Lucimar Candido Martins

PERSONALIZAÇÃO DE VISÕES SOBRE DOCUMENTOS HIPERMÍDIA

DISSERTAÇÃO DE MESTRADO

Departamento de Informática

Pontifícia Universidade Católica do Rio de Janeiro

Rio de Janeiro, 10 de junho de 2002

Lucimar Candido Martins

PERSONALIZAÇÃO DE VISÕES SOBRE DOCUMENTOS HIPERMÍDIA

Dissertação apresentada ao Departamento de Informática da PUC-Rio como parte dos requisitos para obtenção do título de Mestre em Informática

Orientador: Prof. Marco Antonio Casanova

Co-orientador: Prof. Carlos J. P. de Lucena

Departamento de Informática

Pontifícia Universidade Católica do Rio de Janeiro

Este trabalho é dedicado a Deus e aos meus queridos pais, Maria de Lourdes e José Solimar, por tudo o que eles representam em minha vida.

Agradecimentos

A Deus, por estar sempre orientando meus passos, encorajando-me a enfrentar todos os desafios e chegar até aqui.

Aos meus pais, Lourdes e Solimar, a minha irmã Rosemar e aos meus tios Lia e Ary, pela compreensão, amor e carinho de sempre.

Ao meu orientador, professor Marco Antonio Casanova, por ter acreditado em mim, pelo incentivo, pelos ensinamentos e críticas, e pelo apoio dado durante todo o caminho.

Ao meu co-orientador, professor Carlos J. P. de Lucena, pela confiança, pelo incentivo, pela oportunidade de integração no grupo de pesquisa do TecComm e pelo apoio financeiro durante o curso.

Aos professores, Luiz Fernando Gomes Soares e Simone D. J. Barbosa, pela atenção, pelas idéias, revisões e pelo apoio na definição deste trabalho

As minhas irmãs de coração, Luciana, Paulinha e Viviane, pela amizade, pelo carinho, incentivo, companheirismo, compreensão e apoio em todos os momentos. Obrigada por tudo!

A minha prima Andréa, pelo incentivo e apoio, fundamentais para o início desta caminhada.

Ao Gésner, pelo apoio irrestrito, por toda a compreensão, cumplicidade e amor de sempre.

Aos colegas do laboratório TecComm, pela amizade e ajuda, em especial ao João, Guga, Daflon, Matheus e Viviane, pelos ensinamentos e incentivo que muito colaboraram para a realização deste trabalho.

A Tatiana Coelho e ao Davi Romero, pelas idéias, pelas discussões e ajuda na realização deste trabalho.

Aos amigos da Puc, Fabio Marcos, Adelailson, Alésio, Elton, Alessandro, Meire Juliana, colegas do laboratório TecGraf, pela ajuda, pela alegria e companheirismo.

A todos os professores e funcionários do Departamento de Informática da Puc-Rio pelos ensinamentos adquiridos e pela ajuda prestada durante o mestrado.

Aos professores, Lucília e Pietrobon, ao Pedro Filho, pela ajuda e pelo incentivo recebido para dar início a esta caminhada.

Ao Departamento de Informática da UFOP pelo esforço em dar uma formação adequada aos seus alunos, em especial aos professores, por sua dedicação, estímulo e ensinamentos. Não poderia deixar de agradecer aos amigos conquistados durante o curso, pela troca de conhecimento e por todos os momentos que passamos juntos.

A CAPES pelo apoio financeiro recebido durante o curso.

Resumo

Frente à infinidade de dados disponíveis na Web, levar o dado certo a um usuário particular é uma das tarefas mais difíceis. Muitas aplicações relacionadas a e-learning, comércio eletrônico e recuperação de informação vêm sendo desenvolvidas com este propósito. Máquinas de busca, sistemas de recomendação e sistemas de personalização podem ser apontados como exemplos.

Apesar do grande esforço na tentativa de desenvolver sistemas de apresentação e seleção de conteúdo personalizado, ainda existem diversos desafios. Parte deles estão diretamente relacionados à filtragem, empacotamento e formatação das informações quando adaptadas aos objetivos, interesses e preferências de apresentação dos usuários.

Neste contexto, o principal objetivo da dissertação é a definição de um *framework* que contemple as questões relacionadas a personalização, tratando separadamente os processos de filtragem, empacotamento e formatação, utilizando os conceitos de *separation of concerns*. Desta forma, o *framework* facilita a implementação e o refinamento dos algoritmos que endereçam estas questões.

Abstract

One of the most difficult tasks on dealing with the large amount of available data on the Web is to deliver the right data to a specific user. Many applications related to e-learning, e-commerce and information retrieval have been developed for this purpose. Some examples are search engines, recommendation systems and personalization systems.

In spite of the great effort to develop personalized presentation and filtering systems there are challenging problems to be resolved. Some of them are directly related to information filtering, packaging and formatting, wherever this information needs to meet the user goals, interests and preferences.

The main purpose of this dissertation is to define a framework, using the *separation of concerns* concepts, which contemplates personalization issues such as filtering, packaging and formatting processes. The framework facilitates the implementation and fine-tuning of algorithms addressing these issues.

Sumário

Lista de Figuras

Lista de Tabelas

1. Introdução

A “desorientação no hiperespaço” é um sentimento familiar a quase todos que navegam na Web. Dispersamo-nos frente ao mundo de informações que nos é apresentado e com freqüência chegamos a ponto de não sabermos onde estamos, o que queremos ou qual nosso objetivo.

Os sistemas adaptativos procuram evitar esses problemas auxiliando o usuário em suas tarefas através do espaço de informação. Vários sistemas têm sido projetados com a finalidade de adaptar a navegação e o conteúdo ao conhecimento do usuário [Brusilovsky99, Bradley00], às suas preferências e metas [Kamba95, kamba95a, Fink97, ArdGoy99, Ardissono99a, Cingil00, Ardissono00], às suas tarefas e receptividade [Khan00, Billsus99, Billsus00, Billsus00a]. Vários destes estudos mostram a efetividade e o ganho que a utilização de tais técnicas podem trazer [Hof98, Billsus99, Parsaye00].

Apesar do grande esforço na tentativa de desenvolver sistemas de apresentação e seleção de conteúdo personalizados, ainda existem diversos desafios como:

- aquisição e representação de informações relevantes relativas ao conteúdo e aos usuários;
- filtragem do conteúdo de forma que as informações sobre os usuários e do próprio conteúdo sejam consideradas;
- empacotamento e formatação do conteúdo a fim de adaptá-las às preferências de apresentação e dispositivos de acesso dos usuários.

Neste contexto, o principal objetivo da dissertação é a definição de um *framework* que facilite a criação de aplicações voltadas para a recuperação de documentos, baseada nos próprios descritores dos documentos. O *framework* é definido em três camadas que tratam os processos de filtragem, empacotamento e formatação, utilizando os conceitos de *separation of concerns* [Tarr99]. A flexibilidade oferecida pelo *framework* facilita o refinamento de diferentes algoritmos e configurações.

A primeira contribuição da dissertação é, portanto, a definição de um *framework* que organiza genericamente todo o processo de personalização (independente do domínio da aplicação), permitindo a reutilização de código e o mínimo de reprojeto. A utilização do *framework* será ilustrada a partir de duas instanciações chamadas *MyNews*, que implementa um jornal eletrônico adaptativo, e *MyRefs*, que implementa um repositório adaptativo de referências bibliográficas.

A segunda contribuição da dissertação é a proposta de desassociação dos processos de filtragem e formatação dos documentos, de forma que a estruturação dos documentos e da aplicação em si seja tratada isoladamente por um estágio intermediário, o empacotamento. O processo de empacotamento, portanto, irá definir a estrutura abstrata da aplicação, delineando a navegação e estruturação do conteúdo, independente das questões de formatação.

A Figura 1 apresenta uma visão geral do processo de adaptação proposto. A etapa de filtragem obtém os documentos supostamente relevantes para o usuário em questão. A etapa de empacotamento, baseada em padrões de apresentação previamente definidos, organiza e estrutura os documentos obtidos na etapa anterior. Finalmente, a etapa de formatação define o layout final da aplicação.

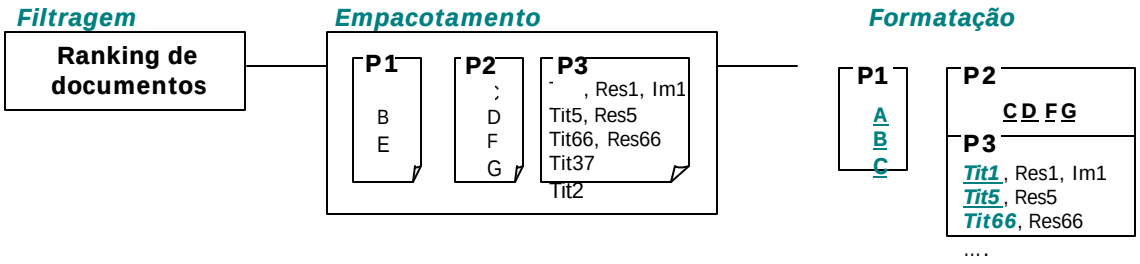


Figura 1: Visão Geral do Processo de Adaptação

É importante ressaltar que outros trabalhos também procuram desassociar os processos de filtragem e formatação [ArdGoy00, Ardissono99a, Billsus02, Carolis99]. Porém, questões relacionadas à estrutura, ordem e agrupamento dos documentos são tratadas juntamente a um dos dois processos ou em ambos.

Os resultados parciais desta dissertação encontram-se publicados em [Martins02].

