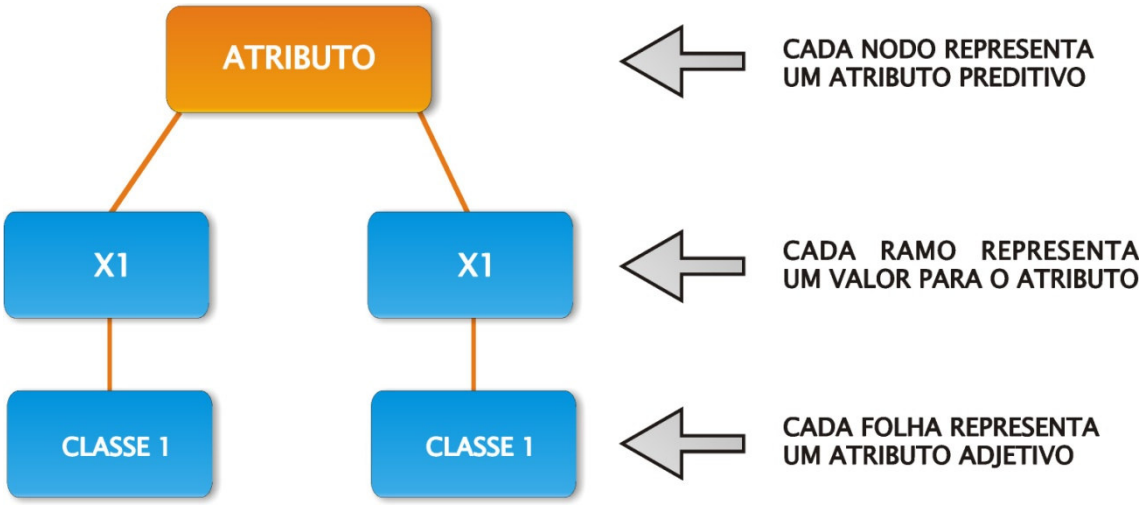

Figura 8 - Estrutura das Árvores de Decisão. Adaptado de Quinlan [56].

Na árvore de decisão, cada nodo de galho (atributos) representa uma escolha entre uma quantidade de alternativas, e cada nó de folhas (classe) representa uma classificação ou decisão [34].

O modelo Árvore de Decisão é amplamente utilizado em métodos práticos de inferência indutiva e consegue fazer aproximações de funções com valores discretos, possui a vantagem de ser robusto para manipular dados com ruídos e ser capaz de aprender expressões disjuntivas [73].

O modelo árvore de decisão classifica as instâncias percorrendo uma árvore a partir do nó raiz até alcançar uma folha. Cada um dos nós testa o valor de um único atributo e, para cada uma de suas valorações, oferece arestas diferentes a serem percorridas na árvore a partir deste nó. Sua vantagem é a estratégia adotada conhecida por dividir para conquistar, que divide um problema maior em outros menores. Assim, sua capacidade de discriminação dos dados provém da divisão do espaço definido pelos atributos em subespaços [45].

De acordo com [55], para a construção de uma árvore de decisão a idéia base é: 1Escolher um atributo; 2 Estender a árvore adicionando um ramo para cada valor do atributo; 3 Passar os exemplos para as folhas (tendo em conta o valor do atributo escolhido); 4 Para cada folha: Se todos os exemplos são da mesma classe, associar esta classe à folha; Senão repetir os passos 1 a 4.

Estes passos constituem a idéia básica do algoritmo de indução de árvores de decisão chamado *C4.5* [57] onde o mesmo permite a construção das árvores. A ferramenta C4.5 implementa o algoritmo c4.5, que é um aprimoramento de um algoritmo chamado ID3, que minera dados de acordo com a técnica de classificação. Sua função básica é a de gerar árvores de decisão e regras de classificação a partir de um repositório de dados [56].

A seguir, tem-se o algoritmo C4.5 em pseudo-código visto na Figura 9.

```
ROTINA ArvoreDecisao(ConjuntoExemplo, Atributos) RETORNA Arvore;
INICIO
  SE todos os elementos do ConjuntoExemplo são da mesma classe
  ENTÃO RETORNAR uma folha rotulado com esta classe;
  SENÃO SE Atributos igual a vazio
        ENTÃO RETORNAR folha com valor mais comum de Atributos no
ConjuntoExemplo
        SENÃO
          INICIO
            SELECIONAR um Atributos a transformando-a em raiz;
           RETIRAR a de Atributos;
            PARA cada valor v de a
              INICIO
                CRIAR um ramo da árvore rotulada com v;
               CONSTRUIR partição_v com elementos de  ConjuntoExemplo(p,v);
               ArvoreDecisao(partição_v, Atributos);
                ADICIONAR retorno recursivo no ramo v;
              FIM
          FIM
   FIM
FIM
```

Figura 9 – Algoritmo C4.5 para indução de árvores de decisão.

4.5.1 Árvores de Decisão em Sistemas em Filtragem

Li e Yamada [34] propõem o uso de tecnologias baseadas em aprendizado indutivo através das árvores de decisão para representar as preferências dos usuários.

Para esta aplicação, informações de usuários de uma universidade com a finalidade de prever em qual área de pesquisa se interessam mais será tomado como exemplo.

A árvore de decisão classificará os usuários prevendo seus interesses de acordo com determinada área (e.g. agentes). As preferências dos usuários estarão representadas no modelo de espaço vetorial contendo itens avaliados pelo usuário.

Os atributos considerados para o exemplo universidade são: *status*, país, área de interesse dos usuários para o valor de atributo meta (Predicado Objetivo) Filtra ou Não Filtra seguida dos demais valores. Inicialmente, obteve-se a representação das entradas conforme a Figura 10.

```
...
Hashtable table;
Vetor ExemploAtrib[] = new Vector();  //Vetor de termos para os atributos
Vetor ExemploVal[] = new Vector();    //Vetor de termos para os valores

ExemploAtrib = {Status, Area de interesse...}
ExemploVal = {Professor, Aluno, Pesquisador}

for (int i=0; i<ExemploAtrib[]; i++)
   for (int j=0; j<ExemploVal[]; j++)
      table.put(ExemploVal [j], ExemploAtrib [i] )  //A cada laço carrega os
atributos e os valores
...
```

Figura 10 - Representação das entradas para a árvore de decisão.

A seguir, na Tabela 5, tem-se o conjunto de treinamento a serem induzidos pelo algoritmo C4.5.

Status	País	Área de Interesse	Predicado Objetivo
pesquisador	Brasil	agentes	Filtra
professor	Uruguai	agentes	Filtra
pesquisador	Franca	agentes	Filtra
Aluno	Uruguai	arvores de decisao	Filtra
pesquisador	Franca	arvores de decisao	Filtra
professor	Brasil	arvores de decisao	Filtra
Aluno	Brasil	arvores de decisao	Filtra
Aluno	Suica	arvores de decisao	Filtra
professor	Uruguai	arvores de decisao	Filtra
professor	Japao	auml	Filtra
pesquisador	Brasil	auml	Filtra
professor	Brasil	EConhecimento	Filtra
professor	Franca	EConhecimento	Filtra
Aluno	Brasil	EConhecimento	Filtra
professor	Suica	EConhecimento	Filtra
professor	Argentina	EConhecimento	Filtra
Aluno	Franca	EConhecimento	Filtra
pesquisador	Argentina	banco de dados	Nao Filtra
pesquisador	Suica	banco de dados	Nao Filtra
Aluno	Japao	banco de dados	Nao Filtra
pesquisador	Japao	corba	Nao Filtra
Aluno	Argentina	uml	Nao Filtra

Tabela 5 - Entrada dos dados para indução da árvore de decisão.

Cada linha da tabela representa uma instância do conjunto de exemplos universidade. Russel e Norvig [61] acrescentam que se pode construir, para cada instância, uma árvore de decisão que tem um caminho para uma folha correspondente, onde o caminho realiza teste em cada atributo por vez e acompanha o valor correspondente à instância e à folha que possui a classificação desta instância.

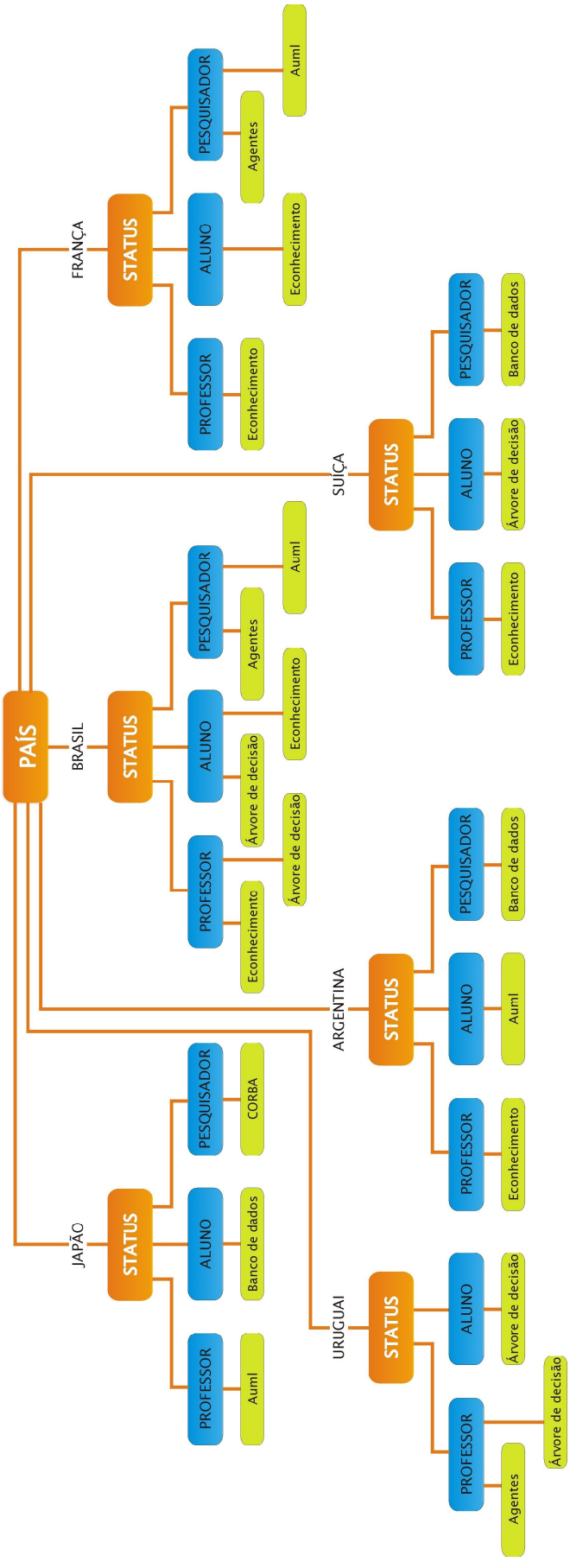

Figura 11 - Árvore de Decisão para o exemplo universidade [57].

Quinlan [56], idealizador dos indutores ID3 e C4.5 baseou-se na teoria de informação, originalmente proposta por Sannon, para a escolha dos atributos durante os testes na construção das árvores de decisão.

Essa teoria está fundamentada em definir a quantidade que pode ter uma mensagem dentro de um universo de informação, ou seja, qual a probabilidade para a ocorrência de cada mensagem num universo de informação.

Voltando-se para a construção de árvores de decisão, a seleção dos atributos a partir da teoria da informação é projetada para minimizar a profundidade da árvore final [61].

A partir dessas considerações e dependendo do indutor aplicado (e.g. ID3, C4.5, CART etc...) a árvore de decisão pode apresentar o formato da Figura 11.

A partir do conjunto de treinamento visto na Tabela 5, aplica-se o algoritmo C4.5, onde primeiramente foi necessária a criação de dois arquivos: universidade.names e universidade.data. O arquivo universidade.names é criado para a definição dos atributos preditivos (i.e.) e o atributo objetivo (i.e. Filtrar ou Não Filtrar), acompanhados dos seus possíveis valores que compõem um exemplo individual (i.e. agentes, EConhecimento etc..), conforme Figura 12.

Filtra, Não Filta.
Status :professor, aluno, pesquisador.
Pais :franca, uruguai, brasil, argentina, suica, japao.
Area de Interesse :agentes, EConhecimento, arvores de decisao,banco de dados,uml,auml,corba.

Figura 12 - Arquivo universidade.names

O arquivo universidade.data é um dataset criado para alocar os dados de cada exemplo, um por linha, como é visto na Figura 13.

```
professor     ,franca      ,EConhecimento      ,Filtra
pesquisador   ,brasil      ,agentes            ,Filtra
aluno         ,uruguai     ,arvores de decisao ,Filtra
pesquisador   ,brasil      ,auml               ,Filtra
aluno         ,franca      ,EConhecimento      ,Filtra
aluno         ,argentina   ,uml                ,Nao Filtra
professor     ,uruguai     ,agentes            ,Filtra
professor     ,brasil      ,EConhecimento      ,Filtra
Pesquisador   ,franca      ,arvores de decisao ,Filtra
pesquisador   ,argentina   ,banco de dados     ,Nao Filtra
aluno         ,brasil      ,arvores de decisao ,Filtra
aluno         ,brasil      ,EConhecimento      ,Filtra
pesquisador   ,suica       ,banco de dados     ,Nao Filtra
pesquisador   ,franca      ,agentes            ,Filtra
professor     ,uruguai     ,arvores de decisao ,Filtra
aluno         ,japao       ,banco de dados     ,Nao Filtra
professor     ,brasil      ,arvores de decisao ,Filtra
pesquisador   ,japao       ,corba              ,Nao Filtra
professor     ,argentina   ,EConhecimento      ,Filtra
aluno         ,suica       ,arvores de decisao ,Filtra
professor     ,suica       ,EConhecimento      ,Filtra
professor     ,japão       ,auml               ,Filtra
```

Figura 13 - Arquivo universidade.data

Após a criação dos arquivos *.names* e *.data*, submeteram-se os dados para que enfim o algoritmo C4.5 pudesse gerar a árvore para o caso universidade, visto na Figura 14.

De acordo com os dados submetidos, as saídas para o exemplo universidade resultaram na árvore de decisão construída observando ainda que o algoritmo representou o atributo área de interesse como o nodo da árvore que permite obter maior informação sobre o conjunto de treinamento.

```
C4.5 [release 8] decision tree generator          Fri Jan  4 00:32:17
1980
          ----------------------------------------

              Options:
                  File stem <universidade>

Read 22 cases (3 attributes) from universidade.data

Decision Tree:

Area de Interesse = agentes: Filtra(3.0)
Area de Interesse = EConhecimento: Filtra(6.0)
Area de Interesse = arvores de decisao: Filtra(6.0)
Area de Interesse = banco de dados: Nao Filtra(3.0)
Area de Interesse = uml: Nao Filtra(1.0)
Area de Interesse = auml: Filtra(2.0)
Area de Interesse = corba: Nao Filtra(1.0)


Simplified Decision Tree:
 Filtra(22.0/7.0)

Tree saved

Evaluation on training data (22 items):

         Before Pruning           After Pruning
        ----------------   ---------------------------
        Size      Errors   Size      Errors   Estimate

           8    0( 0.0%)      1    5(22.7%)    (31.9%)   <<
```

Figura 14 - Árvore gerada pelo Algoritmo C4.5 para o caso universidade.

4.6 Classificador Bayesiano

O Classificador Bayesiano Ingênuo [55] baseia-se no Teorema de Bayes, estando relacionado ao cálculo de probabilidades condicionais. É aplicável, conforme o próprio nome sugere, em tarefas de classificação.

Sejam $X(A_1, A_2,...,A_n, C)$ um conjunto de dados; C1, C2,...Ck, as classes do problema, valores possíveis do atributo, C e R um novo registro que deve ser classificado. Sejam ainda *a1,a2,...,an* os valores que R assume em X.

O classificador Bayesiano Ingênuo possui dois passos:

1. Calcular a probabilidade $P(C=C_i/R), i=1,2,...,k$;
2. Indicar como saída do algoritmo a classe C_i tal que $P(C=C_i/R)$ seja máxima.

O problema reduz-se ao cálculo das probabilidades condicionais $P(C=C_i/R)$, $i=1,2,...,k$. Sabe-se que $P(C=C_i/R)$ pode ser reescrito como: $P(C=C_i/A_1=a_1 \text{ e } A_2=a_2 \text{ e....e } A_n=a_n)$

Por outro lado, pelo teorema de Bayes, como P(A/B)=(P(B/A)*P(A))/P(B), $P(C=C_i/A_1=a_1 \text{ e } A_2=a_2 \text{ e....e } A_n=a_n)= P(A_1=a_1 \text{ e } A_2=a_2 \text{ e....e } A_n=a_n//C=C_i)*P(C=C_i))/ P(A_1=a_1 \text{ e } A_2=a_2 \text{ e....e } A_{ni})$

O denominador na igualdade será sempre o mesmo, independe da classe para a qual a probabilidade esteja sendo calculada. Assim, para fins de comparação entre as probabilidades, o denominador $P(A_1=a_1 \text{ e } A_2=a_2 \text{ e....e } A_{ni})$ pode ser desprezado do cálculo. Dessa forma a expressão se reduz para:

P(A/B)=(P(B/A)*P(A))/P(B), $P(C=C_i/A_1=a_1 \text{ e } A_2=a_2 \text{ e....e } A_n=a_n)= P(A_1=a_1 \text{ e } A_2=a_2 \text{ e....e } A_n=a_n//C=C_i)*P(C=C_i))$

O nome ingênuo no título do método decorre da premissa assumida pelo algoritmo de que os atributos serão sempre independentes entre si, o que, em muitos casos, não deverá ocorrer.

Para ilustrar a aplicação desse método temos o clássico exemplo de "Jogar Tênis", onde o problema tem duas classes Jogar=sim e Jogar=não.

Aparência	**Temperatura**	**Umidade**	**Vento**	**Jogar Tênis**
Ensolarado	Quente	Alta	Fraco	Não
Ensolarado	Quente	Alta	Forte	Não
Nublado	Quente	Alta	Fraco	Sim
Chuvoso	Moderado	Alta	Fraco	Sim
Chuvoso	Fresco	Normal	Fraco	Sim
Chuvoso	Fresco	Normal	Forte	Não
Nublado	Fresco	Normal	Forte	Sim
Ensolarado	Moderado	Alta	Fraco	Não
Ensolarado	Fresco	Normal	Fraco	Sim
Chuvoso	Moderado	Normal	Fraco	Sim
Ensolarado	Moderado	Normal	Forte	Sim
Nublado	Moderado	Alta	Forte	Sim
Nublado	Quente	Normal	Fraco	Sim
Chuvoso	Moderado	Alta	Forte	Não

Tabela 6 – Base de Dados para Classificador Bayesiano. Adaptado de Passos [55].

Considerando o atributo Jogar Tênis. Os atributos são Aparência, Temperatura, Umidade e Vento. Assim pergunta-se: deve-se ou não jogar em dia ensolarado, quente, de alta umidade e vento fraco?

Aplicando o Classificador Bayesiano Ingênuo, tem-se:

P(Jogar=Sim| ensolarado, quente, alta umidade, vento fraco) = P(ensolarado| Jogar=Sim)*P(quente| Jogar=Sim)*P(alta umidade| Jogar=Sim) * P(vento fraco| Jogar=Sim)=0,0071

P(Jogar=Não| ensolarado, quente, alta umidade, vento fraco) = P(ensolarado| Jogar= Não)*P(quente| Jogar= Não)*P(alta umidade| Jogar= Não) * P(vento fraco| Jogar= Não)=0,0274

A resposta do algoritmo seria **Jogar=Não**.

4.7 Clusterização

Clusterização [6] é o processo de Agrupamento de um conjunto de objetos de dados em subconjuntos ou "*clusters*", de tal maneira que elementos em um *cluster* tenham propriedades similares que os distingam dos elementos de outros clusters utilizando alguma medida de similaridade. O objetivo desta tarefa é maximizar similaridade *intracluster* e minimizar a similaridade *intercluster*, ou seja, representem uma configuração em que cada elemento possua uma maior similaridade com qualquer elemento do mesmo *cluster* do que com elementos de outros *clusters*.

Diferente da Classificação que tem rótulos pré-definidos, a clusterização precisa automaticamente identificar os rótulos. Por essa razão, a clusterização é também denominada de indução não supervisionada [6].

Uma etapa inicial na clusterização é a extração de características ou representação de padrões, representados na forma de vetores de atributos ou

pontos em um espaço multidimensional. Em seguida, temos o cálculo de similaridade e o agrupamento.

Em geral, o processo de clusterização requer que o usuário determine qual o número de grupos a ser considerado e, ao se formar os grupos com base nesse número, é possível fazer uma análise dos elementos contidos nele e criar rótulos que representem cada grupo.