Esta dissertação está organizada da seguinte forma. O Capítulo 2 discute os conceitos de hipermídia adaptativa e sua aplicação em alguns sistemas adaptativos abordados na

literatura. O Capítulo 3 apresenta uma breve descrição do *framework*, discute o processo de filtragem e empacotamento dos documentos e alguns aspectos de modelagem. O Capítulo 4 discute as instanciações do *framework* para o *MyNews* e o *MyRefs*. Finalmente, o Capítulo 5 apresenta as conclusões e trabalhos futuros.

2. Hipermídia Adaptativa

Este capítulo apresenta uma visão geral sobre as técnicas utilizadas para a personalização de sistemas hipermídia. Ele descreve os dados que devem ser utilizados na personalização do conteúdo, apresentação e estrutura da aplicação, considerando os interesses e necessidades do usuário. Descreve, também, os métodos de aquisição desses dados e os diferentes tipos de adaptação. Finalmente, ele apresenta alguns trabalhos relacionados ao tema da dissertação.

O leitor poderá cobrir apenas a seção 2.4 sem perda de continuidade. Os conceitos abordados nas demais seções não serão utilizados diretamente no restante do texto.

2.1 Introdução

O objetivo de um sistema hipermídia adaptativo é aumentar a funcionalidade da aplicação tornando-a personalizada. Os sistemas adaptativos podem ser usados em muitas situações em que a aplicação é delineada para pessoas com diferentes objetivos e conhecimento, e onde o hiperespaço é razoavelmente grande. Estes sistemas constróem um modelo individual do usuário e o utilizam para adaptar o conteúdo de uma página hipermídia de acordo com o conhecimento e metas do usuário, ou sugerindo elos mais relevantes para seguir.

Durante a última década, foram construídos vários tipos diferentes de sistemas de hipermídia e Web sites que podem executar algum tipo de personalização. Tais sistemas ou aplicações possuem nomes diferentes de acordo com o tipo de personalização que realizam [DeBra99a]:

- Hipermídia adaptável: O usuário pode prover algum perfil através de um diálogo ou questionário. A coleta de dados neste caso acontece explicitamente.

- Hipermídia adaptativa: A extração dos dados é feita implicitamente. Através do monitoramento do comportamento do usuário e dos recursos de plataforma disponíveis, a apresentação é adaptada ao usuário. A maioria das adaptações são

- baseadas na navegação do usuário e possivelmente no comportamento de outros usuários.

A adaptabilidade e a adaptatividade podem coexistir em uma mesma aplicação. A escolha deve ser cuidadosamente avaliada para cada classe de adaptação, levando em conta a conveniência e demanda do usuário, sua irritação e as conseqüências de uma adaptação incorreta. O controle do usuário nesse processo pode ser permitido em um nível geral (onde os usuários podem permitir ou não a adaptação como um todo), em um nível de tipo (onde os usuários podem aprovar ou não certos tipos de adaptações), ou em um nível específico. Por exemplo, um sistema que monitora taxas de transmissão pode (1) sugerir o abandono de um vídeo e oferecer uma imagem; (2) sugerir o abandono de todos os vídeos; ou (3) simplesmente não apresentar os vídeos sem que o usuário perceba que existem vídeos disponíveis [Kobsa01].

A complexidade da adaptação é uma medida relativa da complexidade do processo que produz a adaptação baseada no número de dados de entrada. Se há uma relação direta entre a entrada e a adaptação, então a complexidade é baixa. Isto ocorre, por exemplo, no caso em que um usuário seleciona o modo de “somente texto” em seu browser e consequentemente não são apresentadas as imagens. Um exemplo de uma adaptação complexa é a inferência daquilo que é interessante ao usuário com base na observação da interação do usuário com uma fonte de informação.

O processo de personalização em aplicações hipermídia pode ser dividido em três grandes tarefas que são geralmente realizadas por componentes distintos do sistema [Kobsa01]:

- Tarefa de aquisição: identifica a informação disponível sobre as características dos usuários, comportamento na interação e estado do ambiente. Isto é possível tanto pelo monitoramento do uso do computador como pela obtenção de informações de fontes externas. Esta tarefa é responsável por tornar a informação acessível para o componente de adaptação da aplicação e por construir os modelos de usuário, de uso e/ou de plataforma.

- Tarefa de representação e inferência: coleta os dados referentes aos usuários e/ ou grupos, ao seu comportamento e ambiente. Esta tarefa também é responsável pela construção dos modelos, permitindo acesso e processamento adicionais. Além disso, deve se responsabilizar pela integração das informações advindas de outras fontes.

- Tarefa de produção: realiza a adaptação do conteúdo, da apresentação e da estrutura, baseando-se nos modelos de usuário, uso e plataforma.

2.2 Dados necessários para adaptação

A personalização é freqüentemente uma tarefa de manipulação intensiva de dados. Alguns dados são tipicamente observados pelo sistema diretamente, enquanto outros podem requisitar um ou mais passos adicionais.

A adaptação é baseada nas informações existentes nos modelos suportados pela aplicação em questão. Exemplos de modelos são: modelo de usuário e de plataforma.

A seguir são apresentados os diferentes tipos de dados que um sistema adaptativo deve considerar, chamados de dados do usuário, dados de uso e dados da plataforma.

2.2.1 Dados do usuário

Quando se fala de algum tipo de sistema adaptativo é preciso identificar quais os aspectos do trabalho do usuário com o sistema podem ser considerados para realizar a adaptação e quais as características que diferem de usuário para usuário. Existem muitas características relacionadas ao contexto do trabalho do usuário e ao usuário individualmente. Os sistemas hipermídia adaptativos usam normalmente cinco características [Brusilovsky96]: metas do usuário, conhecimento, background (dados demográficos), experiência (no hiperespaço) e preferências.

Planos e metas dos usuários

Na interação com os sistemas, os usuários freqüentemente irão acessar páginas com uma meta específica em mente. Uma meta típica pode ser encontrar uma informação sobre um tópico específico ou comprar um tipo de produto. Um sistema que ajude o usuário a atingir suas metas pode ser muito útil.

A meta do usuário é uma característica que pode mudar com muita freqüência: quase sempre muda de sessão para sessão e pode mudar várias vezes em uma mesma sessão de trabalho. Em alguns sistemas pode ser necessário distinguir metas locais, que podem mudar com maior freqüência, de metas globais, de mais alto nível, que são mais estáveis. Por exemplo, em sistemas educacionais a meta de aprendizado é considerada de alto nível,

enquanto que a solução de um problema específico é uma meta de baixo nível ou local, pois pode mudar várias vezes em uma mesma sessão.

Conhecimento do usuário

O conhecimento do assunto representado no hiperespaço parece ser a característica mais importante do usuário para os sistemas de hipermídia adaptativos existentes [Brusilovsky99, Bradley00]. Quase todas as técnicas de apresentação adaptativas contam com o conhecimento do usuário como uma fonte de adaptação. O conhecimento do usuário é uma informação variável relacionada a um usuário particular, o que significa que um sistema de hipermídia adaptativo, que confia no conhecimento de usuário, tem de reconhecer as mudanças no seu conhecimento e atualizar o modelo adequadamente.

O conhecimento do usuário no assunto é representado freqüentemente por um modelo de sobreposição que está baseado no modelo estrutural do domínio do assunto. Geralmente este domínio estrutural é representado como uma cadeia de conceitos de domínio.

A idéia do modelo de sobreposição é representar o conhecimento individual de um usuário sobre o assunto como uma "sobreposição" do modelo de domínio. Para cada conceito do modelo de domínio, um modelo de sobreposição individual armazena algum valor, que é uma estimativa do nível de conhecimento do usuário sobre este conceito. Isto pode ser apenas um valor binário (conhecido/não conhecido) como em [Benaki97] que considera um item do domínio como "interessante" ou "não interessante" ao usuário. Pode ser também um medida qualitativa como em [Wu99], que descrevendo a modelagem de usuários do AHAM (Adaptive Hypermedia Application Model), considera um conceito como "well learned", "learned" ou "not known". Por fim, pode ser uma probabilidade indicando o quanto o usuário sabe sobre o conceito.

Algumas vezes, um modelo de usuário estereotipado, mais simples, é usado para representar o conhecimento do usuário. Um estereótipo distingue vários usuários "típicos" ou "estereotipados". Para cada dimensão da modelagem de usuário o sistema pode ter um conjunto de possíveis estereótipos. No sistema Seta, como descrito em [ArdGoy99], o usuário pode ser classificado em um dos 3 estereótipos: novato, intermediário e especialista.

Um modelo de estereótipos é mais simples e menos poderoso que um modelo de sobreposição, mas também é mais geral e muito mais fácil de ser inicializado e mantido.

Dessa forma, bons resultados podem ser alcançados combinando-se estereótipo e sobreposição: o modelo de estereótipo é usado no início para classificar um usuário novo e fixar valores iniciais para um modelo de sobreposição que será usado em seguida, como em [ArdGoy99].

O conhecimento do usuário sobre os conceitos, relações entre conceitos, fatos e regras relativos ao domínio da aplicação são essenciais no ajuste da apresentação do material hipermídia de forma que os usuários não se aborreçam com explanações desnecessárias nem se confundam com detalhes que não podem entender. No sistema ADAPTS [Brusilovsky99], por exemplo, a apresentação das explicações e detalhes técnicos são dependentes da experiência do usuário. Da mesma forma, no sistema SETA [Ardissono99, Ardissono00, ArdGoy99], a geração das descrições dos produtos são dependentes do conhecimento do usuário.

Experiências e habilidades do usuário

Os sistemas mencionados anteriormente se baseiam principalmente no conhecimento do usuário sobre os conceitos e as relações entre os mesmos no domínio de uma aplicação. Porém, além de conhecer o assunto é importante que o usuário esteja familiarizado com a estrutura do hiperespaço e como navegar por ele. Geralmente o usuário que está bastante familiarizado com o próprio assunto não está familiarizado com a estrutura do espaço e vice-versa.