De uma maneira mais formal podemos definir Problemas de Clusterização [14] da seguinte forma: Dado um conjunto com *n* elementos *X={X1,X2,...,Xn}*, o problema de clusterização consiste na obtenção de um conjunto de *k clusters*, *C={C1,C2,...,Ck}*, tal que os elementos contidos em um *cluster Ci* possuam uma maior similaridade entre si do que com os elementos de qualquer um dos demais *clusters* do conjunto *C*. O Conjunto *C* é considerado uma clusterização com *k clusters* caso as seguintes condições sejam satisfeitas:

$$\begin{aligned} &\bigcup_{i=1}^{k} C_i = X \\ &C_i \neq \emptyset, \text{ para } 1 \leq i \leq k \\ &C_i \cap C_j = \emptyset, \text{ para } 1 \leq i,j \leq k \text{ e } i \neq j \end{aligned} \qquad (11)$$

O valor de *k* pode ser conhecido ou não. Caso o valor de *k* seja fornecido como parâmetro para a solução, o problema é dito como "*problema de k-clusterização*" [19]. Caso k seja desconhecido, o problema é referenciado como "*problema de clusterização automática*" [15].

Um importante ponto a ser considerado é como medir o quanto um elemento é similar ao outro e verificar se pertence a um determinado cluster ou não. Para tanto, utiliza-se uma medida de similaridade a qual é específica para cada problema de clusterização a ser tratado.

Um dos critérios utilizados para identificar a similaridade é medir a *distância* entre eles, onde a menor distância entre um par de elementos maior é

a similaridade entre eles. Como medidas de distância podem ser citadas a *distância euclidiana* e a *distância "city-block"* [14].

A distância euclidiana, equação 12, considera a distância entre dois elementos X_i e X_j no espaço n-dimensional.

$$d(X_i, X_j) = \left[\sum_{l=1}^{p} (x_{il} - x_{jl})^2 \right]^{\frac{1}{2}} \quad (12)$$

A distância "city-block", equação 13, corresponde a soma das diferenças entre todos o *n* atributos de dois elementos X_i e X_j, não sendo indicada para os casos em que existe correlação entre tais atributos.

$$d(X_i, X_j) = \sum_{l=1}^{p} \left| x_{il} - x_{jl} \right| \quad (13)$$

No processo de clusterização, na busca pela melhor solução no espaço de soluções viáveis, têm sido propostos métodos heurísticos ou aproximados, mas que devido a heterogeneidade dos problemas, não existe uma solução genérica que possa obter bons resultados em todas as aplicações de clusterização.

As heurísticas existentes podem ser classificadas em métodos hierárquicos e de particionamento [19].

Nos algoritmos da clusterização hierárquica, os clusters vão sendo formados gradativamente através de aglomerações ou divisões de elementos/*clusters*, gerando uma *hierarquia* de *clusters*, representados através de uma estrutura de árvore, Figura 15.

Nos algoritmos de aglomeração, que utilizam uma *abordagem bottom-up*, cada elemento do conjunto é, inicialmente, associado a um cluster distinto, e novos *clusters* vão sendo formados pela união dos *clusters*

existentes. Esta união ocorre de acordo com alguma medida que forneça a informação sobre quais deles estão mais próximos uns dos outros.

Nos algoritmos de divisão, com uma abordagem *top-down*, inicialmente tem-se um único cluster contendo todos os elementos do conjunto e, a cada passo, são efetuadas divisões, formando novos *clusters* de tamanhos menores, conforme critérios pré-estabelecidos.

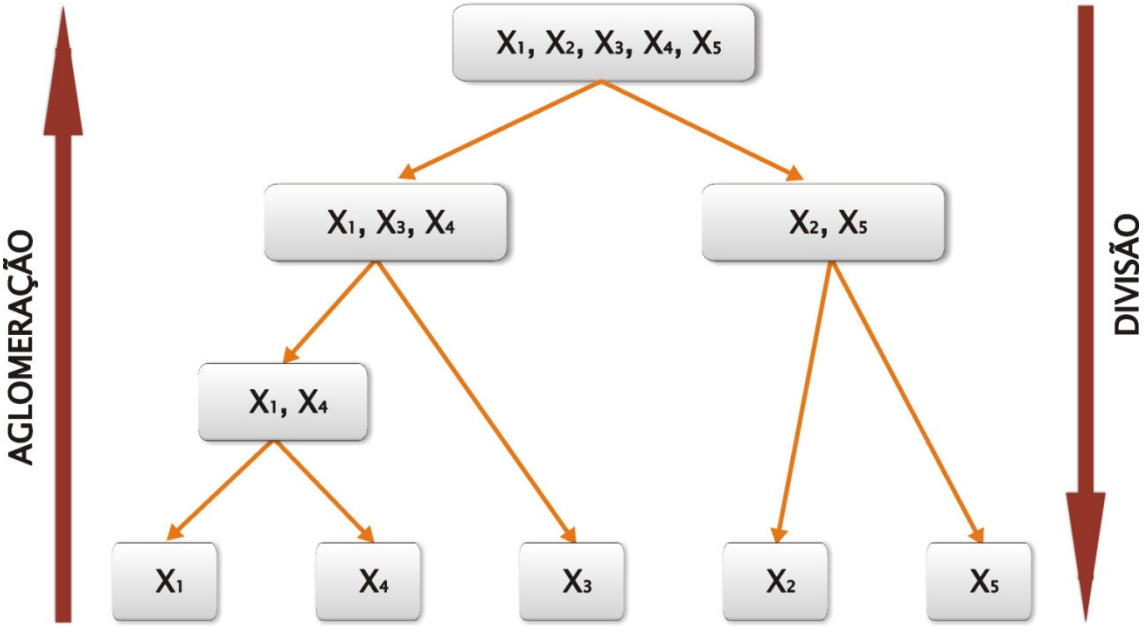

Figura 15 - Exemplo de Árvore de *clusters*. Adaptado de Dias [14].

Nos algoritmos de clusterização que utilizam algum método de particionamento, o conjunto de elementos é dividido em *k subconjuntos*, podendo *k* ser conhecido ou não, e cada configuração obtida é avaliada através de uma função-objetivo. Caso a avaliação da clusterização indique que a configuração não atende ao problema em questão, nova configuração é obtida através da migração de elementos entre os *clusters*, e o processo continua de forma iterativa até que algum critério de parada seja alcançado.

Os métodos de particionamento para *k-clusterização* incluem ainda as técnicas *k-medoids* e *k-means*, de acordo com o tipo de representatividade utilizada para os clusters: no *k-medoids*, o elemento que melhor representa o *cluster*, é definido de acordo com seus atributos sem que haja muita influência

dos valores próximos aos limites do cluster; no *k-means* o elemento representativo de um cluster é o seu centróide, que possui um valor médio para os atributos considerados, relativos a todos os elementos do *cluster*. A utilização do centróide como elemento representativo de um *cluster* é conveniente apenas para atributos numéricos e possui um significado geométrico e estatístico.

A clusterização é muito usada como ferramenta para análise da distribuição dos dados ou como pré-processamento para outros métodos solucionarem determinados problemas.

4.8 Análise Comparativa

Os algoritmos de filtragem são amplamente usados nos sistemas de Filtragem, na tabela abaixo classifica-se as abordagens de Filtragem Baseada em Conteúdo, Colaborativa e Híbrida segundo a análise de Adomavacius [1] em Memória ou Modelo, conforme Tabela 7:

Abordagem de Filtração	Baseado em Memória	Baseado em Modelo
Baseada em Conteúdo	TF-IDF Clustering	Classificadores Bayesianos Clustering Arvores de Decisão Redes Neurais Ontologias
Colaborativa	Vizinhos mais próximos Clustering	Redes Bayesianas Clustering Redes Neurais Modelos Probabilísticos
Híbrida	-	Baseado em item

Tabela 7 – Classificação das abordagens de filtragem

Algoritmos baseados em memória são essencialmente heurísticas que fazem predições de avaliações baseados em uma coleção de itens previamente avaliados pelos usuários enquanto que algoritmos baseados em modelo usam uma coleção de avaliações para aprender um modelo, que é então usado para fazer predições de avaliações [1].

Na Tabela 8, os principais algoritmos usados e estudados neste trabalho são mostrados, destacando suas aplicações, seus pros e contras no uso em aplicações.

Abordagem	Aplicação	Pros	contras
Árvores de Decisão	Classificação de usuário e de itens	resolve os seguintes problemas: minoria, escalabilidade e transparência	Desempenho baixo em árvores muito extensas
Classificadores Bayesianos	Classificação de itens	Tempo de predição é independente do número de exemplos	O naive bayes não modela o texto bem, ele utiliza modelos de texto multinominal, o que não é muito preciso.
Clustering	Classificação e Agrupamento de usuário e de itens	produz *clusters* de qualidade com: Alta similaridade intra-classe; Baixa similaridade inter-classes	Pouca escalabilidade Não refaz o que foi feito previamente (no nível anterior)
Vizinhos mais próximos	Classificação de usuário	Simples e fácil	

Tabela 8 - Tabela comparativa entre abordagens de filtragem.

4.9 Considerações finais

Este capítulo mostrou os principais algoritmos usados nas técnicas de filtragem. Os algoritmos ora considerados foram Booleano, Vetorial, Probabilístico, K vizinhos mais próximo, Árvores de Decisão, Classificador

Bayesiano e Clusterização, mostrando suas principais características e aplicações.

Foi exposta uma tabela comparativa dos algoritmos classificados segunda as técnicas de filtragem e outra com os prós e contras de cada um.

O próximo capítulo trata do ambiente no qual o sistema de filtragem de informação será inserido, o NetClass, como também da aprendizagem colaborativa.

5 APRENDIZAGEM COLABORATIVA APOIADA POR COMPUTADOR

Neste capítulo, apresenta-se o Ambiente *NetClass* [30] que é utilizada para validar a nossa abordagem. Antes, porém, destaca-se as características principais da Aprendizagem Colaborativa e dos Sistemas Tutores Inteligentes, que fundamentam o Ambiente *NetClass*, para se ter com maior clareza a importância da filtragem de informação e, consequentemente, da sua aplicação no processo de ensino-aprendizagem. Além disso, faz-se uma rápida explanação sobre agentes e sistemas multiagentes.

5.1 Aprendizagem Colaborativa

A colaboração é um fator bastante desejável e importante quando um grupo de pessoas visa um objetivo comum. Porém, nas salas de *aula tradicionais*, onde podemos tomar como objetivo comum o aprendizado por parte dos aprendizes, a colaboração ainda não é amplamente utilizada. O que se vê freqüentemente é um comportamento de *competição* entre os aprendizes. O sucesso de um não implica no sucesso de outro. No processo de ensino-aprendizagem tradicional são utilizadas várias metodologias para *repassar* o conhecimento aos alunos, mas, apesar de os alunos comporem um grupo (a classe como um todo), os alunos são avaliados *individualmente*.

Já na *aprendizagem colaborativa*, o sucesso de um aprendiz está correlacionado ao *sucesso do grupo* onde os aprendizes *colaboram* afim de terem um maior aproveitamento no processo de ensino-aprendizagem. Os aprendizes *trabalham juntos* para alcançar um objetivo comum, através da interdependência existente entre eles. Cada membro é responsável pela realização deste objetivo. Para que um grupo seja considerado colaborativo, devem-se considerar os seguintes aspectos [17]:

1. **Interdependência positiva:** é o que diferencia um grupo de aprendizagem colaborativa de um grupo esporadicamente conectado. Este aspecto é intencionalmente planejado de modo que todos os membros devem participar para que a

tarefa seja completada. Há vários tipos de Interdependência Positiva:

a) *Interdependência Positiva do Alvo:* Os alunos percebem que podem alcançar seus alvos de aprendizagem se, e somente se, todos os membros de seu grupo podem também alcançar os seus próprios alvos;

b) *Interdependência Positiva de Recursos:* Cada membro possui só uma parte das informações, dos recursos, dos materiais necessários para a tarefa ser completada, e os recursos dos membros devem ser combinados para que o grupo atinja seu alvo;

c) *Interdependência Positiva de Papéis:* A cada membro se designam papéis complementares e inter-relacionados que especificam as responsabilidades necessárias do grupo para que ele complete uma tarefa conjunta;

d) *Interdependência Positiva de Identidade:* O grupo estabelece uma identidade mútua através de um nome, de uma bandeira, etc.

2. **Responsabilidade Individual e em Grupo:** a responsabilidade individual é a chave para assegurar que cada componente do grupo receba e promova um reforço colaborativo em seu aprendizado [27], ou seja, o indivíduo será reconhecido pela sua contribuição dada ao sucesso do grupo;
3. **Interação Direta:** é essencialmente necessário que haja um auxílio efetivo e eficiente entre os aprendizes. Para isso, deve haver entre os membros dos grupos troca de recursos, assistência e cumplicidade mútua;
4. **Processamento de Grupo:** é necessário saber se as ações dos membros dos grupos foram ou não bem sucedidas a fim

de decidir que ações deverão persistir, quais devem ser aprimoradas e ainda quais devem ser descartadas. O objetivo é verificar a participação ativa dos membros do grupo. Através do processamento de grupo é possível [20]: avaliar a qualidade da interação entre os membros do grupo; examinar o processo pelo qual o grupo trabalha; determinar os objetivos e prover efetividade no grupo; verificar a real situação do grupo no que se refere ao aprendizado.

5. **Habilidade Social:** são evidenciadas todas as formas de iteração entre os membros dos grupos para a realização de determinada atividade.

Considerados esses aspectos, percebe-se que os grupos colaborativos extrapolam o conceito de grupo como um simples agrupamento de aprendizes. Não basta apenas reunir equipes para se estabelecer um processo de ensino-aprendizagem colaborativo.

Vale ressaltar que esta tendência independe do uso de novas tecnologias, exigindo basicamente uma postura pedagógica inovadora e sem preconceitos na qual a educação deve suportar uma aprendizagem baseada na cooperação, colaboração e descobertas [26]. Várias estratégias pedagógicas para a aprendizagem colaborativa podem ser vistas em [18], [30], [51].

5.1.1 O papel do professor na Aprendizagem Colaborativa

Na aprendizagem colaborativa, o professor não se limita apenas a transmitir os conhecimentos aos aprendizes. Além dessa tarefa, essencial e indispensável, o professor deve atuar de forma mais interativa: criando os grupos de aprendizagem e monitorando-os para garantir que cada componente de cada grupo assimile o conhecimento transmitido de maneira colaborativa.

Dentre os diferentes papéis que o professor assume na aprendizagem colaborativa, pode-se destacar [20]:

1. **Criador:** o professor deve criar e manter os aprendizes em uma estrutura colaborativa, com objetivos claros e tarefas bem planejadas, visando favorecer o aprendizado;

2. **Investigador:** o professor deve estar bem informado a respeito dos aprendizes no que diz respeito a habilidades, interesses, necessidades. Para tanto deve acompanhar os grupos de forma ativa inclusive com questionamentos a respeito do que está sendo feito. Assim, ele pode também se auto-avaliar na medida em que estabelece comparações entre o resultado previsto e o alcançado;
3. **Facilitador:** o professor deve estar sempre apto a intervir nos trabalhos dos grupos quando necessário. Ele deve ser visto pelos aprendizes como um membro que interage, reforça, questiona, esclarece. Além disso, é responsável por controlar possíveis conflitos.

O papel do professor pode ainda ser dividido em cinco fases [17]:

1. Especificar os objetivos da sessão de ensino-aprendizagem;
2. Distribuir os grupos antes das sessões;
3. Explicar a estrutura e o objetivo das tarefas a serem executadas;
4. Monitorar a colaboração nos grupos e intervir quando necessário;
5. Avaliar e auxiliar os aprendizes nas discussões.

A seguir, apresenta-se a aprendizagem colaborativa apoiada por computador (ACAC) e como a tecnologia tem sido empregada para fortalecer o uso dos conceitos aqui mostrados.

5.2 Aprendizagem Colaborativa Apoiada por Computador

A ACAC, surgida como uma subdivisão da área de Trabalho Colaborativo Apoiado por Computador, tem sido largamente utilizada desde os anos 80, favorecida pelo desenvolvimento de novas tecnologias de comunicação e designa uma abordagem que visa ampliar a concepção do computador como uma ferramenta, colocando-o como um meio facilitador da aprendizagem [37].

O crescente avanço da tecnologia, principalmente no que diz respeito às comunicações, tem tornado possível a implementação de

ambientes computacionais que permitem e que incentivam uma forte interação entre grupos através de redes [31], tais como os ambientes de aprendizagem colaborativa. Estes ambientes, cada vez mais presentes e requisitados, possibilitam aos aprendizes atuar de forma coletiva, seja resolvendo um problema específico, seja trocando idéias e opiniões ou ainda tomando decisões.

Ele visa uma maior interação entre alunos, professores e o próprio sistema, oferecendo ferramentas e recursos amigáveis que possam estimular o uso do sistema, como por exemplo, chat, fórum, troca de mensagens entre os alunos, que proporciona um aprendizado ainda mais eficiente pelos alunos.

Apresenta-se a seguir algumas características do uso do computador no processo de ensino-aprendizagem:

• Disponibilidade e repetição: O aluno sempre terá à sua disposição o computador, desde que tenha acesso de alguma forma (em casa ou na escola); ao contrário do professor, que raramente pode passar muito tempo com o mesmo aluno. Ele poderá ainda repetir uma lição quantas vezes forem necessárias, visando o seu aprendizado;

• Autonomia do aluno: A aula se faz no ritmo e desejo do aluno. Ele pode recorrer ao computador sem a necessidade de pedir que o professor repita várias vezes o assunto que ele não entendeu. Além disso, é livre para praticar a atividade que estiver à sua disposição, dentro de uma sessão de aprendizagem: entrar em sala de “bate-papo” com outros alunos, participar de fóruns e bancos de dúvidas relacionados à aula, enviar mensagens a outros aprendizes e outras formas de interação, além de ter certa liberdade, até certo ponto, de escolher qual lição estudar e como estudar essa lição, com textos, pesquisas personalizadas à internet, vídeos, enfim, o que estiver à sua disposição;

• Diversidade das ferramentas de transmissão: A diversidade das formas de transmissão do conhecimento é permitida através de vídeos, fotos, imagens, animações, além dos tradicionais textos didáticos.

Apesar de todas essas características, caberá principalmente ao professor a passagem de conhecimento e valores éticos, sociais, morais e

pedagógicos aos alunos, além de estímulo do raciocínio, descobertas e curiosidades pelos alunos. Ele é o responsável pelo conteúdo e avaliações fornecidos ao aluno.

5.3 Agentes e Sistemas Multiagentes

Em [50], um agente de software é definido como uma entidade de software autônoma que pode interagir com o ambiente. Esta autonomia significa que ele pode interagir de forma ativa com outras entidades, incluindo humanas, máquinas e outros agentes de software em vários ambientes e sobre várias plataformas.

A simples interação entre agentes não é suficiente para construir uma sociedade de agentes. Para tanto, necessita-se de agentes que podem ser coordenados para cooperação, competição, ou uma combinação de ambos. Esta sociedade de agentes é chamada Sistemas Multiagentes ou MAS (Multi-Agentes Systems). Sistemas Multiagentes, então, são sistemas compostos de agentes coordenáveis e dos relacionamentos que existem entre eles [50].

5.3.1 Agentes

Um agente pode ser definido como uma entidade que executa uma ação sobre algo, produzindo um efeito. Por se tratar de uma definição genérica, é necessário definir o contexto particular de aplicação da definição, para que seja possível especificar o conceito de agente adotado neste trabalho. É importante, então, deixar claro que o conceito adotado refere-se ao universo da ciência da computação.

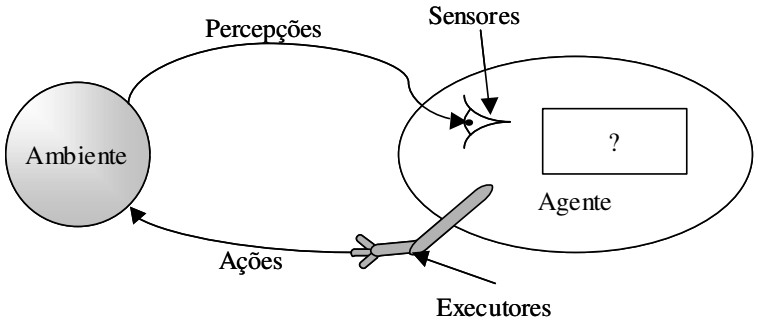

Figura 16 - Interação de agentes com o ambiente

No âmbito da ciência da computação, pode-se falar de agentes como sendo sistemas de computação que executam determinadas tarefas com características que os enquadram como agentes. Agentes humanos são usuários humanos que projetam ou manipulam computadores (hardware ou software).

Russell e Norving [61] afirmam que o trabalho de concepção de um agente inteligente consiste em definir o que será percebido pelo agente no ambiente, suas possíveis ações e, principalmente, os mecanismos através dos quais ele avaliará suas percepções para "escolher" a ação a ser executada, alcançando diferentes graus de autonomia.

Em Wooldrige [74], os autores acrescentaram ao conceito de agentes inteligentes a noção de sociedade, ao afirmar que os agentes inteligentes devem ser capazes de se comunicar, interagindo uns com os outros através de algum tipo de linguagem comum.

Assim, avaliando as definições acima, pode-se dizer que agentes artificiais inteligentes constituem um novo paradigma de concepção de sistemas computacionais, no qual cada sistema deve ser visto como capaz de perceber um ambiente, através de sensores, e agir de forma racional e autônoma sobre esse ambiente. Estes sistemas podem conter mais de um agente, comunicando-se entre si, a fim de alcançarem os objetivos para os quais foram projetados.

Cada uma das definições acima propõe características para o conceito de agente, como, por exemplo, a autonomia, a reatividade, a capacidade social, continuidade temporal, mobilidade e inteligência.

Inteligência é o grau de raciocínio e comportamento aprendido, a habilidade do agente de aceitar a sentença de objetivos (metas) do usuário e desempenhar a sua tarefa.

Autonomia é a capacidade do agente de ter controle sobre suas próprias ações. Um agente autônomo tem a capacidade de decidir como satisfazer o pedido que recebe do usuário.

Pro-atividade é a capacidade do agente de iniciar ações por conta própria para atingir os seus objetivos, não se limitando a responder a estímulos do ambiente.

A continuidade temporal impõe que o agente esteja continuamente ativo no ambiente.

Persistência é a capacidade apresentada pelo agente de manter um estado interno consistente através do tempo, sem alterá-lo ao acaso.

Reatividade é a propriedade que permite aos agentes perceberem seus ambientes e responderem adequadamente às mudanças nelas ocorridas.

Capacidade social é a capacidade do agente de comunicar-se com outros agentes e com seus usuários, geralmente através de uma linguagem de comunicação entre agentes.

Capacidade de adaptação é a capacidade de alterar o seu comportamento com base na experiência, considerada a capacidade de aprendizagem dos agentes.

Mobilidade é a capacidade do agente se mover no seu ambiente.

Os agentes podem ou não possuir uma ou mais capacidades supracitadas a fim de executar suas atividades. Em função disso eles podem ser classificados em como agentes fixos, agentes móveis, agentes inteligentes. Podem ser classificados também de acordo com os tipos de tarefas que executam: agentes de filtragem, agentes de interface, etc.

5.3.2 Tipos de Agentes Artificiais

Para uma melhor compreensão da aplicabilidade da tecnologia de agentes, é interessante falar sobre os diversos tipos de agentes e suas mais variadas diferenças para que tenhamos uma melhor noção de utilidade no emprego de agentes. É possível fazer uma classificação de agentes de acordo com vários aspectos como quanto à mobilidade, quanto ao relacionamento inter agentes e quanto à capacidade de raciocínio.

1. **Agentes Móveis**: são agentes que tem a mobilidade como característica principal. Isto é, uma capacidade de mover-se seja por uma rede interna local (intranet) ou até mesmo pela Web, transportando-se pelas plataformas levando dados e códigos. Seu uso tem crescido devido alguns fatos como uma heterogeneidade cada vez maior das redes e seu grande auxílio em tomadas de decisões baseadas em grandes quantidades de informação;

2. **Agentes situados ou estacionários**: são aqueles opostos aos móveis. Isto é, são fixos em um mesmo ambiente e ou plataforma. Não se movimentam em uma rede e muito menos na Web;
3. **Agentes Competitivos**: são agentes que "competem" entre si para a realização de seus objetivos ou tarefas, ou seja, não há colaboração entre os agentes;
4. **Agentes Coordenados ou Colaborativos**: agentes com a finalidade de alcançar um objetivo maior realizam tarefas especificas, porém coordenando-as entre si de forma que suas atividades se completem;
5. **Agentes Reativos**: é um agente que reage a estímulos sem ter memória do que já foi realizado no passado e nem previsão da ação a ser tomada no futuro. Não tem representação do seu ambiente ou de outros agentes e são incapazes de prever e antecipar ações. Geralmente atuam em sociedades como uma colônia de formiga, por exemplo. Baseiam-se muito também na "teoria do caos" no qual afirma que até mesmo no caos existe uma "certa organização". No caso da formiga, por exemplo, uma única delas não apresenta muita inteligência, mas quando age no grupo comporta-se o todo como uma entidade com uma certa inteligência, ou seja, a força de um agente reativo vem da capacidade de formar um grupo e construir colônias capazes de adaptar-se a um ambiente;
6. **Agentes Cognitivos**: esses, ao contrário dos agentes reativos, podem raciocinar sobre as ações tomadas no passado e planejar ações a serem tomadas no futuro. Ou seja, um agente cognitivo é capaz de "resolver" problemas por ele mesmo. Ele tem objetivos e planos explícitos os quais permitem atingir seu objetivo final. Para que isso se concretize, cada agente deve ter uma base de conhecimento disponível, que compreende todo os dados e todo o "*know-how*" para realizar suas tarefas e interagir com outros agentes

> e com o próprio ambiente. Sua representação interna e seus mecanismos de inferência o permitem atuar independentemente dos outros agentes e lhe dão uma grande flexibilidade na forma de expressão de seu comportamento. Além disso, devido a sua capacidade de raciocínio baseado nas representações do mundo, são capazes de ao mesmo tempo memorizar situações, analisá-las e prever possíveis reações para suas ações.

5.3.3 Sistemas Multiagentes

Sistemas Multiagentes são sistemas constituídos de múltiplos agentes que interagem ou trabalham em conjunto de forma a realizar um determinado conjunto de tarefas ou objetivos. Esses objetivos podem ser comuns a todos os agentes ou não. Os agentes dentro de um sistema multiagente podem ser heterogêneos ou homogêneos, colaborativos ou competitivos, dependendo da finalidade da aplicação que o sistema multiagente está inserido.

Arquiteturas com múltiplos agentes reativos são constituídos por um grande número de agentes. Estes são bastante simples, não possuem inteligência ou representação de seu ambiente e interagem utilizando um comportamento de ação/reação. A inteligência surge conforme os agentes trocam de informações entre si e com o ambiente. Esses agentes não são inteligentes individualmente, mas o comportamento global é.

Já os sistemas multiagentes constituídos por agentes cognitivos são geralmente compostos por uma quantidade bem menor de agentes se comparado aos sistemas multiagentes reativos. Aqueles, conforme a definição de agentes cognitivos, são inteligentes e contêm uma representação parcial de seu ambiente e dos outros agentes. Podem, portanto, comunicar-se entre si, negociar uma informação ou um serviço e planejar uma ação futura.

Esse planejamento de ações é possível, pois em geral os agentes cognitivos são dotados de conhecimentos, competências, intenções e crenças, o que lhes permite coordenar suas ações visando à resolução de um problema ou à execução de um objetivo.

5.4 O Ambiente NetClass

O principal objetivo do NetClass é a concepção de um ambiente de ensino-aprendizagem colaborativo à distância baseado numa arquitetura com múltiplos agentes humanos e artificiais, unindo assim as idéias que fundamentam os STI ao paradigma de Aprendizagem Colaborativa, definindo um Ambiente de Ensino Colaborativo de Ensino-Aprendizagem [33]. Este ambiente integra alunos, professores e sistema computacional num espaço que serve para promover o desenvolvimento de atividades colaborativas, favorecendo a aprendizagem do aluno (individual ou em grupo) através de resolução de problemas, recebendo a assistência do sistema tutor e dos professores, de forma individualizada e adaptada às suas necessidades [32].

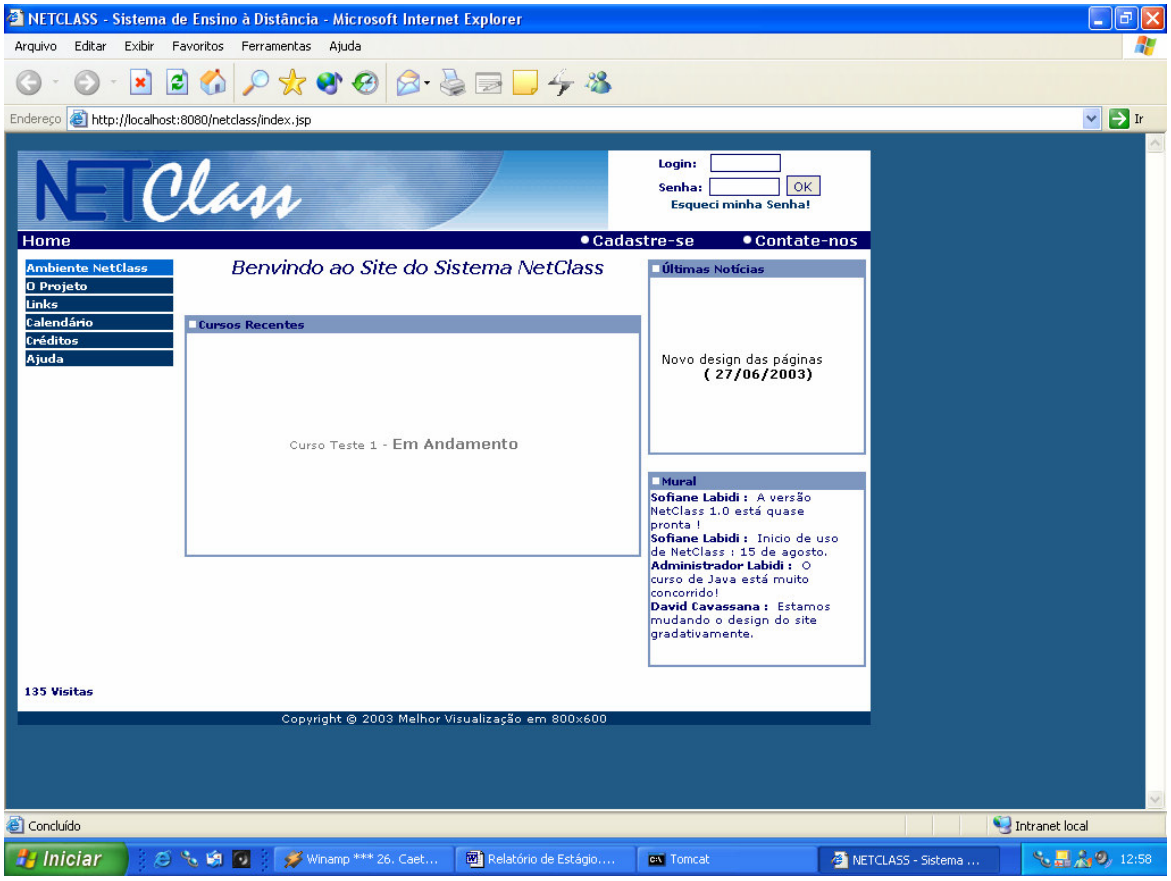

Figura 17 – Interface da página inicial do Ambiente NetClass

No Ambiente NetClass, os alunos são divididos em grupos distintos, chamados de áreas colaborativas. Eles colaboram e aprendem a partir da interação dentro de seus próprios grupos (interação intragrupo) com os

agentes, com o professor, e com os outros grupos (interação intergrupo), através da utilização de recursos multimídia e da tecnologia de redes.

As etapas de aprendizagem possuem diversas atividades que poderão ser desempenhadas pelos alunos e pelos grupos. As atividades de ensino-aprendizagem, no NetClass, são classificadas em seis tipos: preparação de grupos, apresentação do conhecimento, assimilação do conhecimento, aplicação do conhecimento, avaliação de grupo e avaliação individual. Cada tipo de atividade tem funções específicas que são desempenhadas com o uso de estratégias pedagógicas apropriadas, escolhidas de acordo com o modelo do aprendiz [65].

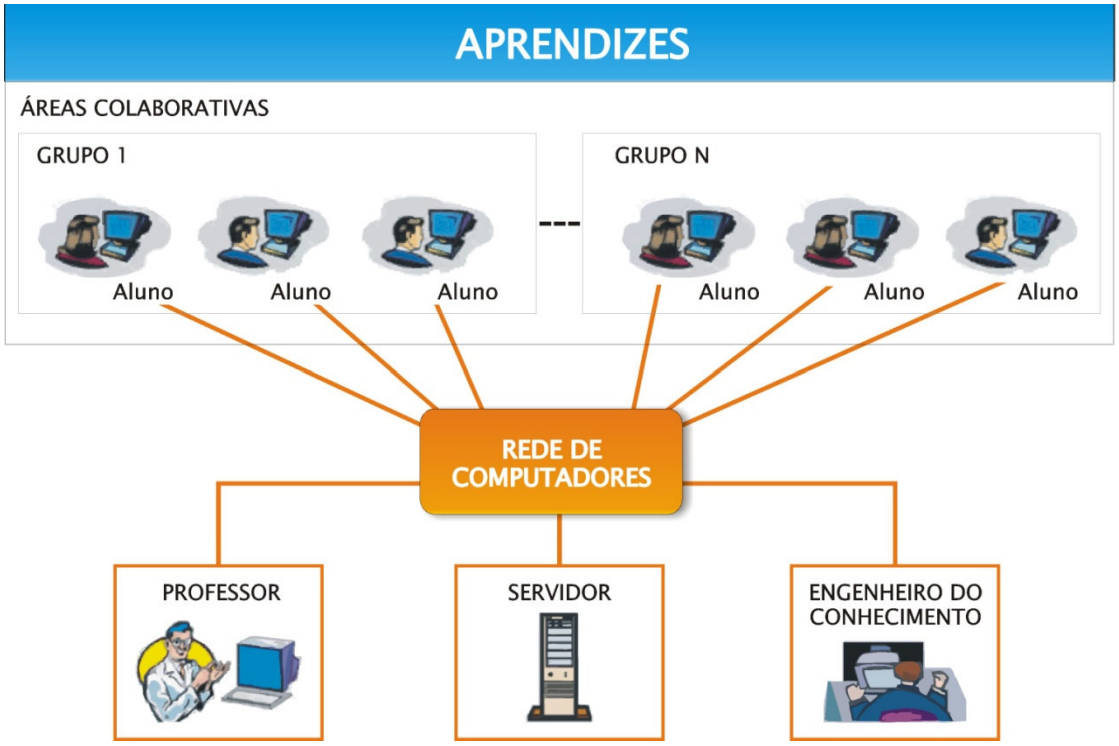

Figura 18 - Ambiente NetClass [32]

Esses alunos podem estar separados fisicamente e, sendo assim, poderão realizar as interações através de recursos de comunicação de rede (salas de conversação, vídeo-conferência, correio eletrônico, banco de dúvidas, etc.). Dessa maneira, os aprendizes podem realizar atividades sugeridas pelo sistema ou professor, resolver problemas, solicitar informações ou tirar dúvidas, entre outras atividades.

Além do aluno e do professor, outras três entidades participam do processo de ensino aprendizagem dentro do Sistema Netclass:

• o monitor, que auxilia o professor na transmissão do conhecimento aos alunos, monitorando o desempenho dos grupos de estudo;

• o coordenador, responsável pelo andamento geral do curso, desde sua criação, inclusão de módulos aos cursos, seleção dos alunos e outras tarefas relativas ao curso.

• o administrador responsável pela administração do sistema como um todo.

Quanto à estrutura e funcionamento geral, o sistema está dividido em cursos, e estes podem conter um ou mais módulos, todos cadastrados por um coordenador específico. Cada módulo é ministrado por um professor, que pode ser auxiliado por um monitor que ele escolhe. É nos módulos em que se encontra todo o conteúdo acessado pelos alunos, e é dentro dele que os alunos interagem entre si. O curso, ao ser aprovado pelo administrador do sistema, possibilita aos alunos se inscreverem. Uma vez inscrito no curso, o aluno deve ainda se matricular nos módulos, com o decorrer do curso. Uma vez inscrito, este terá acesso ao conteúdo do módulo (aulas, atividades, etc) e poderá interagir com os outros alunos do módulo (e-mail, chat, fórum, mural, etc.).

O sistema é inteiramente baseado na Web, permitindo que pessoas o acessem de qualquer lugar com acesso à Internet e a condição para utilização é que o usuário esteja cadastrado e autenticado no sistema. Todas as notificações informativas aos alunos sobre o andamento do curso, as solicitações dos alunos, e informações a respeito das aulas e atividades dos módulos se dão através de e-mails enviados aos alunos automaticamente pelo sistema.

5.4.1 Arquitetura Multiagente do NetClass

O ambiente NetClass é baseado em uma arquitetura multiagente composta pelos seguintes agentes: agente tutor, agentes de domínio, agente de modelagem do aprendiz, agente estrategista, agente de busca e humanos

(alunos/grupos, engenheiro do conhecimento e professor) [30]. Na seção 6, a sociedade de agentes de filtragem, objeto de estudo deste trabalho, é descrita.

5.4.1.1 Agentes do Ambiente NetClass

A arquitetura NetClass é composta por vários agentes humanos e artificiais. Na Figura 19, os principais agentes que compõem essa arquitetura são mostrados. Além destes, outros agentes secundários fazem parte do sistema.

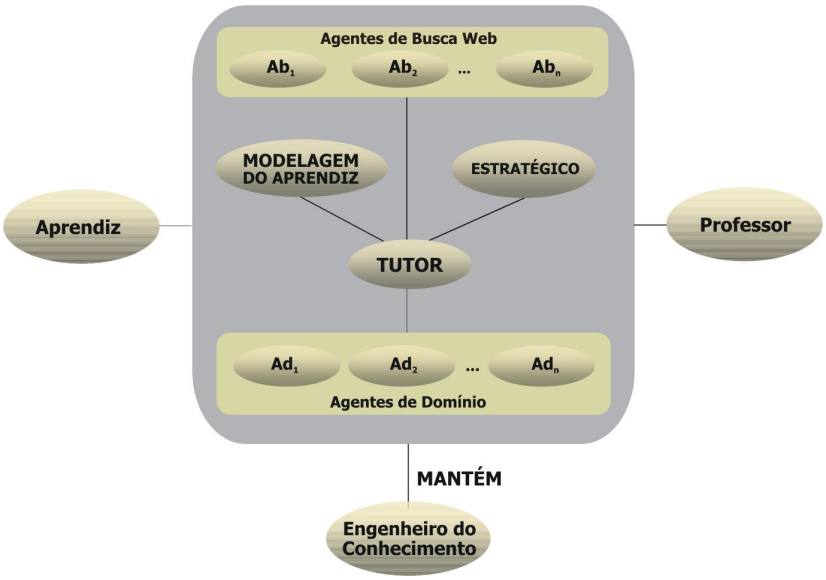

Figura 19 - Arquitetura multiagentes NetClass [32]

a) Agentes Artificiais

- **Agente Tutor**: participa de todas as atividades de ensino. Ele interage com os principais agentes da arquitetura. a fim de apresentar aos aprendizes o conteúdo e tarefas adequados para cada um deles. É responsável pelo controle das interações no sistema durante o processo de ensino-aprendizagem. Além disso, fornece dicas e ajuda o aprendiz no momento da aplicação do assunto;
- **Agentes de Domínio:** um Agente de Domínio é responsável pela representação de um conhecimento específico, ou melhor, representação de um subdomínio do domínio em estudo. Na realidade, existem vários Agentes de Domínio, todos possuindo a mesma estrutura, diferenciando-se uns dos outros apenas pelo seu

conhecimento específico. Eles mantêm *links* para recursos armazenados no servidor. Esses recursos podem disponibilizar vários tipos de mídia;

- **Agentes de Busca Web:** um Agente de Busca Web, também chamado simplesmente de Agente de Busca, é um correspondente de um Agente de Domínio, representando Índices Web do seu subdomínio. Os Agentes de Busca mantêm *links* para recursos localizados na Web e da base de dados do NetClass;
- **Agente de Modelagem do Aprendiz**: é responsável pelo processo de aquisição, representação e manutenção de informações sobre aluno e grupos durante o processo de ensino-aprendizagem;
- **Agente Estrategista**: interage com o Agente Tutor a fim de definir as estratégias pedagógicas mais adequadas a serem adotadas em suas atividades. Por exemplo, o Agente Estrategista decide quais unidades de conhecimento e em que formato o conteúdo será apresentado aos aprendizes.

b) Agentes Humanos

- **Professor**: agente humano que pode assumir diferentes papéis no sistema, dentre os quais, destacam-se os de especialista, orientador e avaliador. Entre outras funcionalidades, o professor poderá: i) Fornecer ao engenheiro do conhecimento o conteúdo a ser ensinado aos aprendizes; ii) Com o auxílio do sistema, formar grupos e reorganizá-los, quando for necessário; iii) Modificar a estratégia pedagógica adotada; iv) Supervisionar e interagir com as áreas colaborativas, monitorando a apresentação dos conteúdos, discutindo e esclarecendo dúvidas dos aprendizes; v) Avaliar os aprendizes, tendo como base as informações do modelo do aprendiz;
- **Aprendizes**: são os alunos, os quais podem ser organizados em grupos. Cada aluno necessita está inserido em um grupo para que possa participar das sessões de aprendizagem. Ele está ligado ao sistema através de um computador e pode estar ou não separado fisicamente dos demais integrantes do seu grupo;

- **Engenheiro do Conhecimento**: é responsável pela manutenção dos agentes da arquitetura. Ele inclui e realiza a edição do conhecimento de cada Agente de Domínio e a organização de seu respectivo subdomínio.