O sistema de informação turística AVANTI [Fink97, Fink99] leva em conta as necessidades de diferentes tipos de pessoas com alguma deficiência física (limitado por cadeira de rodas, com deficiência motora ou de visão). Dessa forma as ações são recomendadas somente quando os usuários são realmente capazes de executá-las.

Dados demográficos

Os dados demográficos de um usuário se referem aos “fatos objetivos” como: dados pessoais (nome, endereço, telefone, etc.), dados geográficos (código de área, cidade, estado e país), características do usuário (idade, sexo, educação, etc.), dados psicográficos (indicando estilo de vida), dados qualificando o comprador (freqüência de uso dos produtos/serviços, etc.) e registro para recebimento de ofertas.

Estes dados podem ser de grande importância quando combinados com dados estatísticos de alta qualidade, isto é, sobre o comportamento de compra de diferentes grupos de consumo.

Interesses e pre ferências do usuário

Os interesses entre os usuários de uma mesma aplicação podem variar consideravelmente. As informações e ofertas de produtos que são direcionados a um grupo de interesse particular podem, além de não interessar a um outro grupo, causar efeitos adversos.

Estas preferências podem ser absolutas ou relativas, ou seja, dependentes do nó acessado, metas e contexto atual. Diferente de outros componentes, as preferências não podem ser deduzidas pelo sistema. O usuário tem de informar ao sistema diretamente ou indiretamente, através de uma realimentação simples. Isto se aproxima mais da idéia de adaptabilidade do que adaptatividade.

Outra característica específica da modelagem de preferência é o modo de representação. Enquanto normalmente outras partes do modelo de usuário são representadas simbolicamente, as preferências são freqüentemente representadas e calculadas numericamente.

Em [Billsus99] é apresentado um agente inteligente projetado para compilar as notícias diárias para um usuário específico. Baseado no feedback do usuário, o sistema automaticamente adapta as notícias às preferências e interesses do usuário. Esse trabalho propôs uma abordagem híbrida para a modelagem dos interesses do usuário, que consiste na separação dos modelos de interesses de “longo prazo” e interesses de “curto prazo”. Isto permitiu lidar com um problema importante: o fato de que as informações sobre o usuário precisam ser atualizadas como um resultado direto da interação com a informação.

2.2.2 Dados de uso

Os dados de uso podem ser diretamente observados e gravados, ou adquiridos analisando os dados observáveis. O que pode ser tecnicamente possível de observar pode variar consideravelmente. Os sistemas hipermídia que são exclusivamente baseados em HTML serão somente capazes de guardar quais páginas foram requisitadas ao servidor (o que inclui um esboço dos caminhos de navegação por links de um mesmo site). Os sistemas podem ter mais ou menos controle sobre essa interação, por exemplo, usando Java applets

onde os dados de uso podem ser obtidos a partir de cliques do mouse e movimentos das barras de rolagem.

Sistemas como o jornal adaptativo Krakatoa [Kamba95] observa cada *scroll* e *resize* realizado em um artigo. Quando o usuário executa essas operações, um interesse no artigo é considerado e refletido no modelo do usuário.

Já em [Cingil00], um agente é colocado no cliente para monitorar cada ação do usuário. O agente faz um log do histórico de navegação em um arquivo XML [XML00]. Deste log são extraídas, dinamicamente, importantes informações estatísticas sobre as preferências dos usuários e expressas em um arquivo RDF [RDF99, RDF00].

Outros sistemas utilizam esse histórico de navegação e tempo de permanência em uma página para atualizar o conhecimento do usuário. Em um sistema de e-learning, por exemplo, um conceito pode ser considerado “well learned” se o usuário acessou, por um certo tempo e freqüência, o material referente à ele.

2.2.3 Dados de ambiente

No lado cliente de um sistema baseado na Web, a diversidade de hardware e software usados é enorme. Este cenário vem se estendendo já que o número de dispositivos que acessam a Web, e que possuem capacidade limitada (como PDAs e celulares), está crescendo rapidamente. Dessa forma, a chance de criar uma aplicação fixa para a Web que se adapte aos possíveis ambientes utilizados se tornou incrivelmente menor.

O uso da Web pode ser influenciado tanto por limitações de software e hardware quanto pela localização do usuário.

Ambiente de software

Nem todas as partes de um software suportam todas as características e particularidades existentes e versões anteriores de browsers têm capacidades muito limitadas. Novos tipos de celulares e outros dispositivos limitados virão se adicionar a esta diversidade. Algumas informações são de grande interesse quando se considera o acesso por diferentes ambientes de software: versão do browser e plataforma, disponibilidade de *plug-ins* e o suporte a Java e Javascript.

Deve-se notar que os dados sobre o ambiente podem impactar nos dados do usuário. Por exemplo, a presença de uma versão beta mais recente de um browser em um PC pode indicar que o usuário tem experiência com o uso de computadores.

Ambiente de hardware

Geralmente há uma grande diferença entre o hardware com o qual os desenvolvedores e designers utilizam e o hardware disponível aos usuários. As maiores diferenças são: banda passante (que influencia o tempo de download, sendo um fator significante para a satisfação do usuário), a velocidade de processamento dos dispositivos de acesso (PCs, celulares, PDAs, etc.) e dispositivos de entrada (para um pequeno dispositivo, pode ser difícil oferecer a entrada de dados para um site interativo: por exemplo, se a área de seleção de um elemento de navegação for muito pequena).

Localização

Informações sobre localização podem ser usadas para filtrar o conteúdo, adaptar o formato da apresentação e fazer recomendações baseadas no conhecimento geográfico [Fink01, Jameson01]. Este tipo de informação inclui:

- a localização corrente do usuário: A granularidade da informação sobre a localização pode variar bastante, no caso de aplicações sensitivas à localização, podendo variar de metros a centímetros de acordo com as métricas utilizadas pelo país. Informações adicionais como direção do olhar ou do movimento também podem ser importantes. Elas podem identificar o objeto que o indivíduo está procurando ou indicar a direção.

- Características do local: informações relacionadas ao nível de ruído e sobre os objetos existentes no ambiente também podem ser consideradas.

2.3 Métodos de aquisição dos dados

Para inicializar e atualizar os modelos são necessários métodos de extração de informação sobre os dados apresentados na seção anterior. A maioria dos dados mencionados não podem ser utilizados diretamente para a adaptação. Eles necessitam ser processados para definir o conteúdo inicial de um modelo de usuário ou um modelo de uso.

Nesta seção serão descritos inicialmente os métodos que são tipicamente utilizados para obter suposições explícitas sobre os dados do usuário. A coleção deste tipo de dados é chamada de modelo de usuário. A seguir, serão apresentados os métodos que agregam informações sobre o comportamento interativo do usuário a partir de observações individuais. Os resultados deste tipo de análise, que são geralmente utilizados para predizer o comportamento futuro do usuário, são chamados de modelos de uso. Por fim, serão apresentados os métodos de aquisição de informação sobre a plataforma do usuário.

2.3.1 Métodos de aquisição para o modelo de usuário

Esta seção descreve os métodos de aquisição de dados de usuários e de armazenamento em um modelo de usuário inicial. Primeiramente serão descritos os métodos baseados na entrada explícita do usuário. A seguir, serão discutidos os métodos de aquisição passiva: regras de aquisição, reconhecimento de metas e planos, e estereótipos para classificação de usuários.

Informações fornecidas pelo usuário

Uma estratégia óbvia de aquisição de informações sobre um usuário é deixar que o próprio usuário forneça os dados necessários. Alguns tipos de dados, como os dados demográficos, têm o usuário como a única fonte possível de informação. Esses dados são geralmente requisitados na fase inicial de uso em um sistema.

A maioria dos sites que oferecem personalização também utilizam questionários. O sistema de guia eletrônico de visitas em museus, o HyperAudio [Petrelli99], utiliza maciçamente esta estratégia. Para obter o comportamento do usuário, os dados sobre os *profiles* dos visitantes e estilos de visitas são coletados através de questionários.

No sistema de informações turísticas AVANTI [Fink97], por exemplo, o questionário inicial contém perguntas referentes à familiaridade dos usuários com o computador, com o próprio AVANTI e a localização de interesse.

É importante notar que existe um risco nas declarações explícitas dos usuários: a auto-avaliação pode estar errada já que os usuários freqüentemente não estão cientes de suas próprias capacidades. Portanto, alguns sistemas apresentam perguntas controladas, testes e exercícios, possibilitando uma avaliação mais precisa do usuário.

Regras de aquisição

O processamento de regras de aquisição é a forma mais freqüente de geração de suposições sobre o usuário. As regras de inferência são tipicamente executadas quando novas informações sobre o usuário estão disponíveis. Na maioria dos casos, as regras de aquisição se referem às ações dos usuários ou à interpretação do comportamento do usuário.

O sistema AHAM [Wu99] não define uma sintaxe específica para definição das regras. A seguinte regra diz que quando uma página é "*relevant*" e foi acessada, o valor do conhecimento do conceito correspondente se torna "*well-learnt*".

```
<access(C) and C.relevant = true => C.Knowledge := well-
learnt>
```

Em geral, o que se deseja com esse tipo de regras é ter a opção de se basear alguma adaptação no estado de conhecimento antes do acesso e leitura de uma página.