5.5 Considerações finais

Neste capítulo abordou a aprendizagem colaborativa apoiada por computador. Mostrou também alguns conceitos sobre agentes e sistemas multiagentes.

Apresentou ainda o ambiente colaborativo de ensino-aprendizagem, NetClass, no qual está inserido este trabalho.

O próximo capítulo trata da modelagem do sistema, que serão inseridos no NetClass.

6 MODELAGEM DO SISTEMA

O principal objetivo do sistema é atender às necessidades de informação dos alunos/usuários do NetClass numa área de interesse específica associada à disciplina estudada. O sistema realiza quatro atividades principais: recuperação de informação a partir de uma consulta [49], filtragem baseada em perfis, descobrimento de informação e atualização da base de informação. A filtragem é realizada de duas maneiras: através da requisição do aluno em interface especifica ou automaticamente pelo sistema.

A arquitetura proposta é denominada de MAFIS (*Multiagent Filtering Information Systems*) e é um Sistema de Filtragem de Informação Multiagente para um ambiente de ensino colaborativo, no caso o NetClass, que tem por objetivo principal prover informações aos seus usuários (alunos) associados à disciplina estudado no ambiente colaborativo de ensino.

O sistema foi modelado através da metodologia de desenvolvimento de agentes PASSI [54] e implementado utilizando a linguagem Java e JADE. As próximas seções descrevem o funcionamento de cada um dos módulos do sistema.

Este capítulo dedica-se a apresentação de uma proposta do modelo, onde primeiro listam-se alguns requisitos que devem ser alcançados com o mesmo no ambiente de ensino NetClass. Em seguida, discorre-se sobre a visão em camadas do modelo proposto detalhando o papel de cada camada. Por fim, descrevem-se os agentes e como é alcançado cada um dos requisitos listados.

6.1 Requisitos

Neste subtópico, lista-se os requisitos que devem ser alcançados com a modelagem do MAFIS que utiliza a metodologia PASSI para o seu desenvolvimento sobre a plataforma do NetClass, e foi desenvolvido segundo a abordagem de agentes.

Um requisito é uma condição ou habilidade necessária para um sistema alcançar um determinado objetivo ou finalidade. O objetivo de todos sistema é atender a um conjunto de requisitos – as necessidades que o sistema deve satisfazer [35].

6.1.1 Requisitos do MAFIS

O MAFIS objetiva a filtragem de informação existentes na base de dados do NetClass ou na Internet aos alunos/usuários cadastrados no sistema

Desse modo existem três requisitos principais e dentro destes os secundários. O primeiro diz respeito ao próprio usuário e cuida da forma como são adquiridas suas preferências, e como é modelado, com base nessas preferências, o seu perfil. Isso para que se possa oferecer uma filtragem efetiva e com itens relevantes ao mesmo.

O segundo requisito é relacionado à atividade de representação dos documentos e do monitoramento constante destas informações, provenientes de uma fonte (Base de Dados do NetClass ou da web) para a verificação de novos elementos que potencialmente satisfaçam os interesses de cada usuário.

Os requisitos secundários podem ser listados abaixo:

- Interagir com o usuário com um meio para o recebimento das requisições ou interesses do usuário e entrega das informações;
- Modelar as necessidades de acordo com o interesse do usuário (perfil);
- Filtrar informações para atender às necessidades do usuário;
- Recuperar informações pra atender a consultas do usuário;
- Monitorar informações em suas bases de dados ou na internet, com informações atualizadas;
- Indexar fontes de informação, afim de facilitar o processo de recuperação;

6.2 Base de Dados

Os itens a serem filtrados aos alunos provêm de dois locais específicos: parte provem da base de dados do próprio NetClass e outro da web indexados pelo agente de busca [49]. A base de dados do NetClass é compota por diversos arquivos nos mais variados formatos: doc, pdf, rft, jpg, gif, etc.

Na indexação, para ambas as bases de dados, é criado um vetor de termos que representam o conteúdo, no caso de arquivos textuais, são compostos por uma coleção de termos, com um valor de peso associado a cada termo calculado pela freqüência relativa de cada termo no texto - TF (*Term-frequency*) - e/ou pela freqüência inversa de termos IDF (*Inverse-Document-Frequency*).

6.3 Agentes de Filtragem de Informação

O principal objetivo do sistema de filtragem é atender às necessidades de informação dos alunos usuários do NetClass numa área de interesse específica associada à disciplina estudada. O sistema realiza quatro atividades principais: recuperação de informação a partir de uma consulta utilizando-se o trabalho realizado por Oliveira [49], filtragem baseada em perfis, descobrimento de informação e atualização da base de informação. A filtragem é iniciada de duas maneiras: através da requisição do aluno em interface especifica ou automaticamente pelo sistema.

Para alcançar este objetivo o SFI precisa realizar um conjunto de tarefas que visa facilitar o processo de filtragem de informação: indexação, registro de conteúdo, comparação e análise de similaridade e ao final avaliação dos itens filtrados.

Os SFI atendem às necessidades de informação a longo prazo e tratam de interesses específicos de um ou mais usuários similares denominado de perfil que também é usado na descoberta de novas informações. Estas informações são geralmente feitas localmente na base de dados do NetClass ou na internet, que contém uma grande quantidade de dados não estruturados ou semi-estruturados.

Para gerar um perfil inicial do usuário (o aluno), será necessário que o professor que ministrará determinada disciplina através do NetClass cadastre a mesma e os seus tópicos principais gerando explicitamente um perfil. Os perfis são representados através de vetores de termos do tipo P_u=*{área, disciplina, sub-tópico}*. A partir daí o sistema de busca do NetClass dará início a uma busca de itens, procurando inicialmente o primeiro tópico abordado na disciplina criando um índice de conteúdo. Como ainda não existe uma avaliação destes itens por nenhum dos usuários do sistema, estes serão comparados ao perfil do aluno através das técnicas de FBC.

O aluno ao receber os itens indicados deverá dar uma nota de 0 a 5 avaliando o seu grau de interesse pelo item filtrado. Dessa maneira poderá ser, tanto atualizado seu perfil, quanto usadas as técnicas de FC para filtrar itens de maneira mais efetiva a partir da similaridade entre outros alunos.

No decorrer da aprendizagem, o Agente Tutor poderá, a qualquer momento, solicitar do agente de filtragem uma relação de itens sobre o assunto que está sendo estudado pelo aprendiz. Ao final do processo o Agente de Filtragem informa ao Agente Tutor quais são os resultados da filtragem e este é mostrado na interface do aprendiz como uma relação de *links*. A interface do aprendiz permite que o mesmo possa navegar pela relação de *links* exibidos e também fará a avaliação dos itens.

Com esta abordagem híbrida consegue-se suprimir as limitações existentes na FBC e FC descritas na seção 3. As próximas sub-seções descrevem a modelagem e implementação do protótipo do sistema.

6.4 Descrição dos Agentes

O sistema é composto de sete agentes: o agente modelador, agente de busca, agente de recuperação, agente de indexação, agente de interface, agente de filtragem e agente de atualização descritos a seguir.

Cada camada descrita na arquitetura, na seção 6.5, é composta por um ou mais agentes que cooperam entre si para prover as funcionalidades pertinentes a cada camada. A seguir, detalham-se os agentes.

6.4.1 Agente de Busca

É o responsável pela busca de informações, buscando novos itens de informações pertinentes ao perfil ou consulta dos usuários como também do perfil de filtragem. Ele *monitora* constantemente as fontes de informação da web e da base de dados do NetClass descobrindo novas informações passíveis de filtragem.

Qualquer requisição de informação gerada pelo usuário ou pelo próprio sistema gera uma requisição ao Agente de Busca. O Agente de Busca é um agente do tipo reativo, ou seja, age em função das mudanças na fonte de informação.

Para a atividade *do Agente de Busca* de descoberta de novas informações inicialmente é realizada a atividade *descobrir elementos de informação*. De posse destes elementos ele realiza a atividade *enviar elementos descobertos*.

6.4.2 Agente de Modelagem

É o responsável pela modelagem do perfil dos usuários do sistema representando seus interesses, através da técnica do modelo vetorial, como também é responsável pelo gerenciamento dos mesmos.

As atividades do *Agente de Modelagem* são *receber informações dos usuários* a partir do Agente de Interface e *receber informações filtradas* que podem ser feitas em paralelo. As duas primeiras atividades também são enviadas ao Agente de Avaliação para atualização dos perfis que realiza a atividade *atualizar perfil*. A última é seguida das atividades *criar* e *manter modelo de usuário* e *enviar informações dos perfis*.

6.4.3 Agente de Recuperação

É o responsável pela recuperação de documentos na base de dados e na web.

6.4.4 Agente de Indexação

É o responsável pela criação da representação dos documentos da base de dados e/ou recuperados da web, através da técnica do modelo vetoria utilizando a norma entre a freqüência de termos e a freqüência inversa, como também é responsável pelo gerenciamento dos mesmos. O agente representa os mesmos em vetores.

O *Agente de Indexação* inicia suas atividades com receber *representação de documentos* para que seja realizada a atividade *indexar documentos*. Durante um processo de recuperação ou de filtragem, por exemplo, este agente então realiza as atividades *recuperar índices* e *enviar índices*.

6.4.5 Agente de Interface

É o agente de interface simplificado para interagir com os demais agentes do sistema, recebendo requisições dos usuários e entregando resultados aos usuários. É através deste agente que o usuário pode avaliar os itens filtrados para gerar seu perfil colaborativo. Neste agente é usado o padrão façade cuja intenção do uso é fornecer uma interface simplificada para um conjunto de interfaces em um subsistema, definindo assim uma interface de nível mais alto que torne o subsistema mais fácil de ser usado.

6.4.6 Agente de Filtragem

É o responsável pela filtragem dos itens utilizando as técnicas baseada em conteúdo e colaborativa. Faz a análise de similaridade entre usuários ou entre documentos e usuários de acordo com a estratégia do sistema e fornece os documentos filtrados com o máximo de relevância.

Para o *Agente de Filtragem* as atividades consistem em *receber representação de documento* e *receber perfil*, que ativam o processo de filtragem por meio da realização das atividades *fazer análise de similaridade da filtragem* e *comparar representação de documento* e *perfil*. Após a realização da filtragem, o agente *envia elementos filtrados*.

Utiliza uma abordagem híbrida para a filtragem, usando a filtragem baseada em conteúdo e colaborativa, que é mais detalhada no subtópico 6.6 através da Figura 21.

6.4.7 Agente de Avaliação

É o responsável pela atualização dos perfis dos usuários e dos grupos de usuários. O interesse dos usuários (perfil) não são constantes e para modifica-los é necessário que o agente possa escrever as novas informações.

Quando recebe a informação a ser armazenada e verifica se já existe, este armazena o dado e retorna uma mensagem de sucesso à entidade que solicitou a escrita ou modificação.

Todas as operações que alterem o estado das informações do perfil do usuário (atualização, inserções, exclusões) são executadas pelo Agente de Avaliação.

6.5 Definição da Arquitetura de Agentes em Camadas

Com esta abordagem baseada em agentes, o sistema é capaz de agir com independência e é organizado em camadas. Cada uma das camadas é composta por uma sociedade de agentes que juntos cooperam. O módulo de filtragem, proposto neste trabalho foi desenvolvido com a intenção de filtrar informações aos usuários do NetClass. Apresenta-se, na Figura 20, o modelo em camadas do sistema de filtragem.

No modelo é mostrado ainda os agentes que compõem cada camada do modelo, e como acontece a interação entre eles.

O detalhamento das responsabilidades de cada camada que compõe a arquitetura do MAFIS é descrita a seguir.

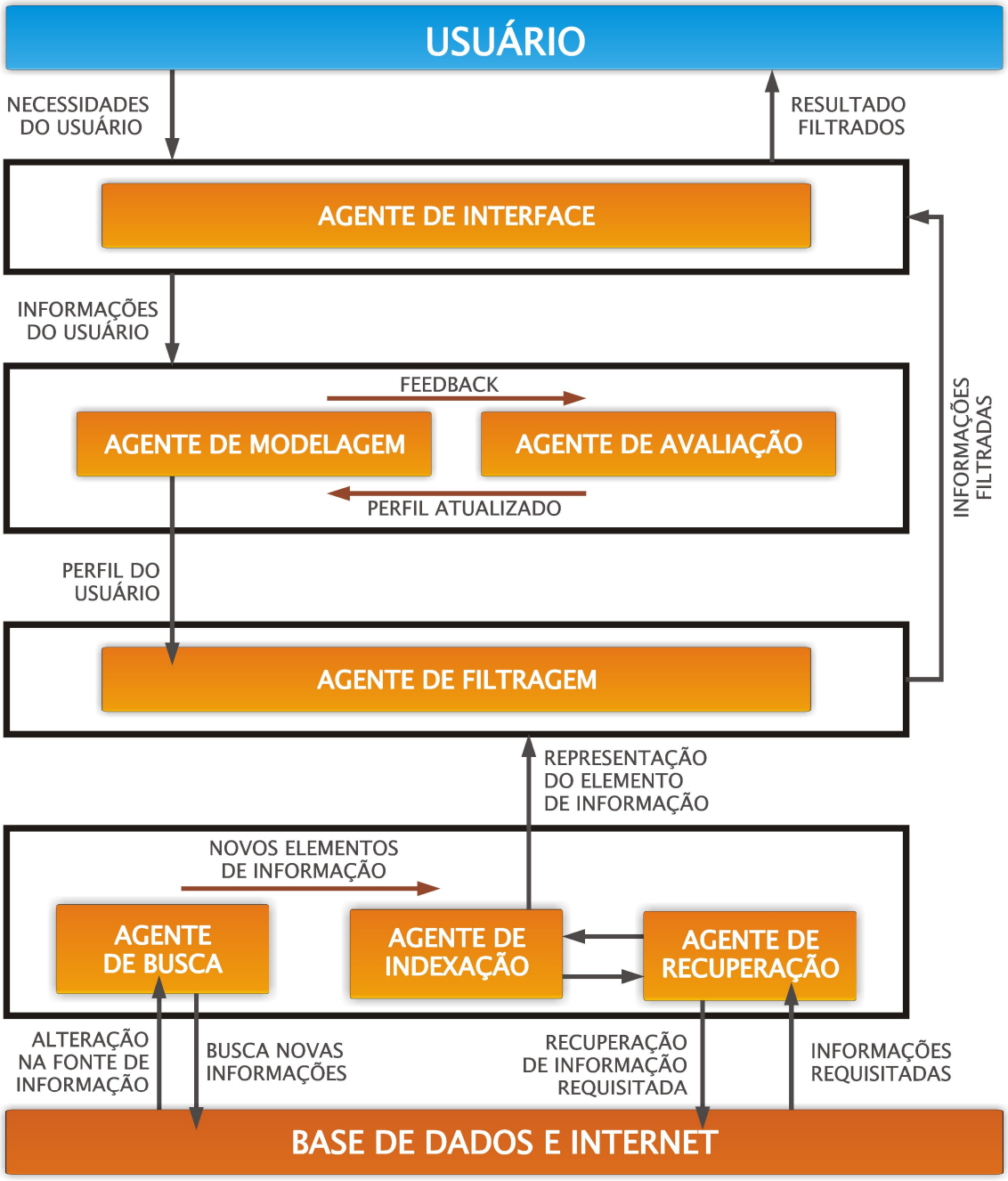

Figura 20 - Arquitetura em Camadas da Filtragem de Informação

6.5.1 Camada de Interação com o Usuário

Recebe as necessidades do usuário, que corresponde aos seus interesses representados através do perfil e entrega resultados filtrados, de acordo com os interesses anteriormente especificados.

É composta pelo Agente de Interface que recebe as necessidades de informação do usuário e envia informações filtradas ao mesmo. Utiliza

informações obtidas com o usuário e envia informações aos agentes *de Modelagem* e *Avaliação* da camada inferior.

6.5.2 Camada de Modelagem de Usuário

Para que o sistema seja capaz de efetuar a filtragem é necessário compor um *modelo* de usuário caracterizado pelo seu perfil que recebe informações de suas necessidades e as caracteriza em seus registro como vetores de termos para efetuar o calculo de similaridade.

Esta camada utiliza as informações do usuário para a representação e manutenção do perfil do usuário com base em seus interesses além de atualizar o perfil de usuário.

É composta pelos agentes de *Modelagem*, responsável pela modelagem do usuário e do agente *de Avaliação*, responsável pela sua atualização.

6.5.3 Camada de Filtragem de Informação

Gera informações filtradas, comparando os perfis com os modelos de documentos ou perfis com grupo de perfis de usuários.

A camada superior envia informações com os perfis dos usuários através do *Agente de Modelagem*. Os perfis são utilizados para gerar a filtragem utilizando algum algoritmo de filtragem.

Para que o sistema seja capaz de efetuar a filtragem é necessário ainda receber do *Agente de Indexação* a representação dos documentos indexados para então comparar este com o perfil do usuário, através do calculo de similaridade, caso seja utilizado a filtragem baseado em conteúdo.

É composta pelo agente de Filtragem, responsável pela filtragem e do envio das informações filtradas ao *Agente de Interface*.

6.5.4 Camada de Monitoramento e representação de Informação

Camada que engloba a detecção qualquer alteração da base de dados, criando a representação de documentos (modelo de documentos) dos novos elementos de informação.

Para tanto, constitui-se dos agentes *Agente de Busca* que é responsável pela descoberta de novos elementos de informação na base de dados do NetClass ou da Web e do envio destes ao *Agente de Indexação* responsável pela indexação dos elementos encontrados.

6.6 O modelo do MAFIS

O modelo foi criado tentando minimizar as limitações derivadas de cada técnica estudada.

Para os documentos a representação dos mesmos foi feita através do modelo vetorial, este modelo propõe um ambiente em que cada documento é visto como um vetor de termos e a cada termo é associado um grau de importância (peso) deste no documento**,** ou seja, cada documento possui um vetor de pesos associado na seguinte forma: *(t1,w1),(t2,w2),...,(tn,wn)*, onde ***t*** é o termo e ***w*** é o seu peso do documento, que é descrito a partir do seguinte pseudocódigo:

- Escolha da coleção de documentos
- Definição de um universo de termos.
- Para cada documento da coleção:
    1. Remove-se a pontuação
    2. Remove-se stopwords
    3. Aplica-se stemming
    4. Calcula-se a freqüência tf de cada termo do vetor de palavras
    5. Calcula-se a idf para cada termo do vetor de palavras
    6. Calcula-se o peso de cada termo do termo

7. Forma-se o vetor de pesos do documento

Onde,

tf – representação do número de vezes que aparece no documento

idf – relação entre os termo e os documentos da coleção

O cálculo do peso é feito através do produto entre o *tf* e o *idf*. Um documento é representado da seguinte maneira: `Doc[0]=[data, defined, filtering, information, type, typical]`, onde após calcular a freqüência das palavras de cada documento de acordo com do vetor do universo de palavras chaves e definição das entradas é criado o seguinte vetor `Doc[0]= [[filtering, 6.0], [information, 7.0], [retrieval, 2.0], [system, 2.0], [text, 2.0]]`.

Para os usuários é criado o seu perfil a partir do conteúdo cadastrado pelo professor utilizando-se também o modelo vetorial

Quando o sistema ou o usuário solicita a filtragem é feito o cálculo de similaridade a representação do documento e o perfil do usuário através do cosseno.

O pseudocódigo abaixo descreve como é feito o cálculo da similaridade:

1. Escolhe-se o universo de termos, n= x.
2. Determina-se as entradas
    a. Vetor de pesos dos documentos
    b. Vetor de peso do perfil.
3. Calcula-ser a similaridade através do cosseno.
4. Cria-se uma lista ordenada de documentos relevantes.

Este lista de itens ordenada é entregue ao usuário para que o mesmo use e/ou avalie os mesmos para gerar um perfil colaborativo, já que o primeiro só poderá ser usado para a filtragem baseada em conteúdo.

A partir deste perfil é criado um modelo do usuário (MU) para compor um modelo de grupo de usuários (MGU) onde, para realizar a filtragem colaborativa é realizada o cálculo de similaridade entre um usuário ativo *x* com o MGU clusterizado, através de uma técnica de clusterização..

O algoritmo usado para comparação entre o usuário e o modelo de grupo de usuário para classificá-lo em um grupo específico é o KNN, onde é feito o cálculo de similaridade entre um usuário ativo e modelo de grupo de usuários para classifica-lo em um determinado grupo (clustering), e assim filtrar informações a este usuário.

Este lista de itens é entregue ao usuário para que o mesmo use e/ou avalie os mesmos para atualizar seu perfil colaborativo e assim criar perfis mais efetivos que possam descrever a real necessidade do usuário.

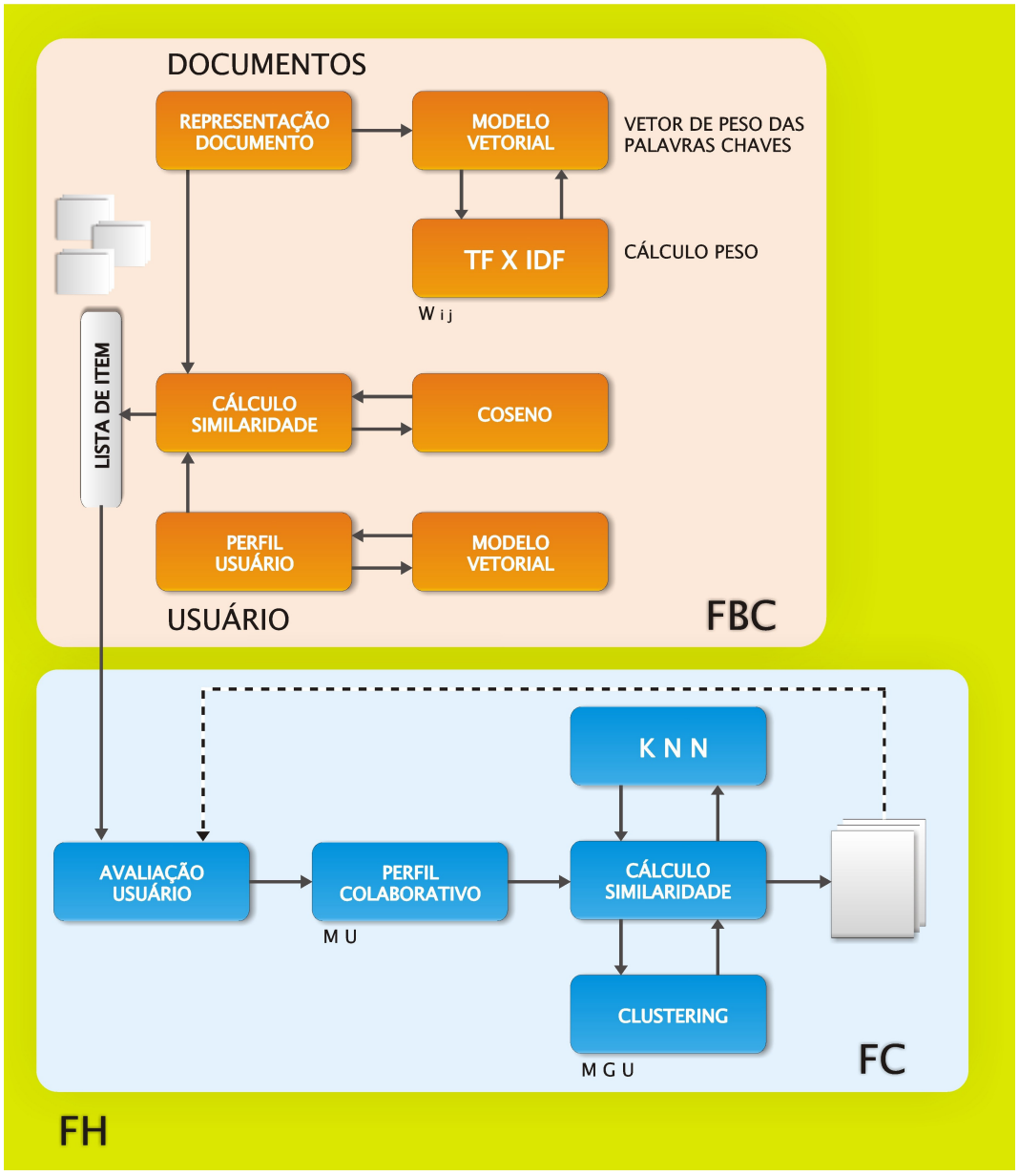

Figura 21 – Filtragem Híbrida do MAFIS

6.7 Considerações finais

Neste capítulo foi descrito a modelagem do MAFIS, sistema multiagente de filtragem de informação, descrevendo quais agentes compõe o sistema e como estão distribuídos em camadas.

Foram apresentados também os requisitos do sistema.

Propõe-se um modelo híbrido de filtragem de informação através da combinação da filtragem baseada em conteúdo e colaborativa.

O próximo capítulo trata da implementação do sistema descrevendo as tecnologias usadas e mostrando os principais diagramas do sistema.

7 IMPLEMENTAÇÃO DO PROTÓTIPO

O modelo apresentado para o Sistema de Filtragem de Informação (MAFIS) constitui-se de uma proposta genérica aplicada a um ambiente de ensino aprendizagem colaborativo,o NetClass. Neste contexto, descreve-se o estado da implementação do protótipo do mesmo.

7.1 Modelagem e Ambiente de Desenvolvimento

A modelagem do MAFIS utiliza a metodologia PASSI para o seu desenvolvimento sobre a plataforma do NetClass, e foi desenvolvido segundo a abordagem de agentes.

Para o desenvolvimento dos agentes utilizou-se como plataforma de implementação o JADE (*Java Agent Development Framework*).

7.2 PASSI

PASSI (*Process for Agent Societies Specification and Implementation*) [54] é uma metodologia que aborda as fases de análise e projeto para desenvolvimento de sistemas multi-agentes, integrando a definição da modelagem e a filosofia de SMA como também de orientação a objetos, utilizando UML.

Esta metodologia é composta de cinco modelos que endereçam diferentes visões e doze passos durante seu processo de desenvolvimento de Sistemas Multiagentes. A PASSI adota a UML como linguagem de modelagem, por esta ser aceita amplamente em ambientes acadêmicos e industriai, evitando impactos na produtividade decorrente da adoção de uma linguagem completamente nova [54].

Os modelos existentes nesta metodologia estão dispostos da seguinte forma: *Modelos de requisitos do sistema, sociedade dos agentes, implementação dos agentes, código e distribuição*.

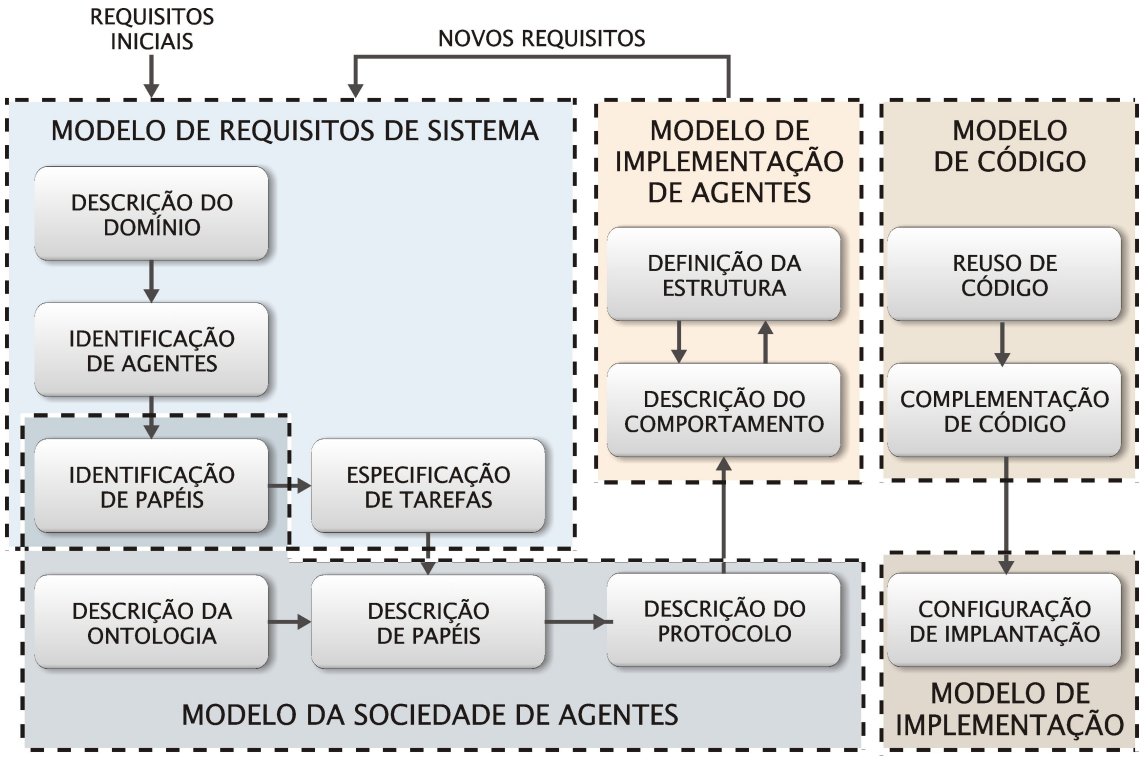

Figura 22 – A metodologia PASSI [54]

O modelo relacionado aos requisitos do sistema compreende quatro passos: descrição do domínio, identificação dos agentes, identificação dos papéis e especificação das tarefas, descritos abaixo:

- **Descrição do Domínio** (*Domain Description*): uma descrição funcional do sistema usando diagrama de casos de uso. Os cenários de detalhamento dos diagramas de caso de uso são explicados usando Diagrama de Seqüência;
- **Identificação de Agentes** (*Agent Identification*): A identificação dos agentes é feita a partir dos diagramas de caso de uso da fase anterior, onde as responsabilidades são separadas por agentes, sendo que agentes são identificados como pacotes decompostos por um ou mais casos de uso. Um diagrama de pacotes representando todo o sistema ou parte deste é o resultado desta fase;
- **Identificação de Papéis** (*Role Identification*): Nesta fase são construídos diagramas de seqüência com a finalidade de explorar as responsabilidades ou papéis de cada agente em

cada cenário específico, identificados no diagrama de caso de uso;

- **Especificação de Tarefas** (*Task Especification*): Nesta fase é desenhando um *diagrama de atividades*, sendo que cada *diagrama* representa uma *tarefa* que um agente pode realizar;
- **Descrição de Ontologia** (*Ontology Description*): Nesta fase são construídos os *diagramas de classe* descrevendo o conhecimento das entidades envolvidas (agentes, atores) e seus relacionamentos. O conhecimento dos agentes é descrito através de cada classe e o conteúdo das mensagens através de associações entre as classes.

7.3 PASSI ToolKit (PTK)

A PTK (PASSI ToolKit) [12] é uma ferramenta especialmente concebida para prestar total suporte à metodologia PASSI, sendo composta conceitualmente por duas partes completamente integradas: um plugin para o Rational Rose destinado à construção dos modelos; e uma ferramenta Java, que permite o reuso de padrões tanto na plataforma JADE quanto no FIPA-OS através da representação do código dos agentes em uma meta-linguagem baseada em XML.

A opção por uma ferramenta CASE comercial baseada em UML como o Rational Rose se deu pelo fato de ser ela amplamente conhecida e utilizada, o que ajuda a minorar a dificuldade que os iniciantes no paradigma multiagente enfrentam ao projetar novos sistemas, mesmo sendo experientes, por exemplo, na orientação a objetos.

Especificamente, a ferramenta realiza operações de verificação baseadas na correção de diagramas isolados e na consistência entre passos e modelos relacionados. As suas principais funcionalidades são:

- compilação automática de diagramas;
- suporte automático à execução de operações recorrentes;
- consistência de projeto;

• compilação automática de relatórios e de documentos de projeto;

• acesso a bases de dados de padrões;

• geração de código e engenharia reversa.

7.4 JADE

Como plataforma para implementação do MAFIS utilizou-se o JADE (Java Agent Development Framework) [25]. A escolha se deve ao fato deste ser um framework totalmente implementado em JAVA e de acordo com as especificações FIPA [21], que simplifica o desenvolvimento de sistemas multiagentes.

Além disso, JADE é um software livre e distribuído em código aberto.

7.5 Construção do Sistema

Após a concepção do modelo e a escolha da metodologia a ser utilizada para construí-lo, no caso a PASSI, passou-se a fase de desenvolvimento. Nas subseções seguintes, apresentam-se a definição dos agentes, a definição da ontologia da sociedade de agentes e as responsabilidades e tarefas de cada agente. Em seguida, na seção 7.5.6 e 7.5.7, descreve-se a fase de codificação, sobre a plataforma JADE, do MAFIS.

7.5.1 Definição dos Agentes

Segundo a metodologia PASSI, para desenvolvimento de Sistemas Multiagentes, o primeiro passo é a descrição do domínio através do Diagrama de Descrição do Domínio, que contém uma descrição funcional do sistema usando diagrama de caso de uso, como mostrado na Figura 23.

Neste diagrama é mostrado a construção do diagrama, onde cada caso de uso representa uma funcionalidade do sistema. Os atores representam as entidades, componentes internos do MAFIS e entidades externas, que interagem com o sistema em construção.

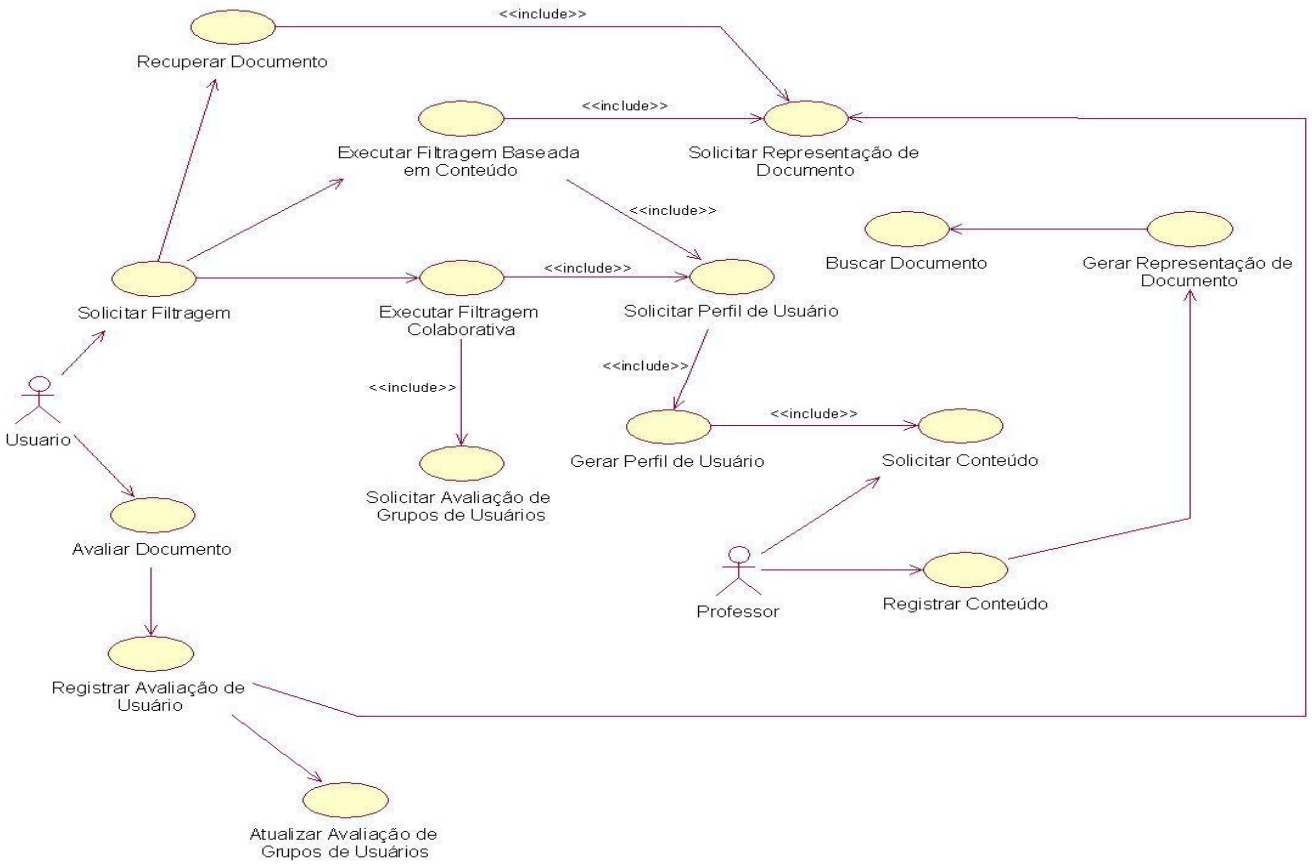

Figura 23 - Diagrama de Descrição do Domínio

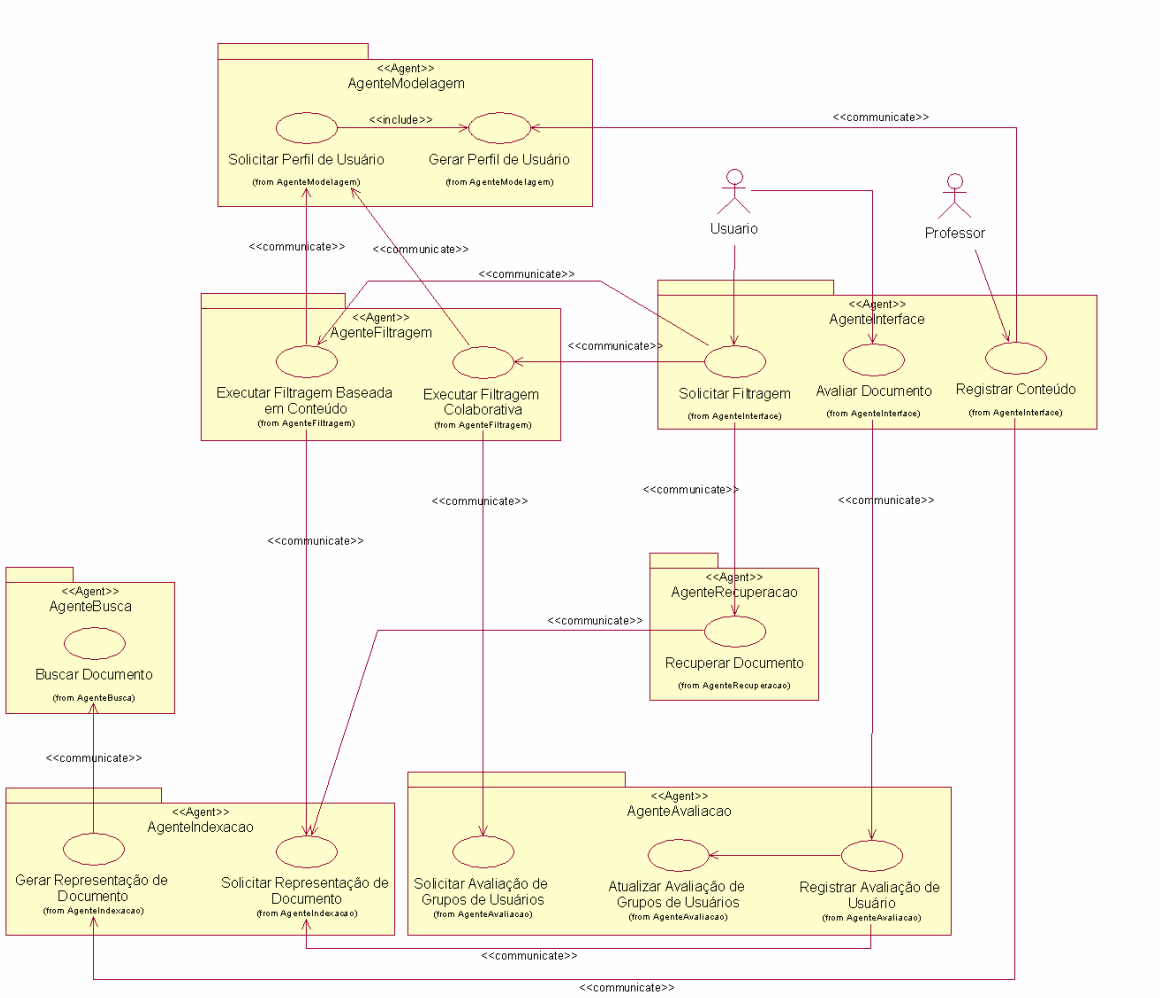

Figura 24 - Diagrama de Classes da Fase de Identificação de Agentes

Assim descrevem-se os requisitos em termos de diagrama de caso de uso. Como resultado, a fase de descrição de domínio é uma descrição funcional do sistema composta de diagramas de caso de uso usando notação UML.

O segundo passo, Figura 24, é a Fase de Identificação dos Agentes onde é feita a partir dos diagramas de caso de uso da fase anterior, sendo que agentes são identificados como **pacotes** decompostos **por um ou mais casos de uso**. As **entidades externas** que interagem com o sistema são representadas como **atores**. A interação entre agentes e atores pode ser descritas como atos de comunicação e para alcançar seus objetivos os agentes atuam e interagem com outros atores ou agentes.

Neste Diagrama mostra-se de forma macro como os agentes do MAFIS se relacionam e suas atribuições principais.

7.5.2 Definição da Ontologia

Para cada comunicação entre agentes é preciso especificar três elementos: ontologia, linguagem e protocolo de interação. Enquanto várias linguagens e protocolos de interação são padronizados pela FIPA, a ontologia, foi definida como conseqüência da aplicação específica.

Para detalhar a ontologia concebida para compor a solução, usou-se um diagrama de classe chamado Diagrama de Descrição de Ontologia. Neste Diagrama descreve-se a ontologia do domínio, onde cada classe representa uma entidade.