No sistema de modelagem de usuários BGP-MS [Kobsa95] foi utilizado a idéia de "*dialog acts*" (elementos pragmáticos na interação do usuário com o sistema que freqüentemente têm implicações no modelo de usuário). Como as regras de aquisição, os "*dialog acts*" são independentes da aplicação, mas tornam-se tipicamente mais úteis quando inseridos no cenário da aplicação. A Figura 2 mostra a definição de um tipo de "*dialog act*" na notação usada no BGP-MS. Seu parâmetro (identificando o tópico a ser explicado) também aparece no padrão de suposições (indicando, de forma lógica, que o sistema acredita que o usuário não sabe o tópico em questão). Quando uma instância deste tipo é processada, o valor real do parâmetro *topic* será substituído dentro do padrão, e finalmente uma especificação do conteúdo do modelo de usuário é construída.

```
(define-d-act  : name REQUEST-EXPLANATION
               : parameters (topic)
               : presuppositions ((B S (not (B U topic))))
```

Figura 2: Definição simples de um tipo de dialog act (BGP-MS)

As regras de aquisição que são específicas ao domínio são provavelmente as formas mais populares de aquisição do modelo de usuário, pois são mais simples de implementar.

Contudo, elas não são muito flexíveis e suas propriedades tipicamente não podem ser descritas formalmente.

Reconhecimento de planos

O reconhecimento de planos se refere ao processo de identificação de metas que o usuário pode ter, e da seqüência de ações que podem ser executadas para atingir estas metas. Um sistema de reconhecimento de planos consiste de uma base de conhecimento de tarefas que define as possíveis ações do usuário e relações entre essas ações, e de um mecanismo que identifica o plano atual (e as metas associadas) do usuário a partir das interações observadas.

O reconhecimento de planos é especialmente adequado para aplicações com um pequeno número de metas possíveis e formas de atingir essas metas. Por exemplo, em centros de mensagens e sistemas de informações, os usuários freqüentemente possuem metas específicas como listar as novas mensagens ou receber a previsão de tempo para a região local.

Em [Bushey99] é apresentado um método de geração de modelos de usuários CDM (Método de Categorização, Descrição e Modelagem). Este método foi aplicado a um sistema de telecomunicação para negociações de compra e venda, e foi projetado para acomodar a diversidade de comportamento dos grupos de usuários, o que contribui fortemente para que as metas de negócios sejam atingidas. Este método, essencialmente, permite que o sistema seja customizado para facilitar o comportamento desejado e otimizar comportamentos pré-existentes considerados lucrativos no competitivo ambiente de negociação de serviços.

Estereótipos

O método mais simples para realizar a primeira avaliação dos usuários consiste em classificá-los em categorias, e então fazer previsões com base em um estereótipo que está associado a cada categoria. Os estereótipos contêm as suposições padrões sobre os membros da categoria em questão. Os principais componentes de um estereótipo são: um corpo, que contém as informações que são tipicamente verdadeiras para os usuários aos quais o estereótipo se aplica; e um conjunto de condições de ativação para aplicação do estereótipo ao usuário.

Em aplicações que estabelecem relações com consumidores, os estereótipos podem ser utilizados para descrever as classes de consumidores [ArdGoy99, Ardissono00].

2.3.2 Métodos de aquisição para o modelo de uso

Assim que surgiram os sistemas adaptativos, o uso de métodos de representação do conhecimento para a modelagem de usuários foi o foco freqüente de pesquisa. Recentemente, contudo, a aplicação de técnicas de aprendizado para controlar a interação adaptada ao usuário tem se tornado popular.

Em vários sistemas atuais, o comportamento do usuário não só é observado como também modelado como uma base direta para a personalização do sistema, como em [Cingil00].

Geralmente, os agentes de interface aprendem as correlações entre as situações que o usuário encontra e as ações que executa. Estes dados são utilizados para prever o comportamento do usuário em situações futuras, sugerir as ações apropriadas, e talvez executar automaticamente algumas ações em nome do usuário.

A abordagem de modelagem de usuário LaboUr (Learning about the User) combina técnicas de aprendizado e de representação do conhecimento. O ELFI (Eletronic Funding Information) é um sistema de informação de fundos de pesquisa. Ele implementa as idéias do LaboUr [Pohl99].

A adaptatividade no ELFI é baseada na análise do *log* de uso. O ELFI grava todas as interações do usuário com o sistema em um arquivo de *log.* Este *log* de uso é explorado de três formas diferentes:

- Os arquivos de *log* são escaneados por seqüências de chaves que provavelmente indicam o interesse do usuário em um tópico de pesquisa específico. Por exemplo, se um usuário faz seleções subseqüentes em uma lista de documentos relacionados com um tópico específico, então o usuário está provavelmente interessado neste tópico.

- A atividade de navegação, ou seja, a seleção de itens na árvore de navegação é analisada por um componente de aprendizado. Seu objetivo é descobrir os itens de dados usados freqüentemente e as transições freqüentes entre os mesmos. Os resultados podem ser utilizados para gerar uma árvore de navegação personalizada.

- Finalmente, os algoritmos de aprendizado são empregados dentro de um componente de aprendizado para adquirir as informações de interesse, baseado na seleção de visões detalhadas. Isto é problemático, pois estas seleções somente

oferecem exemplos de aprendizado positivo. Os resultados do aprendizado geram uma avaliação dos interesses dos usuários relativos ao conteúdo da visão detalhada.

Dessa forma, o processo de aprendizado deste sistema permite a previsão das ações que um usuário pode executar em uma situação particular, além de prever a seqüência de ações.

2.3.3 Métodos de aquisição para o modelo de ambiente

Os métodos para aquisição de dados sobre o ambiente de software e hardware ou sobre a localização do usuário serão discutidos separadamente. Um problema importante se refere ao mapeamento de um dispositivo físico a um indivíduo específico. Em alguns contextos, como PCs, é razoável considerar que o sistema é de um único usuário. Neste caso, os dados de ambiente podem ser armazenados juntamente com os demais dados do usuário. Esta abordagem não é possível em outras situações, como em quiosques de informações em *showrooms*, já que são inerentemente multi-usuário. Neste caso, os dados de ambiente devem ser armazenados separadamente dos dados do usuário.

Ambiente de software

A maioria dos sites leva em conta as restrições do browser do usuário. As informações sobre o cliente podem ser obtidas a partir do cabeçalho de requisições HTTP que são recebidas pelo servidor. Cada requisição carrega valores para um número de variáveis. Um exemplo deste tipo de informação pode ser a requisição HTTP GET:

```
GET/index.html HTTP/1.0
Accept: */*
User-Agent: Mozilla/3.03 [en] (WinNT;I)
```

Ambiente de hardware

As restrições de hardware são geralmente difíceis de acessar. Alguns aspectos podem ser inferidos a partir do tipo de browser utilizado (em dispositivos móveis é necessário um software especial para acesso à Web). Outras características de hardware, como banda passante e velocidade de processamento, são mais difíceis de determinar. Particularmente, o montante efetivo de banda passante disponível depende da rota que a página deve seguir pela Internet para atingir a máquina cliente, e esse montante pode mudar significativamente entre curtos períodos de tempo.

Localização

As informações sobre a localização de um usuário podem ser adquiridas por uma grande variedade de métodos. Para dispositivos estacionários, a informação sobre a localização relativa a outros objetos é geralmente gravada em um banco de dados (por exemplo, o mapeamento de um dispositivo como uma impressora para uma sala ou prédio específico). O desenvolvimento de padrões e interfaces (Jini, XML) possibilitarão a consulta a esses dispositivos de forma direta. Os dispositivos móveis podem indicar sua localização de forma ativa ou passiva tanto pelo uso de tecnologia dedicada, como sensores e transmissores em ambientes fechados, como por tecnologia geral, como GPS (Global Position Systems).

Geralmente, a combinação de métodos pode gerar melhores resultados. Por exemplo, a precisão necessária para determinar em qual direção um veículo está se deslocando em uma estrada para a geração de alertas de tráfego difere do nível de detalhe e velocidade de atualização necessária para suportar a navegação em um prédio não conhecido como um museu.

2.4 Como realizar a adaptação?

A adaptação em um sistema hipermídia pode ser realizada em diferentes níveis:

- Adaptação de conteúdo: descreve os métodos de personalização do conteúdo dos objetos hipermídia e das páginas de acordo com os dados do usuário, de uso e de ambiente.
- Adaptação da apresentação e da modalidade: descreve os métodos para mudança da apresentação e do formato da mídia.
- Adaptação da estrutura: descreve os métodos para personalização da apresentação dos links.

2.4.1 Adaptação de conteúdo

Um número de técnicas de diferentes níveis de granularidade tem sido desenvolvido para adaptar o conteúdo às diferentes necessidades dos usuários. Algumas técnicas descritas abaixo se assemelham às técnicas utilizadas pelo *framework* proposto, como as técnicas de variantes de página e de fragmentos.

A organização e níveis de detalhes das notícias apresentadas na página principal do *MyNews*, por exemplo, deve ser previamente definida pelo instanciador do *framework*. Contudo, estas diretivas podem ser sobrepostas com base na receptividade do usuário, permitindo uma mudança do nível de detalhe do conteúdo a ser apresentado.

Variantes de página

Com esta técnica, um sistema mantém duas ou mais variantes de uma mesma página com apresentações diferentes do mesmo conteúdo. Como uma regra, cada variante está preparada para um dos possíveis estereótipos de usuário. Logo, ao apresentar uma página, o sistema seleciona a variante da página de acordo com o estereótipo de usuário. Esta técnica pode ser usada com estereótipos de *background*, com estereótipos de conhecimento, e com estereótipos que refletem a proficiência do usuário com o idioma de apresentação. Por exemplo, para adaptar a apresentação de acordo com o *background* do usuário, o sistema armazena vários exemplos que ilustram um conceito particular e oferece ao o usuário o exemplo que é mais adequado à sua experiência prévia e aos seus interesses. Variantes de página têm sido usadas para usuários pertencentes a diferentes estereótipos como em ORIMUHS [Encarnação99] e KBS Hyperbook [Henze00].