A ontologia é composta de conceitos, ações e predicados utilizados pelo domínio [54], os quais são descritos abaixo:

- **Conceitos**: representados no diagrama pelas chaves em amarelo, um conceito representa uma entidade chave dentro do domínio que possui uma estrutura complexa e que pode ser definida em termos de atributos.
- **Ações**: é um tipo especial de conceito que indica uma ação que pode ser executada por um agente. As ações são

geralmente executadas em reação a uma mensagem recebida ou após a avaliação de um predicado.

- **Predicados**: São expressões que dizem algo sobre o estado do domínio. Essas expressões são sempre avaliadas como verdadeiras ou falsas.

7.5.3 Descobrindo as Responsabilidades ou Papéis dos Agentes

A Identificação de Papéis é o próximo passo onde são construídos diagramas de seqüência com a finalidade de explorar as responsabilidades de cada agente em cada cenário específico, identificados no diagrama de caso de uso. Está baseada na exploração de todos os possíveis caminhos do Diagrama de Identificação de Agentes, Figura 25, que envolve a comunicação inter-agente. Cada comunicação descreve um cenário onde agentes interagem e trabalham para alcançar um comportamento exigido pelo sistema. Este comportamento é composto de várias relações de comunicação.

Estes cenários são extraídos por meio de diagramas de seqüência, gerando o Diagrama de Identificação de Papéis, nos quais são usados objetos para simbolizar as responsabilidades. Cada agente pode participar em diferentes cenários com diferentes responsabilidades, assim como, em um mesmo cenário participar com responsabilidades distintas.

Nas Figura 26, Figura 27 e Figura 28, mostram-se os principais cenários utilizados para descobrimento das responsabilidades dos agentes que compõem o MAFIS. Cada objeto nos diagramas representa um papel e são nomeados da seguinte forma: *<nome_papel>: <nome_agente>*.

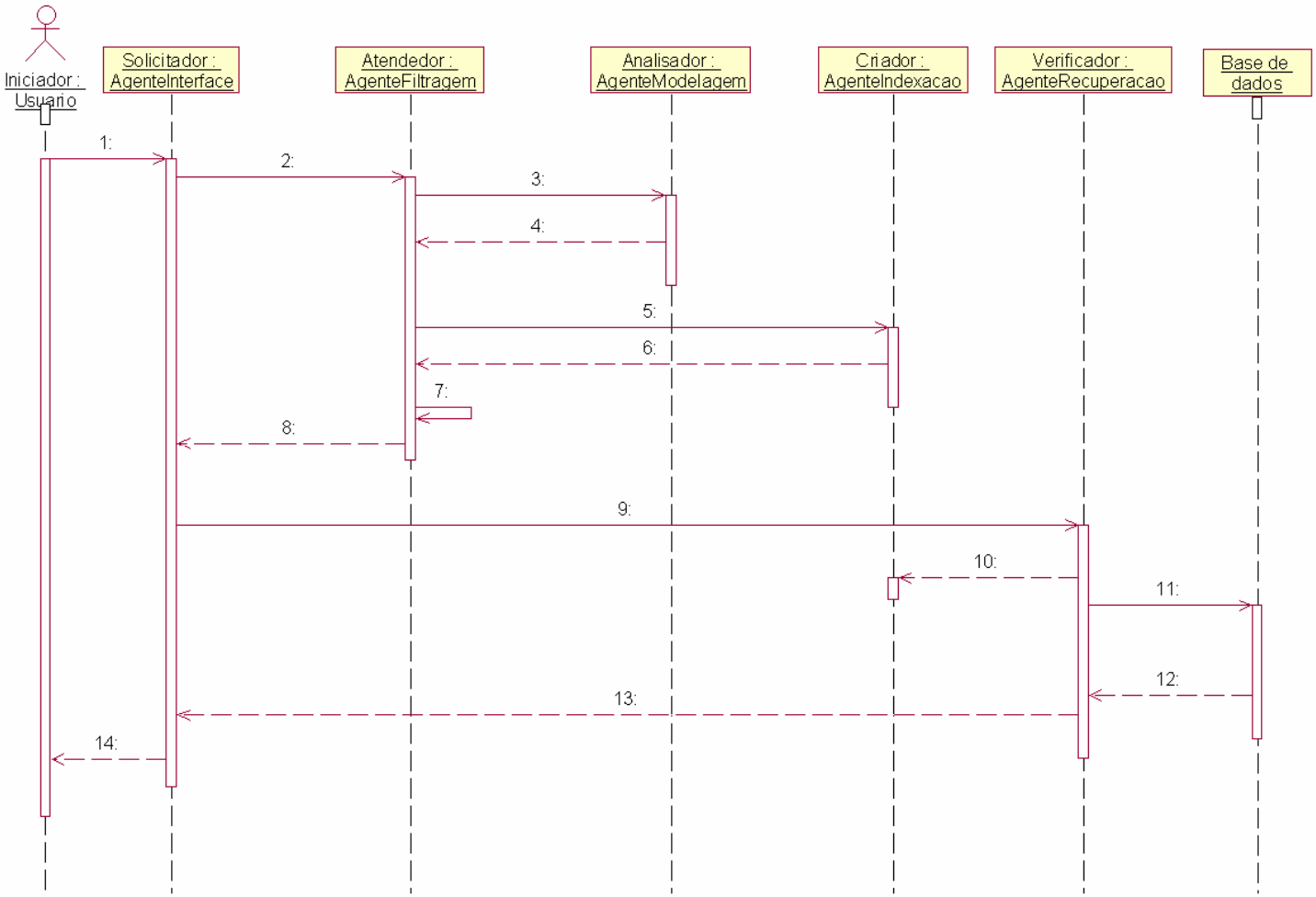

Figura 25 - Diagrama de Interação da Fase de Identificação de Papéis. Cenário: Filtragem Baseada em Conteúdo.

Mostra-se, na Figura 25, o cenário da Filtragem Baseada em Conteúdo. Neste cenário, interagem o Agente de Filtragem, Agente de Modelagem, Agente de Indexação, Agente de Recuperação o Agente de Interface para enviar ao usuário os documentos ou itens que sejam similares ao seu perfil.

O Diagrama de Seqüência descrito na Figura 25 desenvolve-se da seguinte forma: (1) O usuário faz uma solicitação de filtragem através do Agente de Interface, (2) que envia a requisição ao Agente de Filtragem, (3) que faz a requisição do perfil do usuário ao Agente de Modelagem, (4) este por sua vez envia o perfil do usuário ativo ao Agente de Filtragem, (5) logo após o Agente de Filtragem solicita ao Agente de Indexação a Representação dos documentos indexados, (6) e os envia ao Agente de Filtragem, que faz a (7) análise de similaridade entre o perfil e a representação dos documentos indexados e (8) envia os índices dos documentos filtrados ao Agente de Interface, que então (9) solicita ao Agente de Recuperação os documentos filtrados, (10) o Agente de Recuperação verifica a lista dos índices filtrados e (11,12) busca na base de dados os mesmos, (13) estes são enviados ao Agente de Interface e (14) entrega ao usuário.

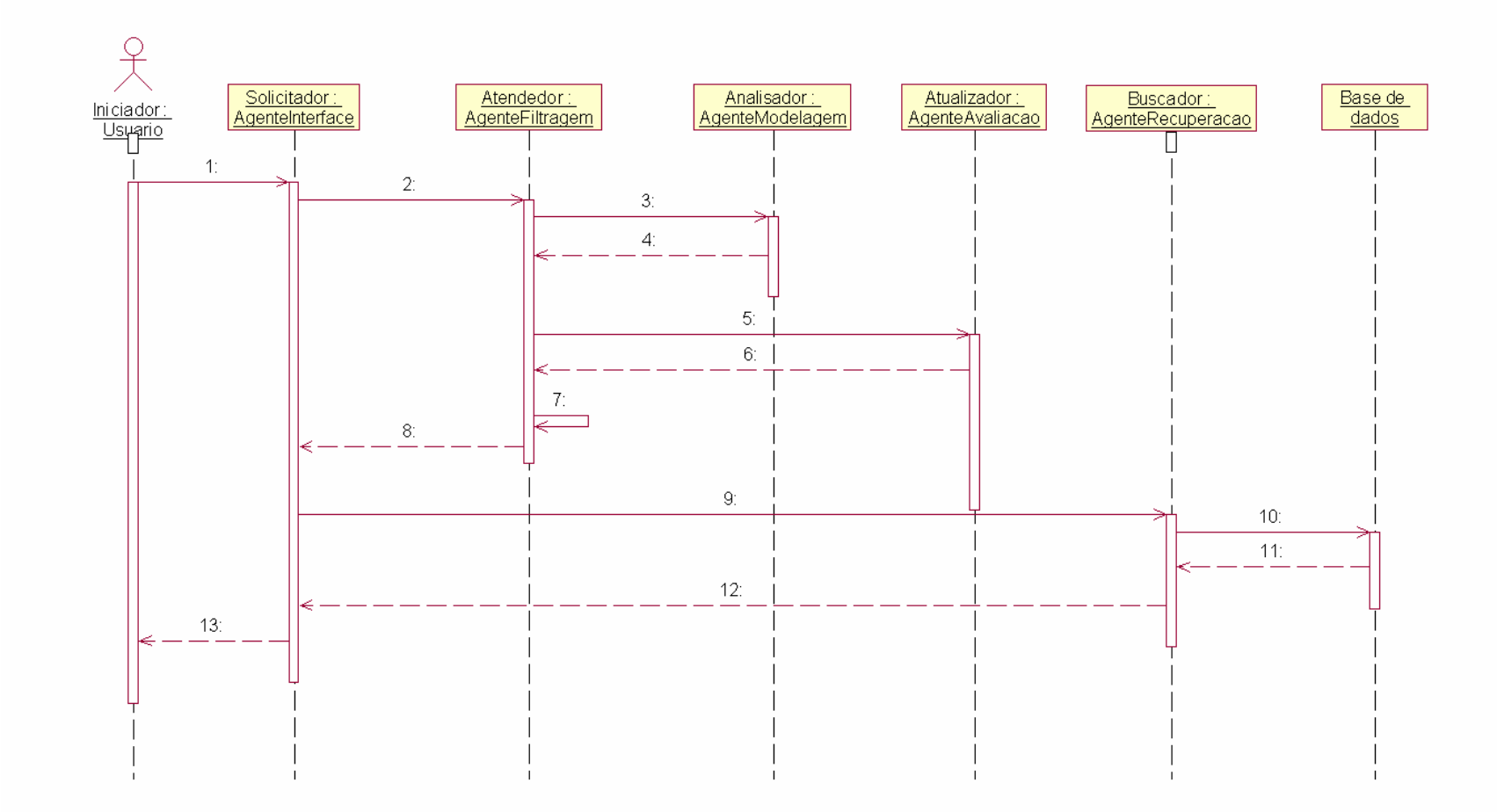

Figura 26 - Diagrama de Interação da Fase de Identificação de Papéis. Cenário: Filtragem Colaborativa.

O Diagrama de Seqüência representado na Figura 26 serve para ilustrar como ocorre a Filtragem Colaborativa, no qual aparecem os agentes os mesmos agentes da FBC a exceção do Agente de Indexação que é substituído pelo Agente de Avaliação. Ë importante ressaltar que aqui é feita a similaridade entre os perfis dos usuários.

O cenário descrito na Figura 26 desenvolve-se da seguinte forma: (1) O usuário faz uma solicitação de filtragem através do Agente de Interface, (2) que envia a requisição ao Agente de Filtragem, (3) que faz a requisição do perfil do usuário ao Agente de Modelagem, (4) este por sua vez envia o perfil do usuário ativo ao Agente de Filtragem, (5) logo após o Agente de Filtragem solicita ao Agente de Avaliação o Modelo de Grupo de Usuário, (6) este envia-os aos Agente de Filtragem, que faz a (7) análise de similaridade entre o perfil do usuário ativo com o modelo de grupos de usuários armazenados e o classifica e agrupa-o em um grupo especifico e (8) envia os índices dos documentos filtrados ao Agente de Interface, que então (9) solicita ao Agente de Recuperação os documentos filtrados, (10) o Agente de Recuperação verifica a lista dos índices filtrados e (11) busca na base de dados os mesmos, (12) estes são enviados ao Agente de Interface e (13) entrega ao usuário.

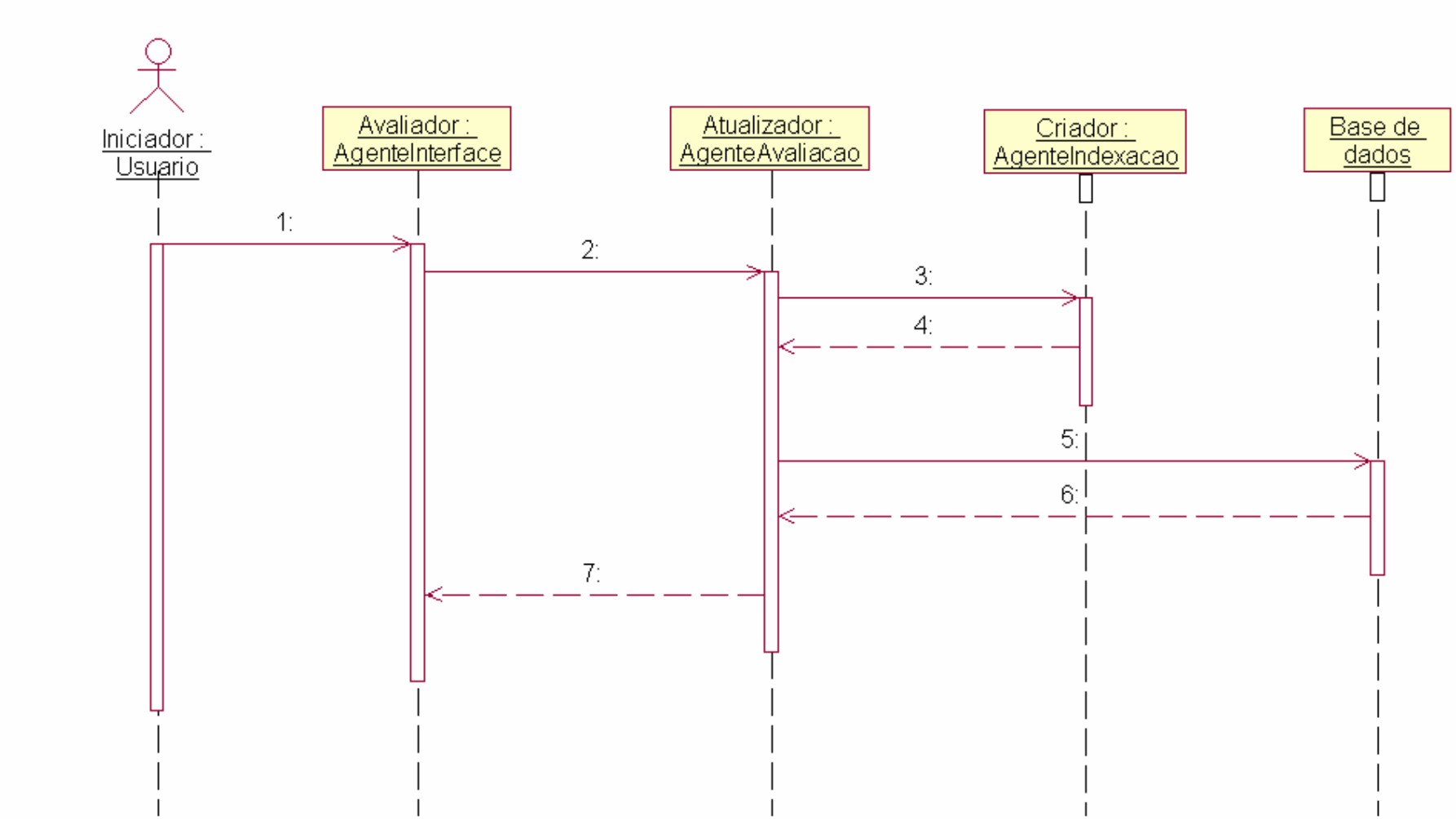

Figura 27 - Diagrama de Interação da Fase de Identificação de Papéis. Cenário: Avaliação de Documentos.

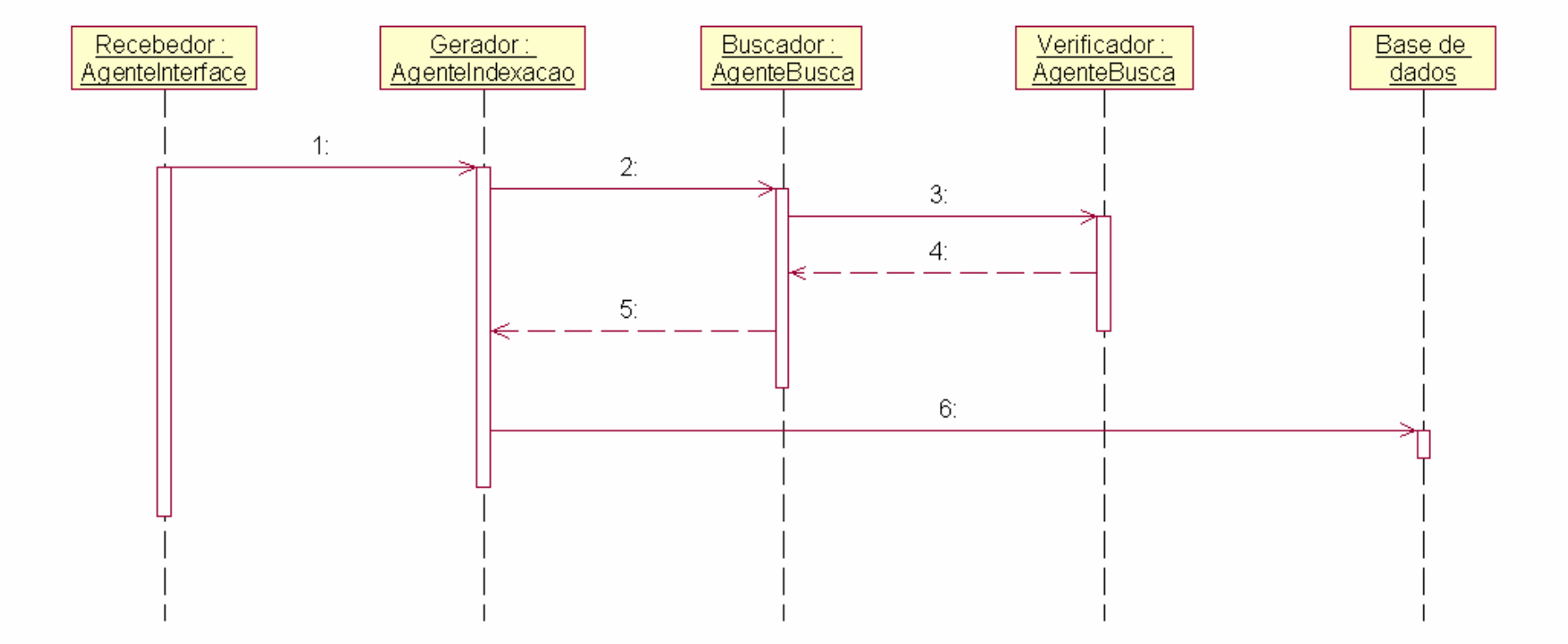

Figura 28 - Diagrama de Interação da Fase de Identificação de Papéis. Cenário: Registrar Conteúdo.

O quarto cenário, Figura 28, Registrar Conteúdo, mostra o Agente de Indexação com o papel de gerar representação do documento e o Agente de Busca com os papéis Recuperar documentos de uma base externa e buscar documentos, trabalham para registrar o conteúdo cadastrado pelo professor. A descrição da Figura 28 é feita a seguir: (1) o Agente de Interface solicita ao Agente de indexação que este gere a representação de um determinado documento em palavras chave ou vetores, (2) é então solicitado uma recuperação de documentos ao Agente de Busca, que (3) faz a busca na base de dados ou na internet do documento solicitado (4,5) enviando ao Agente de Indexação que faz sua representação e solicita o (6) registro da Representação do documento em base de dados especifica.

7.5.4 Descobrindo as Tarefas de cada Agente

Nesta etapa, deve-se focar no comportamento ordenado de cada agente afim de decompô-lo em tarefas. As tarefas que um agente é capaz de executar denotam a capacidade deste agente. A capacidade de cada agente é especificada através de diagramas de atividades, que são *Diagramas de Especificação de Tarefas* da metodologia PASSI.

Para todos os agentes do modelo, foi desenhado um diagrama de atividade que é composto de duas partes. O lado direito do diagrama contém uma coleção de atividades que simbolizam as tarefa do agente, enquanto que o lado esquerdo contém algumas atividades que representam os outros agentes interagindo com este.

Nos Diagramas de Especificação de tarefas, Figura 29 e Figura 30 resume-se o que o agente é capaz de fazer, ignorando informações sobre quais papéis um agente realiza ao finalizar tarefas em particular.