Variantes de fragmento

Nesta abordagem as variantes são criadas para cada fragmento de página adaptativo. Neste caso uma página não é equivalente a um conceito como em alguns sistemas e pode incluir explicações sobre vários conceitos. O sistema armazena várias variantes de explicação para cada conceito e o usuário obtém a página que inclui as variantes que correspondem ao seu conhecimento sobre os conceitos apresentados na página. Isto é sustentado pela idéia de que usuários com conhecimentos diferentes sobre um determinado conceito precisam de explicações estruturalmente diferentes sobre este conceito.

Esta abordagem é tecnicamente mais difícil que a variante de página pois ela requer a geração dinâmica das páginas em tempo de execução. O sistema AVANTI [Fink97], com este propósito, incluiu a composição de páginas utilizando *middleware*.

Uma combinação de variantes de página e variantes de fragmento pode apoiar a adaptação para o *background* e para o conhecimento do usuário. A variante da página atual para um nó acessado é selecionada de acordo com o *background* do usuário. Esta página pode ser

adaptada ainda mais: para cada conceito mencionado na página, o sistema seleciona a explicação que é mais satisfatória de acordo com o nível de conhecimento do usuário.

Esta modalidade pode se reduzir à modalidade anterior (variante de página), quando a escolha de uma página se reflete na escolha de outra.

Stretchtext

O *stretchtext* é uma técnica de nível mais alto que também pode ligar ou desligar partes diferentes do conteúdo de acordo com o conhecimento de usuário. Em um hipertexto regular, o resultado da ativação de uma palavra chave é mover para outra página relacionada. No *stretchtext* este texto relacionado pode simplesmente substituir a palavra chave ativada (ou uma frase com esta palavra) expandindo o texto da página atual. Se for preciso, este texto expandido pode ser contraído novamente para a palavra chave. O sistema de informação online KN-AHS [Kobsa94] usa esta técnica de personalização.

A idéia de apresentação de *stretchtext* adaptativa é apresentar uma página pedida com todas as informações (extensões de stretchtext) que não são relevantes para o usuário contraídas e todas as extensões relevantes expandidas. Para alcançar este resultado, um autor pode declarar alguma informação textual expandida contida em um nó como uma explicação adicional de um conceito particular, ou como um nível mais detalhado de um conceito particular. Opcionalmente, um usuário, com alto grau de conhecimento sobre um conceito, sempre receberá as explicações adicionais deste conceito ocultadas (ou contraídas) e todos os detalhes de nível mais baixo expandidos. Por outro lado, um usuário com pouco conhecimento sobre um conceito sempre terá as explicações adicionais sobre este conceito visíveis e os detalhes de nível mais baixo contraídos. O usuário com um nível médio de conhecimento verá ambos os tipos de informação.

Uma característica importante da técnica de stretchtext adaptativa é que ela deixa tanto o usuário quanto o sistema adaptarem o conteúdo de uma determinada página e pode levar em conta não só conhecimento do usuário, mas também as suas preferências. Depois de apresentar a página de stretchtext, ela pode ser adaptada pelo usuário que pode expandir ou contrair as explicações e os detalhes de acordo com as suas preferências. O sistema atualiza o modelo de usuário de acordo com as preferências demonstradas para assegurar que o usuário sempre verá uma combinação preferida de partes expandidas e contraídas. Por exemplo, se o usuário tem as explicações adicionais sobre um determinado conceito

contraídas, o sistema sempre mostrará as explicações adicionais deste conceito contraídas até o usuário mudar as suas preferências.

Texto Condicional

É uma técnica simples, mas efetiva para adaptação de conteúdo. Com esta técnica, toda a informação existente sobre um conceito é dividida em vários textos. Cada texto é associado a uma condição sobre o nível de conhecimento do usuário representado no modelo do usuário. Quando o sistema apresenta a informação sobre o conceito ele só mostra as partes onde a condição é verdadeira.

Esta técnica é uma técnica de baixo nível, pois requer algum trabalho de programação do autor para fixar todas as condições exigidas, mas também é muito flexível. Escolhendo condições apropriadas no nível de conhecimento do conceito atual e nos conceitos relacionados representados no modelo de usuário, o autor pode implementar todos os métodos de adaptação listados acima exceto ordenação. Um exemplo simples é o de esconder textos com explicações irrelevantes se o nível de conhecimento do usuário sobre o conceito atual for bastante bom, ou ativar uma parte de texto com explicações comparativas se o conceito relacionado correspondente já for conhecido.

2.4.2 Adaptação da apresentação

Adaptações em uma apresentação hipermídia são aquelas onde o conteúdo da informação de um objeto hipermídia idealmente permanece o mesmo (em contraste com as mudanças discutidas na seção anterior), enquanto o formato e *layout* dos objetos mudam. Um tipo especial de mudança de apresentação é a mudança de modalidade, por exemplo, de imagens para texto, de texto para áudio, ou de um vídeo para imagens. Nas aplicações existentes, geralmente são encontradas adaptações realizadas em conjunto com a adaptação de conteúdo e de estrutura.

A adaptação em sistemas hipermídia referentes às apresentações multimídia é geralmente baseada nas preferências dos usuários.

Alguns sistemas, como o *Cuypers* [Ossenbruggen01], realiza a geração da apresentação multimídia baseando-se nos modelos de usuário e plataforma.

O sistema AVANTI [Fink97], por exemplo, baseia-se na seleção de diferentes modalidades referentes às habilidades físicas dos usuários.

Esta modalidade de personalização é bastante discutida na literatura e não será explorada neste trabalho.

2.4.3 Adaptação da estrutura

Enquanto a adaptação de conteúdo muda a informação que é apresentada nas páginas, e a adaptação da apresentação muda a forma como a informação é comunicada ao usuário, a adaptação da estrutura se refere às mudanças na forma com que a estrutura de links dos documentos ou sua apresentação ao usuário são modificadas.

O *framework* de personalização proposto realiza esta forma de adaptação a partir da técnica de classificação. Uma ontologia de domínio é utilizada para a contextualização do conteúdo.

Nesta seção vamos discutir as principais técnicas utilizadas na adaptação da estrutura.

Orientação Direta

A *Orientação Direta* é talvez a mais simples das técnicas de suporte à navegação adaptativa e sua aplicação consiste em decidir, em cada ponto da navegação, qual o *melhor* nó a ser visitado a seguir, levando em conta os objetivos, preferências, conhecimento e outros parâmetros representados no modelo do usuário. Para oferecer orientação direta, o sistema pode destacar visualmente o elo para o *melhor* nó ou apresentar um elo dinâmico adicional (normalmente denominado “next”) que é conectado ao melhor nó selecionado.

A orientação direta é uma técnica clara e fácil de implementar que pode ser empregada sobre todas as quatro classes de elos anteriormente citadas. O problema é que, em geral, ela não oferece suporte aos usuários que não quiserem seguir a sugestão do sistema, o que é um indicativo de que, apesar de útil, esta técnica não deve ser empregada como único recurso de adaptação em um sistema.

Classificação

A técnica de *Classificação Adaptativa* consiste em classificar todos os elos partindo de um nó de acordo com a sua *relevância*, calculada sobre o modelo do usuário. Os elos são apresentados então em ordem decrescente desta relevância. A classificação adaptativa possui uma aplicação limitada: pode ser empregada satisfatoriamente com elos não-contextuais, mas seu uso com índices e tabelas de conteúdos é muito difícil e a técnica não pode ser absolutamente usada para elos contextuais nem para mapas. Entretanto, esta

técnica parece ser útil em recuperação de informações. Resultados empíricos mostraram que a classificação adaptativa pode reduzir significativamente o tempo de navegação em aplicações de recuperação de informações, onde cada nó pode apresentar muitos elos não-contextuais. A classificação adaptativa pode ainda ser empregada em sistemas de documentação on-line e em hipermídia educacional.

Ocultação

A técnica de *Ocultação* é a mais freqüentemente empregada em navegação adaptativa e consiste em restringir o espaço de navegação ocultando os elos para nós não relevantes. Um nó pode ser considerado não-relevante por diversas razões, por exemplo, se não está relacionado com o objetivo corrente do usuário, ou este não possui conhecimento suficiente para entendê-lo.

De um ponto de vista superficial, a ocultação parece ser a técnica mais óbvia e fácil de implementar. Seu uso protege o usuário da complexidade de um hiperespaço irrestrito e reduz a sua sobrecarga cognitiva. A ocultação possui ampla aplicação, podendo ser empregada com todos os tipos de elos. Com elos não-contextuais, índices e mapas a ocultação se dá tornando invisíveis ao usuário botões, itens de menu e áreas *clicáveis* irrelevantes aos seus objetivos. Com elos contextuais, a técnica é eliminar a diferenciação entre as palavras associadas a elos e o texto comum. A ocultação é também mais transparente ao usuário e produz uma apresentação mais estável do que a classificação adaptativa.

Anotação

A idéia da Anotação Adaptativa é aumentar a informação presente nos elos com alguma forma de anotação ou comentário que podem dizer mais sobre o estado corrente dos nós a que se conectam. Esta informação adicional pode ser oferecida sob a forma de texto ou sob a forma de indicadores visuais, tais como ícones especiais, cores ou tamanho dos caracteres. Em geral a anotação oferecida em hipermídia convencional é do tipo estático, isto é, independente do usuário. Em se tratando de hipermídia adaptativa, entretanto, tem-se a expectativa que a anotação seja dinâmica, orientada ao modelo do usuário.

A anotação pode ser usada naturalmente com todos os quatro tipos de elos considerados. Seu uso mantém uma ordenação estável dos elos e, em geral, é uma técnica mais poderosa do que a ocultação, na medida em que esta última oferece somente dois estados possíveis

para os elos (relevante/visível e não-relevante/oculto), enquanto que a anotação adaptativa pode estabelecer diversos níveis de relevância. Além disso, a anotação pode simular a ocultação simplesmente obscurecendo os itens considerados não-relevantes. O obscurecimento pode reduzir em alguma extensão a sobrecarga cognitiva (o usuário pode aprender a ignorar os elos obscurecidos) mas estes se manteriam ainda visíveis (e operacionais, se for necessário).

A idéia de métodos de adaptação baseados em anotação é informar o usuário acerca do estado corrente dos nós, além do que é visível nos elos explícitos. São quatro os métodos mais utilizados nesta categoria. Primeiro, a anotação pode ser usada para apresentar as diversas graduações da relevância dos nós para os objetivos do usuário. O segundo método consiste em refletir os diversos níveis de conhecimento do usuário acerca dos nós. Os dois outros métodos empregam anotação onde usualmente é usada a ocultação. Um deles consiste em *sublinhar* os elos que estão diretamente relacionados com o objetivo corrente. A outra é oferecer anotação especial para os elos que o usuário ainda não está preparado para aprender.

Mapas Adaptativos

A técnica dos *Mapas Adaptativos* compreende diversas formas de adaptação de mapas de hipermídia global e local apresentados ao usuário. Como foi visto, as técnicas de orientação direta, ocultação e anotação podem também ser usadas na adaptação de mapas hipermídia, entretanto o emprego de tais técnicas não modifica a forma ou a estrutura dos mapas. A pesquisa realizada na área da interação humano-computador oferece diversas técnicas para a adaptação da forma e estrutura de diversos tipos de redes, incluindo mapas hipermídia.

2.5 Sistemas Adaptativos

Os sistemas adaptativos utilizam um modelo conceitual do domínio e usam os conceitos para rotular as informações armazenadas ou para gerar descrições do conteúdo da informação e, às vezes, para gerar descrições gráficas do domínio de conhecimento ou da forma que a informação está estruturada. Apesar destas observações apontarem para um *framework* para especificação da informação, a maioria dos sistemas não propõem a utilização de um *framework* .

As principais contribuições relacionadas à apresentação e a seleção de conteúdo personalizadas abordam estas questões considerando um domínio específico, como: Krakatoa Chronicle [Kamba95] e OtaOnLine [Ardissono99a, Ardissono00], no domínio de notícias; AVANTI [Fink97, Fink99], sobre informações turísticas; e o SETA [Ardissono99, Ardissono00a, ArdGoy99], relacionado a comércio eletrônico.

No que se refere à técnica utilizada no processo de geração do conteúdo personalizado, trabalhos como [Bradley000] e [Khan00] utilizam ontologias para comparação de consultas e documentos a partir de métricas de distância conceitual. No AHAM [Wu99a] esse processo é definido a partir de regras contidas no próprio conteúdo ou contidas num mecanismo de inferência.

Em [Bordegoni97] foi proposta uma arquitetura de referência para sistemas de apresentação multimídia inteligente. A Figura 3 mostra o esquema da arquitetura.

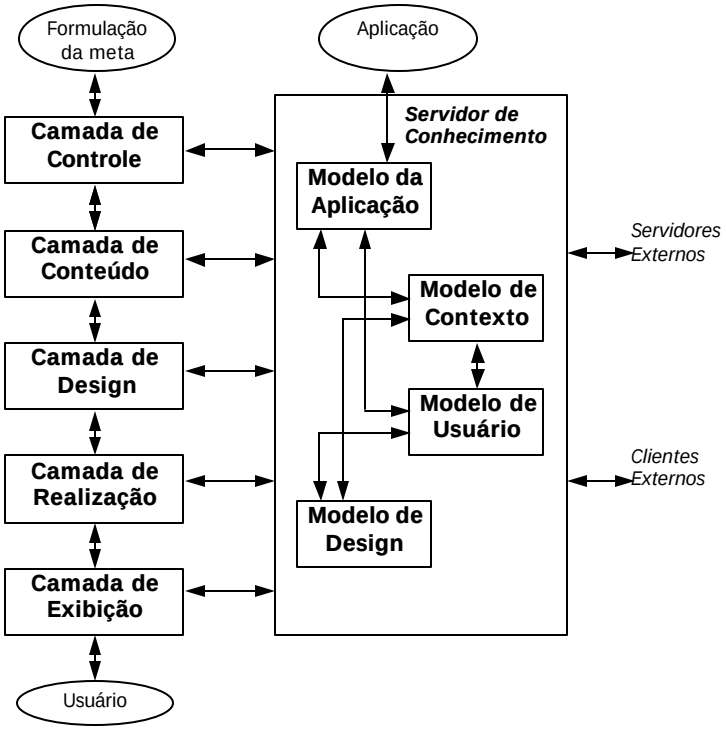


Figura 3: Arquitetura de referência para IMMPSs

À esquerda estão os processos envolvidos em tratar uma simples interação de usuário, como clicar em um elo (que é um modo primitivo de formular uma meta). O lado direito mostra partes funcionais do sistema de hipermídia adaptativa que estão envolvidas em manter os diferentes modelos (modelo de design, modelo de usuário, modelo de contexto) e projetar apresentações. A descrição das camadas é a seguinte:

- Camada de Controle: na qual o elo (URL) é resolvido. Isto significa que quando o elo não identifica uma única página Web, mas um conceito composto, o sistema de hipermídia adaptativa tem que deduzir (do modelo de domínio e do modelo de usuário) que página mostrar.

- Camada de Conteúdo: onde o conteúdo apropriado é selecionado e recuperado. Em um sistema Web o conteúdo é uma página Web. A página Web pode ser gerada ou pode ser montada a partir de fragmentos. A seleção dos fragmentos apropriados é realizada nesta camada.

- Camada de *Design:* é responsável pela montagem de página a partir dos fragmentos selecionados. Os fragmentos podem precisar ser classificados e os elos podem ser designados para diferentes classes de elo para tornar possível a anotação de elo. O modelo de IMMPS distingue três possíveis modos para alcançar uma página Web corretamente projetada que satisfaça exigências específicas do usuário, como também exigências específicas da plataforma: *layout* depois da produção, *layout* antes da produção e *layout* intercalado com a produção. Separadamente da adaptação baseada no modelo de usuário, o sistema de hipermída adaptativa tem que adaptar também a página Web para a interface do usuário (Web-browser) que será usada para ver a página. O código HTML enviado a um computador com uma tela de alta resolução será diferente daquele enviado a uma Web-TV ou para um PDA por exemplo.

- Camada de Realização: é responsável por finalizar a página Web de forma que ela possa ser exibida pelo browser. Uma das tarefas desta camada é acrescentar o *style sheet* apropriado. O *style sheet* determina os aspectos de *layout* do conteúdo da página (fontes, alinhamento, etc.) como também para os elos (cor do elo para associação de classe).

- Camada de Exibição: representa a visualização da página atual, como executada pelo Web-browser.

Sua principal contribuição é o fato de esta arquitetura refletir uma visão independente da implementação do processo requerido para geração das apresentações multimídia.

Um outro trabalho, que se baseou no IMMPSs, é o sistema de geração automática de apresentação multimídia *Cuypers* [Ossenbruggen01, Geurts01]. O *Cuypers* é um protótipo

desenvolvido para a geração de apresentações multimídia para a Web a partir de um banco de dados multimídia semi-estruturado. Dada uma estrutura retórica [Rutledge00] (ou outro tipo de estrutura semântica pragmática) e um conjunto de regras, o sistema gera uma apresentação que se adequa às limitações da plataforma e às preferências dos usuários.

Como a adaptação parte de uma abstração de alto nível do documento, que expressa apenas suas relações semânticas, o processo de adaptação não é trivial. Assim como um designer humano tem dificuldades para chegar a uma apresentação ideal da informação, o mesmo acontece com o processo automático. O processo de criação da apresentação não é linear. As soluções para um problema estético (como a formatação do *layout* de um documento que deve ser apresentado em um PDA) pode acarretar problemas semânticos e pragmáticos e vice-versa.

O *Cuypers* foi projetado para operar no contexto de uma arquitetura cliente/servidor, como ilustrado na Figura 4. A figura mostra dois servidores diferentes: um dedicado ao fornecimento de mídias contínuas (como áudio e vídeo) ao cliente, e outro responsável pelo fornecimento dos demais dados, incluindo a própria apresentação multimídia (que utiliza o padrão SMIL [SMIL98]). O núcleo do sistema *Cuypers* recebe, de um sistema de extração de informação, uma descrição semântica do documento multimídia como entrada e envia a apresentação gerada ao servidor para posterior entrega ao cliente.

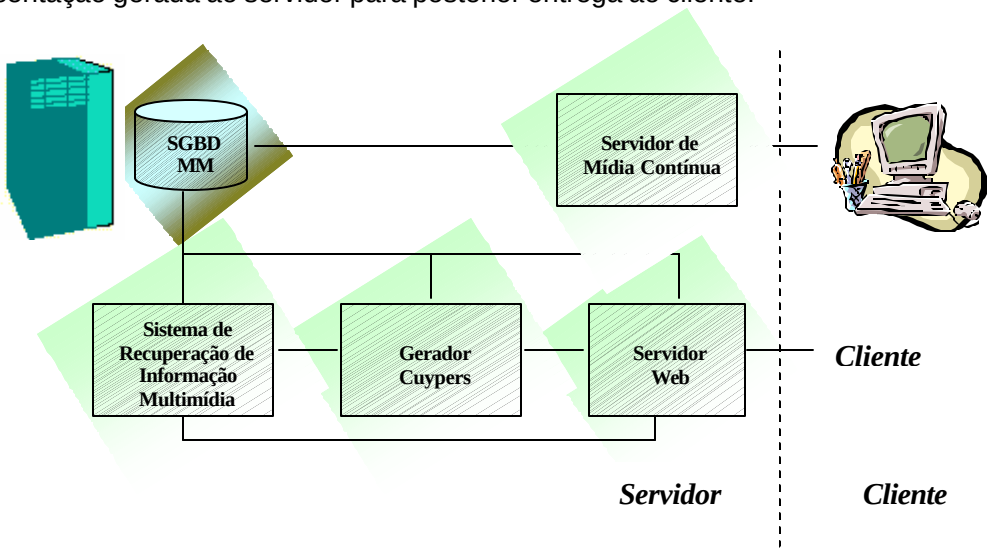


Figura 4: O ambiente do motor de geração Cuypers

O processo de geração no *Cuypers*, como mostrado na Figura 5, é subdividido em cinco camadas de abstração conceitual discutidas com detalhes em [Ossenbruggen01].