Para encontrar as tarefas dos agentes observam-se os Diagramas de Identificação de Papéis ou Responsabilidades, explorando então, todas as interações e ações internas que os agentes executam para realizar um cenário específico. Em cada Diagrama de Identificação de Responsabilidades obtêm-se uma coleção de tarefas relacionadas que são agrupadas de forma apropriada.

A seguir ilustram-se na Figura 29 e Figura 30, o detalhamento das tarefas dos agentes Agente de Filtragem - para a filtragem baseada em conteúdo – Agente de Modelagem – para a modelagem do usuário, e Agente de Avaliação. para a atualização do perfil do usuário e do grupo de usuários.

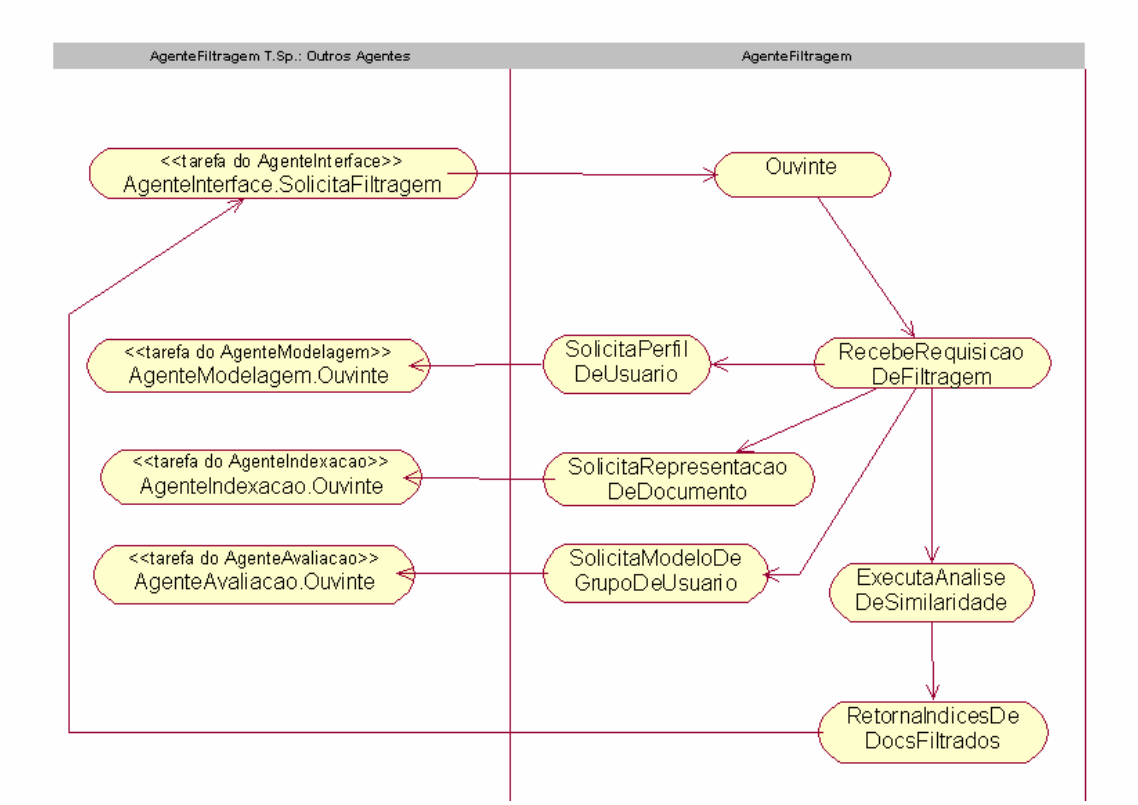

Figura 29 – Agente de Filtragem

A Figura 29 mostra o Diagrama de Identificação de Tarefas do Agente de Filtragem. A tarefa *Requerer Representação de Documento* é necessária para iniciar a Filtragem. Foram identificadas as tarefas que se relacionam e controlam o agente que são *Requerer Pefil do Usuário*,*Calcular Similaridade* e *Gerar Vetor de Documentos*.

Na Figura 30, a geração do perfil do usuário e o seu registro na base de dados são mostrados.

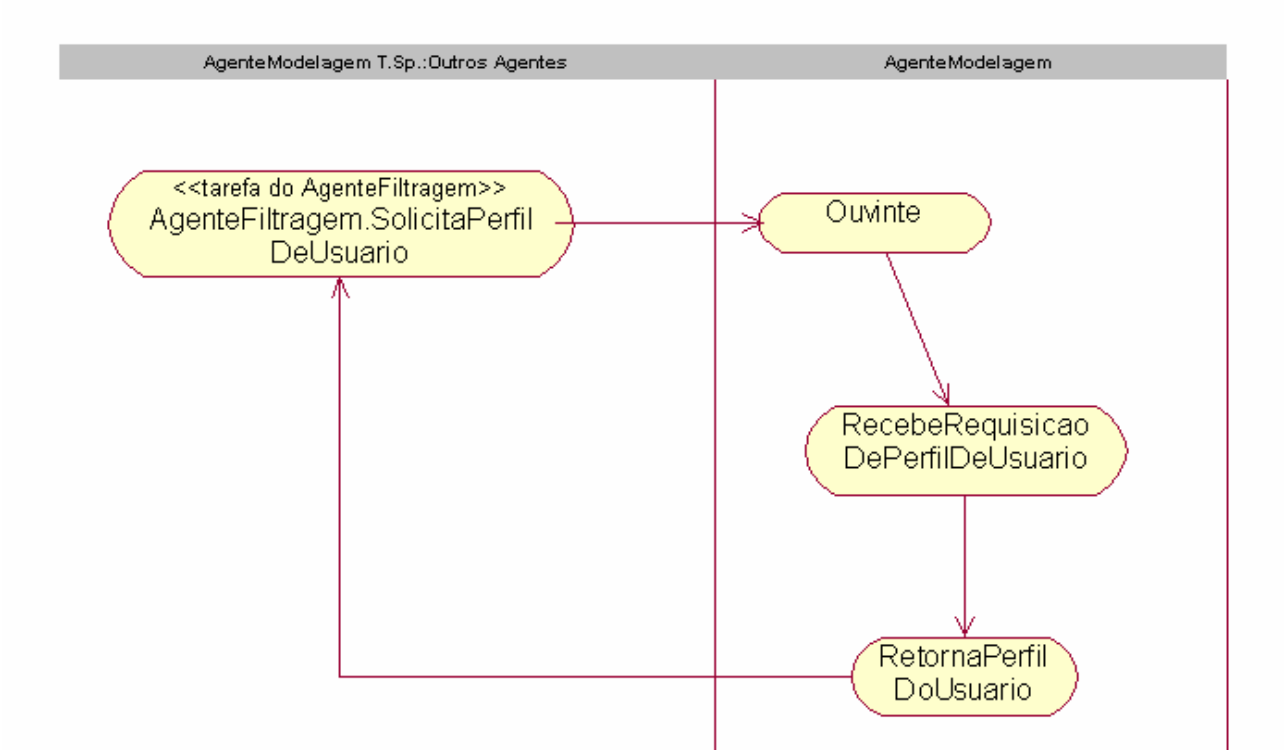

Figura 30 - Agente de Modelagem.

7.5.5 Implementação dos Agentes

Após a fase de análise, definição das funcionalidades, identificação dos agentes, identificação das responsabilidades e detalhamento das tarefas, procedeu-se a implementação do protótipo do MAFIS.

Como linguagem de programação utilizou-se o Java [26], por ter, dentre outras características a virtude de ser completamente orientada a objetos e multiplataforma, proporcionando capacidade de programação distribuída para ambientes heterogêneos e o protocolo FIPA para sistemas baseados em agentes.

O ambiente para codificação foi a IDE (*Integrated Development Environment*) Eclipse [16], um editor para programas Java que aceita o JADE como um *plugin*. O Eclipse é uma IDE gratuita e de código aberto, criada pela IBM em 2001.

Após a escolha e configuração das ferramentas, a linguagem de comunicação entre os agentes foi definido, que é feita através de troca de

mensagens, no caso a ACLMessage (*Agent Communication Language*) da plataforma JADE de acordo com o padrão FIPA. Como linguagem de comunicação inter-agentes, escolheu-se o LEAP que transmite as mensagens através de um conjunto de bytes.

O protocolo de interação, ou seja, o conjunto de regras que devem ser estabelecidos para a comunicação e obedecidas pelo emissor e pelo receptor, foi definido de acordo com os comportamentos de cada agente.

Encontram-se implementados os agentes *Agente de Filtragem, Agente de Busca e Agente de Avaliação*. Nas subseções seguintes, os agentes terão suas implementações detalhadas.

7.5.6 Implementação do Agente de Filtragem

A criação do Agente de Filtragem, Figura 31, é feita a partir da extensão da classe *Agent* da plataforma JADE.

```
public class FilteringAgent extends Agent{
    /*manipulador de conteudo*/
    private ContentManager manager = (ContentManager)getContentManager();
    /*linguagem do agente*/
    private Codec codec = new LEAPCodec();
    /*ontologia do agente*/
    private Ontology ontology = MafisOntology.getInstance();
    [...]
```

Figura 31 - Criação do Agente de Filtragem

Dentre os atributos deste agente, pode-se destacar: manager, que é o manipulador de conteúdo das mensagens recebidas; codec, que recebe uma instância do LEAPCodec, usado para codificar as mensagens; e ontology usado para representar uma instância da ontologia definida para a comunicação inter e intra do Gerenciador de Informações.

```
protected void setup(){
    try{
        manager.registerLanguage(codec);
        manager.registerOntology(ontology);

        /*adiciona o comportamento que vai atender às solicitações*/
        addBehaviour(new FilterBehaviour(this));

        /*adiciona o comportamento que busca e e mantém dados em memória*/
        addBehaviour(new MemoryRefreshBehaviour(this,60000));//executado a cada minuto
    }
    catch(Exception e){
        System.out.prinln("Setup");
        e.getMessage();
```

Figura 32 – Método Setup do Agente de Filtragem

A Figura 32 é usada para ilustrar o método *setup()* do *Agente de Filtragem*. O método *setup()* é utilizado para conter o código que inicializará o agente.

Este agente possui três comportamentos, conforme se ilustra na Figura 33, o *FilterBehaviour* e o *MemoryRefreshBehaviour*.

```
class FilterBehaviour extends CyclicBehaviour{

class MemoryRefreshBehaviour extends TickerBehaviour{
```

Figura 33 – Comportamentos do Agente de Filtragem.

O SearchBehaviour é o comportamento principal do Agente de Filtragem. É construído estendendos-se a classe CyclicBehaviour que pertence ao pacote jade.core.Behaviours. Este comportamento é cíclico, ou seja, é iniciado assim que o agente é iniciado e fica sempre ativo, funcionando como uma espécie de *listener* para o mundo exterior.

O segundo comportamento é o MemoryRefreshBehaviour . Com o intuito de melhorar o desempenho do serviço foi adicionado ao Agente de Filtragem um comportamento responsável por manter informações em memória.

A partir deste comportamento, podem-se manter em memória de acesso rápido, informações que são ao mesmo tempo pequenas e muito acessadas.

```
class MemoryRefreshBehaviour extends TickerBehaviour{
    public MemoryRefreshBehaviour (Agent a, long b){
        super (a,b);
    }

    protected void onTick (){
        [...]
```

Figura 34 – Codificação Comportamento *MemoryRefreshBehaviour*.

A Figura 34 mostra o comportamento MemoryRefreshBehaviour. As tarefas de ArmazenaDadoMemoria e SincronizaDadoMemoria são executadas por este comportamento.

7.5.7 Implementação do Agente de Busca

A criação do Agente de Busca, Figura 35, é feita a partir da extensão da classe *Agent* da plataforma JADE.

```
public class SearchAgent extends Agent{
    /*manipulador de conteudo*/
    private ContentManager manager = (ContentManager)getContentManager();
    /*linguagem do agente*/
    private Codec codec = new LEAPCodec();
    /*ontologia do agente*/
    private Ontology ontology = MafisOntology.getInstance();
    [...]
```

Figura 35 - Criação do Agente de Busca a partir da Plataforma JADE.

7.6 Protótipo do Sistema

No desenvolvimento do sistema, os agentes de software propostos na modelagem do sistema com a finalidade de filtrar informações foram implementados. Na Figura 36, é apresentada uma visão macro do protótipo.

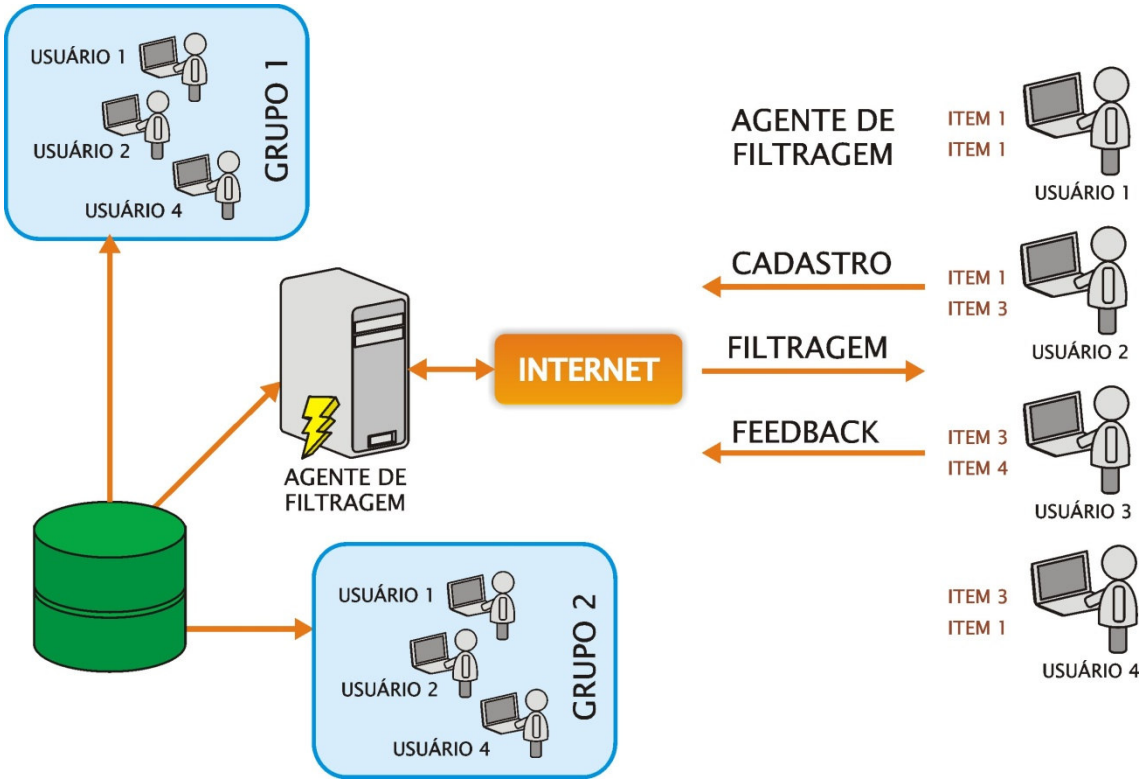

Figura 36 - Visão Macro do Protótipo

Nesta visão macro, com uma versão da Filtragem Colaborativa, são mostrado os usuários, à direita, com suas respectivas preferências (Item 1, Item 2, etc), por conta da avaliação feita na primeira filtragem de itens através da Filtragem Baseada em Conteúdo.

Os usuários são então agrupados segundo suas preferências e assim pode-se classificar um novo usuário em um desses grupos para fazer uma Filtragem Colaborativa.

7.7 Experimentos e Resultados

Para realização dos experimentos do protótipo, um teste foi feito com 4 (quatro) turmas do curso técnico do Centro Federal de Educação Tecnológica do Maranhão (CEFET-MA), com um total de 90 alunos, nas disciplinas de Informática Aplicada ao Saneamento Ambiental, Desenho Assistido por Computador aplicado a Estradas e Sistemas de Informação Geográfica do curso de Gestão Ambiental e Saneamento Ambiental no 1º semestre de 2007 no período de um mês, paralelo às aulas, todas elas turmas

de ensino médio-técnico. Além disso, fizemos um teste inicial com a turma do curso superior do 7 º período de Licenciatura em Informática do CEFET-MA.

Os documentos utilizados no experimento estavam em formato doc para facilitar a indexação e representação dos documentos e a fim de facilitar os testes foram armazenados na base de dados do NetClass.

Foi selecionada manualmente, uma coleção de documentos que tinham relevância a cada um em relação ao domínio especifico de cada área de conhecimento e outros com relação direta com os assuntos abordados.

Para a coleção de documentos submetidos ao agente de indexação, o sistema gerou uma tabela contendo os termos que melhor representam o conteúdo da coleção.

O professor criou o grupo de alunos de acordo com as disciplinas, os alunos fizeram o cadastro no sistema e o professor cadastrou o conteúdo para gerar o perfil baseado no conteúdo inicial dos alunos. Como exemplo cita-se o conteúdo da disciplina Sistemas de Informação Geográfica (SIG): a) Introdução ao Geoprocessamento , b)Noções de Cartografia, c) Sensoriamento Remoto, etc.

A partir daí o sistema de busca do NetClass dará início a uma busca de itens, procurando inicialmente o primeiro tópico abordado na disciplina criando um índice de conteúdo.

Logo após as primeiras interações dos alunos no sistema, é iniciada a filtragem. Como ainda não existe uma avaliação destes itens por nenhum dos usuários do sistema, estes serão comparados ao perfil do aluno através das técnicas de FBC e enviados indicações ao usuários através de uma interface específica, Figura 37.

Em seguida há uma solicitação de uma avaliação da filtragem. O aluno ao receber os itens indicados deverá dar uma nota de 0 a 5 avaliando o seu grau de interesse pelo item filtrado. Dessa maneira poderá ser, tanto atualizado seu perfil, como também ser usado técnicas de FC para filtrar itens de maneira mais efetiva a partir da similaridade entre outros alunos. Dessa maneira aperfeiçoa-se a filtragem e utiliza-se uma abordagem híbrida.

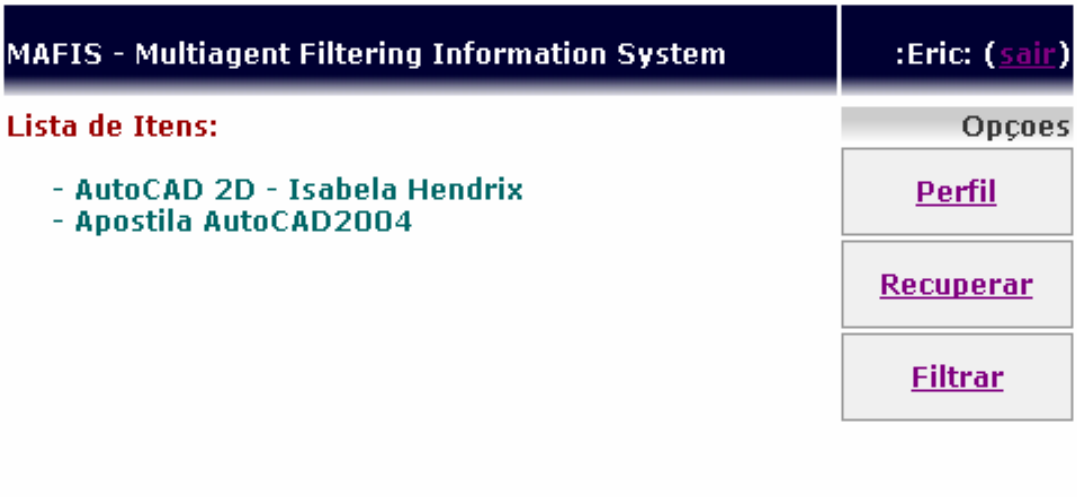

Figura 37 – Interface do MAFIS - Itens Filtrados.

Com estes procedimentos consegue-se resolver as limitações de ambas as técnicas no que diz respeito a superespecialização (overspecialization), analise de conteúdo limitado ao textual, avaliação da qualidade do texto para a filtragem baseado no conteúdo e filtragem de novos itens, pois com FBC todo item pode ser filtrado, número de usuários insuficiente (esparsidade) e usuário "ovelha negra.

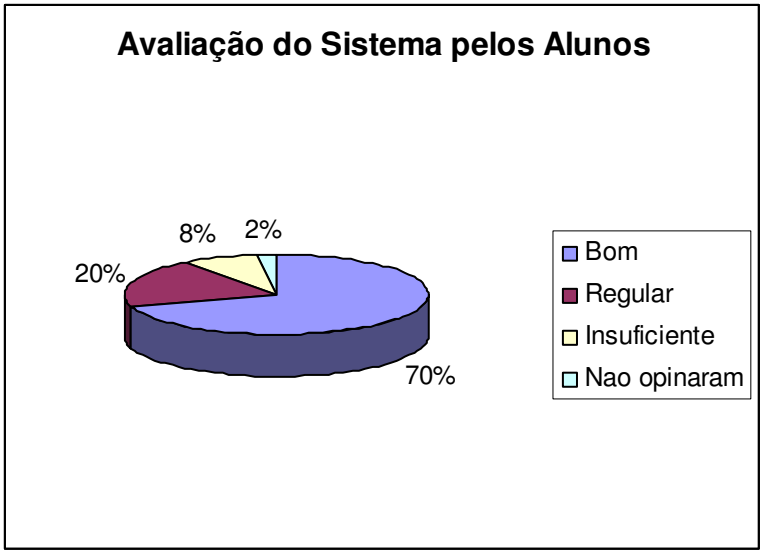

Figura 38 – Avaliação do MAFIS pelos alunos.

Nos testes realizados, de maneira geral os alunos consideraram o sistema muito bom para ser utilizado de maneira auxiliar num Ambiente de Ensino-Aprendizagem haja vista que os documentos filtrados complementam o conteúdo ensinado pelo professor naquela disciplina específica. Foi aplicado

um questionário para medir a satisfaçam dos alunos e destes 70% avaliaram como boa a inserção da abordagem de filtragem, como mostra na Figura 38.

7.8 Considerações finais

Este capítulo se deteve a descrever a implementação do MAFIS, onde inicialmente foi mostrado as tecnologias utilizados para a prototipação do sistema. Mostrou-se ainda os principais diagramas elaborados com a metodologia PASSI.

Em seguida mostra-se a implementação do agente de filtragem mostrando o arcabouço de inicialização e comportamento do agente.

Por fim, a visão macro do protótipo é mostrada e relatado os experimentos realizados.

O próximo capítulo trata das considerações finais com as contribuições do trabalho, com os resultados alcançados e trabalhos futuros.

8 CONCLUSÕES E TRABALHOS FUTUROS

Encontrar informação relevante de maneira fácil e rápida não é uma tarefa fácil, e para usuários de um ambiente colaborativo de ensino-aprendizagem, informação é essencial para a construção do conhecimento.

Nessa perspectiva, a aplicação de um sistema, que possa disponibilizar informação relevante utilizando abordagens distintas e inovadoras, num Ambiente Colaborativo de Ensino-Aprendizagem, levando-se em consideração ainda o tempo gasto na procura por informação relevante e que possam contribuir com a aprendizagem do aluno numa disciplina específica, se faz extremamente necessária.

No presente trabalho, ao se propor o MAFIS, tentou-se tirar proveito das características tanto dos agentes, utilizando uma metodologia voltada para construção dos mesmos, como das técnicas de filtragem de informação, revertendo-as em favor das necessidades de informação dos usuários/alunos do NetClass.

8.1 Contribuições do Trabalho

A principal contribuição deste trabalho para a área de Ambientes de Ensino Colaborativo é a inserção/integração de técnicas de filtragem de informação, especificamente a filtragem híbrida, para melhorar a aprendizagem através da indicação de materiais complementares aos que o professor/tutor possa sugerir, ampliando assim as possibilidades de desenvolvimento dos aprendizes, considerando as suas características individuais e de grupos.

Outra contribuição deste trabalho decorre na tentativa de solucionar as limitações das técnicas de filtragem utilizando uma técnica hibrida derivada das mesmas, onde através da criação do perfil inicial do usuário se possa realizar a filtragem, sem que haja avaliação de itens por parte do aluno.

De modo sucinto, há uma diversidade de trabalhos propondo técnicas de filtragem híbridas, mas nenhum deles contempla os requisitos de: i) iniciar a filtragem sem que seja necessária a avaliação de algum item pelo

usuário para a geração de um perfil; e, ii) ser aplicável ao domínio de Ambientes de ensino colaborativo.

8.2 Resultados alcançados

O presente trabalho, além de ter permitido, na medida do possível, fazer uma compilação do estado da arte do temas (Capítulos 3, 4 e 5), alcançou resultados relacionados com sua proposta nos seguintes itens:

- Revisão, atualização do Agente de Busca, originalmente proposto por Oliveira [49];
- Utilização de uma abordagem híbrida de filtragem de informação com algoritmos clássicos de filtragem;
- Definição de uma sociedade de agentes pra o sistema;
- Criação do modelo do Sistema Hibrido de Filtragem na concepção de Agentes de Software;
- Aplicação do sistema no Ambiente Colaborativo de Ensino-Aprendizagem, o NetClass;

É importante ressaltar que com esse trabalho não se buscou a definição de novos métodos para as tarefas associadas à Filtragem de Informação, mas sim a modelagem de um sistema que fosse capaz de atender às necessidades de informação dos usuários do Ambiente NetClass.

8.3 Trabalhos Futuros

Uma das limitações deste trabalho diz respeito à filtragem inicial se dá apenas com itens textuais.

Em estudos futuros, pretende-se utilizar outros algoritmos e técnicas para verificar a eficiência dos mesmos na filtragem de informação. Nessa ocasião, será possível fazer uma avaliação precisa sobre a eficiência da filtragem de informação realizada pelo sistema multiagentes criado.

Como proposta de trabalhos futuros e para aperfeiçoamento deste trabalho, apresenta-se a lista abaixo:

- Melhorar a performance dos agentes de Filtragem, Avaliação, Modelagem, Interface, Recuperação e Busca;
- Integrar todo os agentes ao Ambiente NetClass de forma definitiva;
- Utilização de outras técnicas para a modelagem do usuário;
- Implementar outros algoritmos de filtragem e realizar testes para comparar o desempenho entre os mesmos;
- Utilizar outras plataformas de construção de agentes, e implementar o mesmo protótipo do modelo desenvolvido neste trabalho, fazendo uma comparação de desempenho entre eles;
- Realizar mais testes e refinar o modelo.

9 REFERÊNCIAS

[1] ADOMAVICIUS, Gediminas; TUZHILIN Alexander. **Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions.** IEEE Transactions on knowledge an Data Engineering, vol 17, n. 6, June 2005.

[2] Amazon Book Store. http://www.amazon.com, acessado em 11 de Outubro de 2005.

[3] BALABANOVIC, M. and SHOHAM, Y. **"Fab: Content-Based, Collaborative Recommendation," Comm**. ACM, vol. 40, no. 3, pp. 66-72, 1997.

[4] BELKIN, Nicholas J, CROFT, W. Bruce. **Information Filtering and Information Retrieval**: Two sides of the same coin. Comm. ACM. vol. 35, nº 12, 1992.

[5] BEZERRA, Byron Leite Dantas. **Estudo de Algoritmos de Filtragem Baseados em Conteúdo**. Monografia de Graduação. UFPE. Recife, 2002.

[6] BREESE, J.S. HECKERMAN, D. and KADIE, C. **Empirical Analysis of Predictive Algorithms for Collaborative Filtering.** Proc. 14th Conf. Uncertainty in Artificial Intelligence, July 1998.

[7] BUCKLEY, C. **Optimization of relevance feedback weights**. In: Proceeding of 1995 ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 351 - 357, Seattle, Whashignton, USA.

[8] BURKE, R. **Knowledge-Based Recommender Systems**. Encyclopedia of Library and Information Systems, A. Kent, ed., vol. 69, Supplement 32, Marcel Dekker, 2000.

[9] BURKE, R. **Hybrid Recommender Systems: Survey and experiments.** User Modeling and User-adapted Interactions. S.1, vol. 12 nº 4, p331-370, Nov-2002.

[10] CAZELLA, Silvio Cesar. REATEGUI, Eliseo Berni. **Sistemas de Filtragem**. Dissertação de Mestrado. UFRGS. 2004.

[11] CLAYPOOL, M., GOKHALE, A., MIRANDA, T., MURNIKOV, P., NETES, D., & SARTIN, M. (1999). **Combining content-based and collaborative filters in an online newspaper.** ACM SIGIR ’99 Workshop on Recommender Systems, Berkely, CA, August, 1999.

[12] COSSENTINO, M., and POTTS, C. **A case tool supported methodology for the design of multi-agent systems**. Las Vegas (NV), USA: The 2002 International Conference on Software Engineering Research and Practice. 2002.

[13] COVER, T. M., and HART, P. E. (1967). **Nearest Neighbor Classifiers**. IEEE Transactions on Computers, 23-11, November, 1974, pp. 1179-1184.

[14] DIAS, C.R., SOARES, S.S.F., OCHI, L.S. **Problemas de Clusterização em Mineração de Dados**. Status: Publicado integralmente (46 páginas) nos Anais do Encontro Regional de Informática (ERI)(em CD-ROM), regional RJ/ES, decorrente de Mini Curso a ser proferida em Vitória e Rio das Ostras

[15] DOVAL, D., MANCORIDIS, S. e MITCHELL, B. S. **Automatic Clustering of Software Systems using a Genetic Algorithm**. In 1999 International Conference on Software Tools and Engineering Practice (STEP '99), 1999.

[16] ECLIPSE. Eclipse – an open development plataform. Disponível em: http://www.eclipse.org/. Acesso em 22 mar. 2007.

[17] ELLIS, S.; WHALEN, S. F. **Cooperative Learning**: getting started. Scholastic, New York, 1990.

[18] ESMA, A.; FRASSON C.; LIBERT G. **Towards new learning strategies in intelligent tutoring systems**. SBIA-95. Campinas: Springer-Verlag Edition. October 1995.

[19] FASULO, D. An **Analysis of Recent Work on Clustering Algorithms**. Technical Report, Dept. of Computer Science and Engineering, Univ. of WASHINGTON, 1999.

[20] FERREIRA, J. S. **Concepção de um ambiente multi-agentes de ensino inteligente integrando o paradigma de aprendizagem cooperativa.** Dissertação de Mestrado. Programa de Pós-Graduação em Engenharia da Eletricidade - Universidade Federal do Maranhão, São Luís. 1998.

[21] FIPA. **The Foundation for Intelligent Physical Agents**. Disponível em <http://www.fipa.org/>. Acesso em 22 ago. 2006.

[22] GEY, F. **Models in Information Retrieval**. Folders of tutorial presented in 19th ACM Conference on Research and Development in Information Retrieval (SIGIR). 1992.

[23] HERLOCKER, J. **Understanding and Improving Automated Collaborative Filtering Systems.** PhD. Thesis, University of Minnesota. Disponível em http://web.engr.oregonstate.edu/~herlock/, acessado em 01 novembro 2005.

[24] HUANG, Z. **Extensions to the K-means algorithm for Clustering large data sets with categorical values**. Data Mining and Knowledge Discovery, Kluwer Academic Publishers, Hingham, MA, USA, Vol. 2, Issue 3. 1998.

[25] JADE. **Java Agente Development Framework**. Disponível em <http://jade.tilab.com/>. Acesso em 22 ago. 2006.

[26] JAVA. The Source for Java Developers. Disponível em: <http://java.sun.com/index.jsp>.Acesso em 02 mar. 2007.

[27] JOHNSON, D. W.; JOHNSON, R. T. **Learning together and alone, cooperative, competitive and individualistic learning.** Allyn and Bacon, Paramount. 1994.

[28] KAZUHIRO, Iwahama, YOSHINORI, Hijikata, SHOGO, Nishida. **Content-based Filtering System for Music Data**. saint-w, p. 480, 2004 Symposium on Applications and the Internet-Workshops (SAINT 2004 Workshops), 2004.

[29] KONSTAN, J.A., MILLER, B.N., MALTZ, D., HERLOCKER, J.L., GORDON, L.R. and RIEDEL, J. **GroupLens: Applying Collaborative Filtering to Usenet News**. Communications of the ACM, 40 (3). (1997). 77 - 87.

[30] LABIDI, S.; SILVA, J.; COUTINHO, L.; COSTA, E. **Agent-Based Architecture for Cooperative Learning Environment**. Anais do XI Simpósio Brasileiro de Informática na Educação (SBIE'2000). Maceió-AL: 18 a 20 de Novembro, 2000.

[31] LABIDI, S.; FERREIRA, J. S. **Technology-assisted instruction applied to cooperative learning: the SHIECC project.** In: Proceedings of the

IEEE International Conference Frontiers in Education (FIE'98). Tempe, Arizona, november 4-7, 1998.

[32] LABIDI, S.; SOUZA, C. M., NASCIMENTO, E. **NetClass: Cooperative Learner Modeling in a Web Based Environment.** In Proceedings of the 6th Int. Conf. on Computer Based Learning Science (CBLIS) Nicosia, Cyprus: University of Cyprus. 2003.

[33] LABIDI, S., COSTA, N., FERREIRA, J. **Modeling of an Authoring Tool for an Inteligent tutoring System**. In the Proceedings of the 6th Int. Conf. on Computer Based Learning in Science (CBLIS), 2003, Nicosia. University of Cyprus, 2003.

[34] LI, Peng; YAMADA, Seiji. **A Movie Recommender System Based on Inductive Learning**. In: Conference on Cybernetics and Intelligent Systems – Proceedings of the IEEE, Singapore, 1-3 December 2004.

[35] LIMA, Adilson da Silva. **UML 2.0 – Do Requisito à Solução**. 1 edição. Editora Erica. São Paulo. 2005.

[36] LORENZI, Fabiana. SANTOS, Daniela Scherer. BAZZAN, Ana L. C.. **Case-Based Recommender Systems inspired by social insects**. Universidade Luterana do Brasil.

[37] LUCENA, C. J. P.; FUKS, H.; MILIDIU, R.; LAUFER, C.; BLOIS, M.; CHOREN, R., TORRES, V.; FERRAZ, F.; ROBICHEZ, G.; DAFLON, L. **AulaNet: Ajudando professores a fazerem o seu dever de casa.** In: Anais do XXVI Seminário Integrado de Software e Hardware, págs. 105-117. Rio de Janeiro, RJ. 1999.

[38] MARINHO, Leandro Balby. **Um Framework Multiangente para a personalização da web baseado na modelagem de usuários e na mineração de uso**. Dissertação de Mestrado. UFMA. 2004.

[39] McNEE, M. S., et al. **On the recommending of citations for research papers**. Communications of the ACM, New Orleans, Nov. 2002.

[40] MIDDLETON, Stuart E.; ROURE, David C. de.; SHADBLOT, R. **Capturing knowledge of User Preferences: Ontologies in Recommenders Systems**. Disponível em:<http://eprints.ecs.soton.ac.uk/archive >. Publicado em: 2001. Acesso em: 08 out. 2005.

[41] MIDDLETON, S.; SHADBOLT, N.; and ROURE, D. D. 2004. **Ontological user profiling in recommender systems**. In ACM Transactions on Information Systems, volume 22(1), 54– 88.

[42] MIDDLETON, Stuart E.; ALANI, Harith.; ROURE, David C. de. **Exploiting Synergy Ontologies and Recommender Systems**. Disponível em: <http://semanticweb2002.aifb.uni-karlsruhe.de/proceedings/ > Publicado em: 2002. Acesso em: 08 out. 2005.

[43] MILLER, Bradley N. KONSTAN, Joseph A., RIEDL, John. PocketLens: **Toward a Personal Recommender Systems**. University of Minnesota. ACM Transactions of Informations Systems, Vol. 22, nº 3, pages 437-476. July 2004.

[44] MIN, Sung-Hwan. HAN, Ingoo. **Detection of the customer time-variant pattern for improving recommender systems**. Expert systems with applications 28. Elsevier. 2005.

[45] MITCHELL, Tom M. **Machine Learning.** New York, United States of America: McGraw-Hill, 1997.

[46] MOTTA, Claudia Lage Rabello da. **Um ambiente de Filtração e Filtragem Cooperativas para apoio a equipes de trabalho**. Tese (Doutorado em Ciências em Engenharia de Sistemas e Computação). Universidade Federal do Rio de Janeiro (COPPE/UFRJ). Rio de Janeiro.1999.

[47] OLINDA, N. P. C.. **Recuperação de Informação**. Científico, Ano IV, v. I, Salvador, janeiro-junho 2004.

[48] OLIVEIRA, Calos Augusto F. de. REIS, Josiene Fabíola T. **Filtragem Colaborativa - uma forma de personalizar informações**. Científico, Ano IV, v. I, Salvador, janeiro-junho 2004.

[49] OLIVEIRA, R., SERRA JR., G., LABIDI, S., RABELO, W. Recuperação de Informações Web para um Ambiente de Aprendizagem Computadorizada: uma abordagem multiagente. Dissertação de Mestrado. UFMA. 2005.

[50] OMG, Green Paper. **Agent Technology**. Agent Working Group OMG Document c/2000-04-01,2000.

[51] PASCAL, L.; MARTIAL, V.; PATRICK, B. **Cooperation between humans and a pedagogical assistant in a learning environment**. In:

International Conference on the Design of Collaborative Knowledge-Based Systems (COOP'96). Juan-les-Pins, France, 1996.

[52] PALIOURAS, G; KARKALETSIS, V; PAPATHEODOROU, C.; SPYROPOULOS, C. **Exploiting Learning Techniques for the Acquisition of User Stereotypes and Communities.** Proceedings of the 7th International Conference on User Modeling, Canada, June, 1999.

[53] PAPATHEODOROU C. **Machine Learning in User Modeling.** Machine Learning and Applications. Lecture Notes in Artificial Intelligence. Springer Verlag, 2001.

[54] PASSI. **A Process for Agent Societies Specification and Implementation**. Disponivel em: http//Mozart.csai.unipa.it/passi/. Acesso em 18 ago. 2006.

[55] PASSOS, E., GOLDSCHIMIDT, R. **Data Mining: Um Guia Prático**. Editora Campus e Elsevier. 2005.

[56] QUINLAN, R. **Induction of decision trees.** Machine Learning, vol. 1, pp. 81 – 106, 1986.

[57] QUINLAN, **R. C4.5: Programs for Machine Learning.** Morgan Kaufmann, Sao Mateo, California, 1993.

[58] RESNICK, P. LACOUVOU, N. SUSHACK, M. BERGSTROM, P. RIEDL, J. **GroupLens: An Open Arquiteture for Collaborative Filtering of NetNews**. In Proc. CSCW 94, Chapel Hill, 176-186,1994.

[59] RIJSBERGEN, C. J. **Information Retrieval**. Butterworths, London, 1979

[60] ROCHA, Catarina Carneiro. **RECDOC: Um Sistema de Filtração para uma biblioteca digital na web**. Dissertação de Mestrado. UFRJ.2003.

[61] RUSSELL, S. J.; NORVING P. **Artificial Intelligence: A Modern Approach**. Prentice Hall, 1995.

[62] SALTON, G. **Automatic Text Processing: The Transformation, Analisys, and Retrieval of Information by Computer**, Addison Wesley, 1989.

[63] SALTON, G. and McGILL, M. J. **Introduction to Modern Information Retrieval**. McGraw-Hill, New York, 1983

[64] SARWAR, B. M.; KARYPIS, G.; KONSTAN, J. A. and RIEDL, J. 2001. **Item-based collaborative filtering recommendation algorithms**. In

Proceedings of the 10th International World Wide Web Conference - WWW10, Hong Kong.

[65] SERRA JR., G.; COUTINHO, L.; LABIDI, S. **Formation of Groups for Cooperative Learning: a Genetic Algorithm Approach**. In Proceedings of Conference on Computers and Advanced Technology in Education (CATE 2001). Banff, Canada: June 27-29, 2001.

[66] SHARDANAND, Upendra. MAES, Pattie. **Social Information Filtering: algorithms for Automating "Word of Mouth"**. MIT Media Labs.

[67] TRAJKOVA, Joana; GAUCH, Susan. **Improving Ontology-Based User Profiles**. Disponível em: <http://www.ittc.ku.edu/keyconcept/publications>. Acesso em: 20 out. 2005.

[68] TORRES, Roberto. **Personalização na Internet**. Novatec Editora. 2004.

[69] TORRRES JUNIOR, Roberto Dias. **Combining Collaborative and Content-based Filtering to recommend research papers**. Dissertação Mestrado. UFGRS. 2004.

[70] VAN RIJSBERGEN, C. J. **Information Retrieval**, Butterworths, 2nd edition, 1979.

[71] WEBB, G.; PAZZANI, M.; BILLSUS, D. **Machine Learning for user modeling.** User Modeling and User-Adapted Interaction, vol. 11, pp.19-29, 2001.

[72] WEBB, G.; PAZZANI, M.; BILLSUS, D. **Machine Learning for user modeling**. User Modeling and User-Adapted Interaction, vol. 11, pp.19-29, 2001.

[73] WITTEN, Ian H.; FRANK, E.. **Data Mining – Practical Machine Learning Tools and Techniques with Java Implementations.** San Francisco, CA, United States of America: Morgan Kaufmann Publishers, 2000.

[74] WOOLDRIDGE et al., 1995) WOOLDRIDGE, Michael; NICHOLAS R. Jennings (1995). **Agent Theories, Architectures, and Languages: a Survey**. In Wooldridge and Jennings Eds., Intelligent Agents, Berlin: Springer-Verlag, 1995.

[75] CADE. Disponível em http://br.cade.yahoo.com. Acessado em: 20 out. 2005.

[76] GOOGLE. Disponível em http://www.google.com.br. Acessado em: 20 out. 2005.

[77] ALTAVISTA. Disponível em http://www.altavista.com. Acessado em: 20 out. 2005.