- *Estrutura semântica*: A estrutura semântica é uma descrição semântica de alto nível da apresentação. Esta estrutura é totalmente abstrata, não endereçando as questões de *design* e *layout* da aplicação. O projetista fornece um conjunto de regras de transformação desta estrutura em uma especificação baseada em *communicative devices*.

- *Communicative devices*: São construtores abstratos que especificam como a informação deve ser comunicada ao usuário. Os *communicative devices* são independentes dos dispositivos de acesso e podem ser usados para comunicar apresentações similares para diferentes dispositivos. Um exemplo de um *communicative device* é o "*bookshelf*" [Rutledge00a], que assegura que a seqüência de itens de mídia serão apresentados em uma ordem específica. O *layout* que é escolhido para este fim pode depender da capacidade dos dispositivos de acesso, preferências do usuário, etc.

- *Restrições qualitativas*: São utilizados para especificar os *communicative devices* em um alto nível.

- *Restrições quantitativas*: Fazem a transformação de restrições qualitativas em restrições numéricas. Neste nível, pode-se determinar a posição exata das diferentes mídias da apresentação e se elas se adequarão à tela, se não excederão um limite de tempo dado, etc.

- *Apresentação final*: Finaliza a apresentação em um formato adequado ao *player* ou browser do dispositivo de acesso do usuário. O *Cuypers* atualmente foca na geração de um documento SMIL.

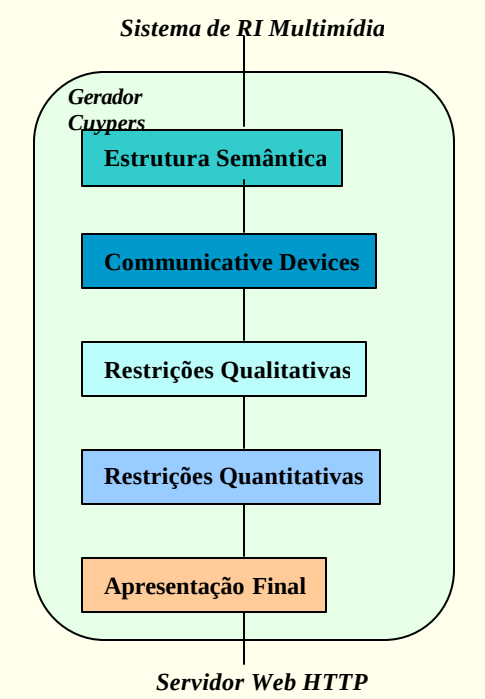


Figura 5: Níveis de abstração do motor de geração do Cuypers

Trabalhos como o de [Bordegoni97] e [Geurts01] endereçam o processo de geração automática de documentos multimídia, propondo uma arquitetura genérica e definindo estratégias para realizar a adaptação da apresentação dos documentos considerando as preferências e plataforma de apresentação do usuário. Estes trabalhos foram definidos baseados em modelos de documentos mais elaborados como o SMIL.

Esta dissertação propõe uma abordagem genérica similar à da Figura 3, voltada para o modelo de documentos da Web, mas focando as fases de filtragem dos documentos e geração abstrata da apresentação. Dessa forma, a contribuição deste trabalho pode ser posicionada em um nível anterior ao motor de geração *Cuypers*. O resultado do processo de empacotamento proposto, devidamente adicionado de intenções comunicativas, poderia representar a estrutura semântica utilizada como entrada pelo *Cuypers*.

Comparações adicionais com trabalhos correlatos encontram-se na seção 5.1.

3. Um Framework para Personalização de Documentos Hipermídia

Este capítulo discute o *framework* proposto para personalização de documentos hipermídia, apresentando a arquitetura proposta e uma breve descrição dos seus *hotspots*. Ele descreve e formaliza cada etapa do processo de personalização, especificamente as etapas de filtragem e empacotamento dos documentos, enfatizando os aspectos de modelagem.

3.1 Introdução

O objetivo da proposta de definição de um *framework* para o processo de filtragem e empacotamento de documentos hipermídia é analisar, da forma mais independente possível, as questões relativas à filtragem e empacotamento dos documentos, modelagem de usuários e adaptação da apresentação, utilizando os conceitos de *separation of concerns.*

O processo de empacotamento é responsável pela definição da estrutura abstrata da aplicação, delineando a navegação e estruturação do conteúdo, independente das questões de formatação.

Essa abordagem facilita a construção de uma variedade de aplicações em domínios diversos a partir de múltiplas instanciações do *framework* usando diferentes algoritmos e configurações. Esta característica é essencial para os sistemas adaptativos, já que pode exigir refinamentos sucessivos dos algoritmos de acordo com o sucesso ou limitações de cada configuração.

O *framework* foi implementado utilizando a linguagem Java (jdk1.1.3). Para dar suporte às linguagens de marcação XML e XSL, foram utilizados os *parsers* Xerces (XML4J 3.2.1) e Xalan (xalan-j_2_3_1).

Para ilustrar o *framework* proposto e os processos de personalização, será utilizada como exemplo uma instanciação do *framework* para o domínio de notícias – o *MyNews*. O *MyNews* cria um jornal personalizado selecionando o conteúdo e as estruturas de

apresentação adequados ao usuário em questão. A segunda instanciação, o *MyRefs*, será apresentada na seção 4.2.

3.2 Arquitetura

O propósito geral da arquitetura é delinear o processo de adaptação pelo qual o conjunto de documentos, que foi requisitado pelo usuário, deve passar antes de ser apresentado. Este processo envolve várias etapas para que finalmente o conjunto de documentos seja apresentado obedecendo aos modelos de usuário, empacotamento e formatação.

A análise de domínio de aplicações que realizam adaptação revelou a importância de várias tarefas, dentre elas:

- Gerenciar as várias fases de interação com o cliente. Em cada fase o sistema deve decidir se necessita extrair informações do usuário, quais ações podem ser executadas, etc. Além disso, as ações do usuário e as variações relacionadas à plataforma devem ser monitoradas. Essas informações são utilizadas para adaptar o estilo de interação do sistema às necessidades do usuário.

- Decidir como adaptar o estilo de interface do sistema:

    - ao usuário: onde o conteúdo apresentado deve ser direcionado às características e interesses do usuário, dependendo do seu comportamento;

    - ao ambiente: onde a aplicação deve considerar, na apresentação, as condições da rede e o dispositivo onde será apresentado e as preferências de apresentação do usuário.

- Geração dinâmica do conjunto de documentos:

    - Apenas alguns documentos podem tipicamente ser apresentados em conjunto, enquanto outros devem ser gerados em tempo de execução dependendo dos objetivos dos usuários em curto prazo.

    - Os documentos devem ser adaptados (convertidos) para diferentes plataformas: PC, celulares, PDAs, iTV, etc.

A Figura 6 apresenta uma arquitetura funcional de um sistema de personalização de documentos. As estruturas arredondadas correspondem à base de conhecimento do sistema. Elas representam as configurações que irão delinear o comportamento de cada

estágio do processo de personalização proposto. Esta atuação é representada pela ligação existente entre os modelos e respectivos subsistemas.

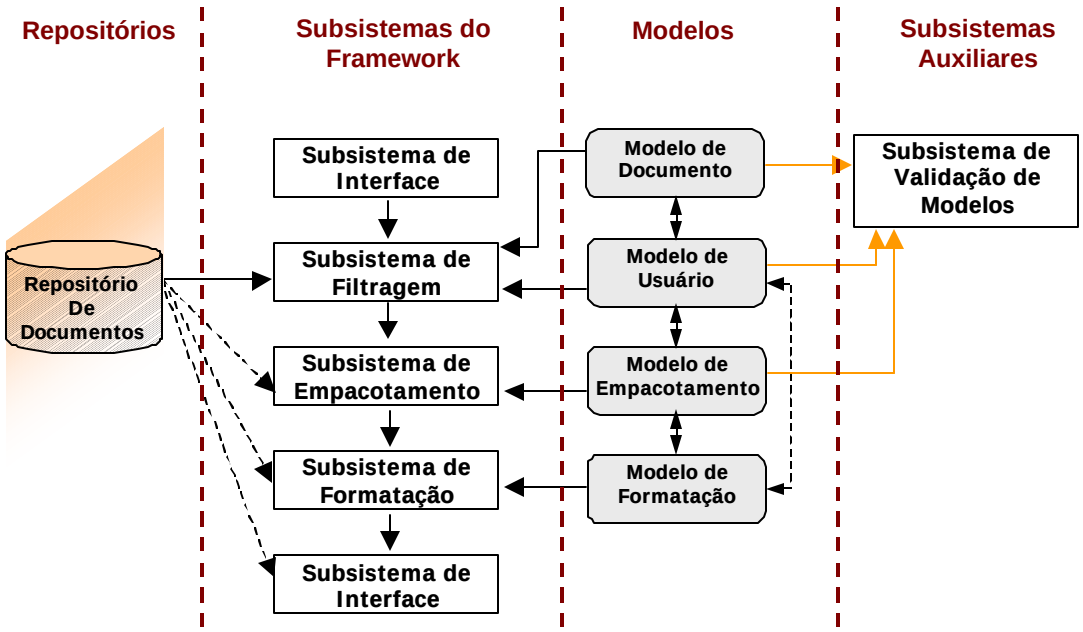


Figura 6: Arquitetura Funcional

Para a definição dos *framework*, os termos “meta-modelo” e “modelo” foram utilizados com o seguinte significado:

- *meta-modelo*: refere-se à definição de um modelo genérico voltado para o domínio de aplicações tratado pelo *framewok*. O meta-modelo de documentos, por exemplo, define um modelo hipermídia específico.

- *modelo*: refere-se à especificação da estrutura do documento para uma aplicação particular, ou seja, para uma instância do *frameworlk*.

O repositório de documentos representa o conteúdo da aplicação e seus metadados. Como ilustrado, este repositório é inicialmente acessado pelo subsistema de filtragem. No decorrer do processo de personalização há duas possíveis opções de projeto: o subsistema de filtragem acessa a base, detendo todo o conteúdo acessado e repassando-o para os estágios posteriores; ou os subsistemas de filtragem, empacotamento e formatação

baseiam-se apenas nos metadados, e o subsistema de interface é o único que acessa o conteúdo.

O processo de personalização se inicia quando o usuário requisita uma visão personalizada do conjunto de documentos contido no repositório. Esta requisição é processada pelo Subsistema de Interface e despachada para o Subsistema de Filtragem, que é responsável pela criação de um conjunto ordenado de documentos, baseando-se nos modelos de documento e usuário. Estes modelos serão descritos nas seções 3.4 e 3.5, respectivamente.

O Subsistema de Empacotamento é responsável por agrupar, reordenar e reestruturar o conjunto de documentos resultante do processo de filtragem, baseando-se no modelo de empacotamento descrito na seção 3.7. O resultado desse processo representa a estrutura abstrata da organização da aplicação.

O Subsistema de Formatação cria o *layout* final e o *design* visual de cada documento de acordo com o modelo de formatação. Esta tarefa vem sendo discutida em diversos trabalhos na literatura e não será abordada nesta dissertação.

Por fim, para fazer a inicialização e manutenção dos modelos, o *framework* conta com o Subsistema de Validação dos Modelos.

O *framework* proposto contém vários *hotspots* a fim de obter um alto grau de *separation of concerns.* Apesar de a arquitetura não variar, os modelos e algoritmos devem diferir de um domínio para outro. Os *hotspots* do *framework* são:

- ***Modelo de documento***: define a estrutura do documento, incluindo os metadados que classificam os documentos;
- ***Modelo de usuário***: permite a definição das restrições capturando os interesses dos usuários, ou seja, define uma visão sobre o repositório de documentos;
- ***Algoritmo de filtragem***: constrói uma consulta combinando os modelos de documento e usuário, e retornando um conjunto de documentos ordenado segundo a relevância dos documentos para o usuário.
- ***Modelo de empacotamento***: define quais partes da estrutura dos documentos devem ser apresentadas, sua organização e estruturas de apresentação.

- ***Algoritmo de empacotamento***: usa o modelo de empacotamento para reagrupar, reordenar e reestruturar os documentos filtrados;

- ***Modelo de formatação***: define como os conjuntos de documentos, resultantes do empacotamento, devem ser exibidos para o usuário, incluindo dependências de plataforma.

- ***Algoritmo de formatação***: usa o modelo de formatação para criar o layout da apresentação final dos conjuntos de documentos resultantes do empacotamento.

3.3 Subsistema de Validação dos Modelos

Este subsistema é responsável por validar sintaticamente o modelo de documento, o modelo do usuário e o modelo de empacotamento. Ele opera com o auxílio de três meta-modelos definidos como XML Schemas [Schema01]. Os modelos em si são definidos como documentos XML. A Figura 7 ilustra o processo de validação dos modelos.

Brevemente, o documento *ModeloDocumento.xml*, se refere à definição da estrutura do conteúdo da aplicação e é validado contra o XML Schema *MetaModeloDocumento.xsd.* O papel do validador do modelo de documentos é exatamente verificar se o *ModeloDocumento.xml* satisfaz o *MetaModeloDocumento.xsd*.

O *ModeloVisoes.xml* é validado utilizando o *ModeloDocumento.xml* e o *MetaModeloVisoes.xsd*.

O *ModeloEmpacotamento.xml* é validado, por sua vez, utilizando o *ModeloVisoes.xml* e o *MetaModeloEmpacotamento.xsd*.

Nas próximas seções, estes modelos serão discutidos detalhadamente.

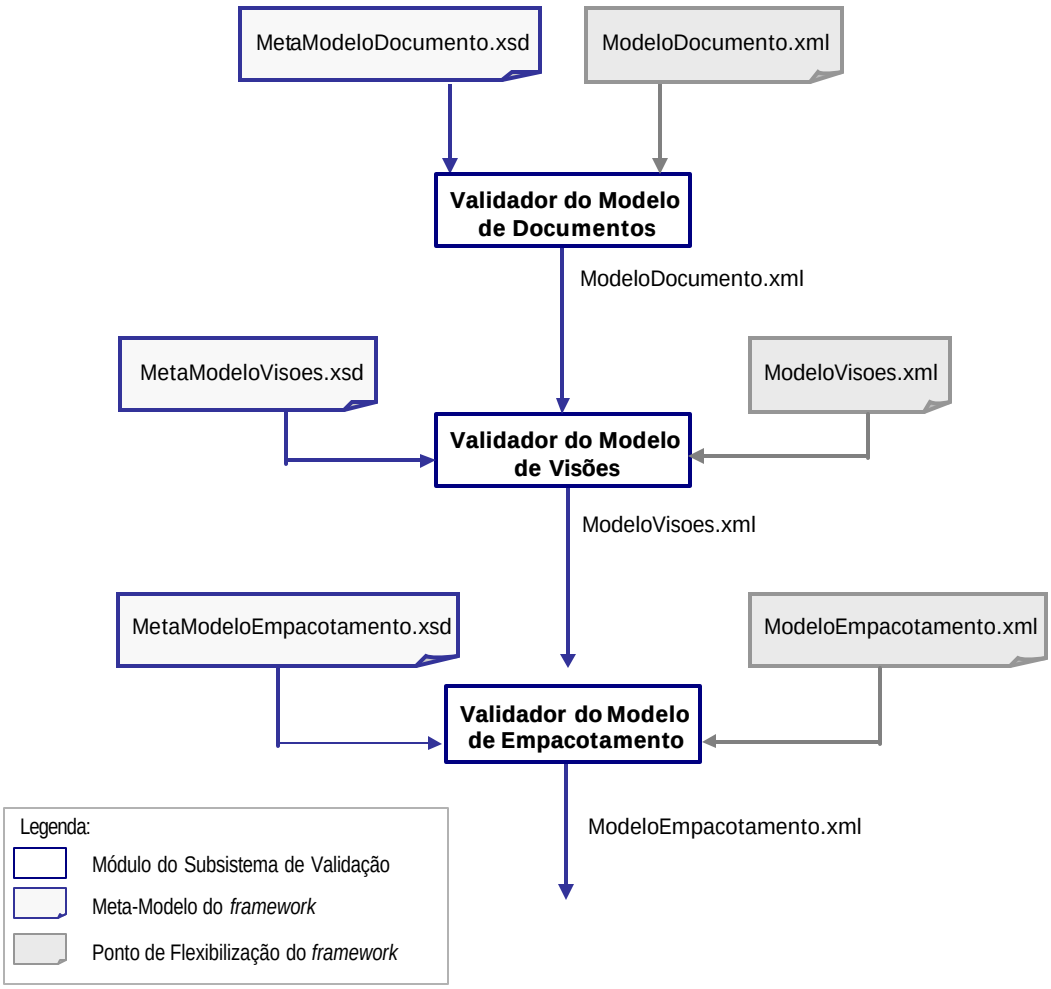


Figura 7: Esquema Funcional do Processo de Validação

3.4 Modelo de Documentos

3.4.1 Descrição Informal

Para obter o conjunto de documentos que interessa ao usuário, é necessário que os documentos possuam descrições declarativas que os identifiquem e que possam ser usadas na comparação com as requisições do usuário. Estas descrições são comumente denominadas metadados. Os metadados são armazenados com o conteúdo para auxiliar na consulta ao banco de dados. A exatidão desse processo de consulta depende da técnica utilizada para a geração do metadado.

O modelo de documento define a estrutura de um documento, incluindo os metadados. Cada metadado deve refletir uma característica do conteúdo. Seu domínio deve ser definido como um espaço métrico (contendo a especificação de operadores de comparação e métricas), sendo denominado, portanto, como um *eixo de valoração* do documento.

Informalmente, no contexto do *MyNews*, os documentos são estruturados da seguinte forma: título, subtítulo, autores, resumo, imagem e texto. Os eixos de valoração definidos são:

- o *eixo semântico* é representado por um grafo dirigido com nós e arcos rotulados, onde o rótulo de um nó representa um tema no domínio da aplicação e o rótulo de um arco representa a importância da ligação entre os dois nós. A distância entre dois nós A e B é definida como a soma do número de arcos do menor caminho entre A e B, ponderada pelos rótulos dos arcos. Este eixo indica a relação entre os conceitos do domínio da aplicação. A Figura 8 apresenta um exemplo de grafo semântico;

- o *eixo de importância* é representado pelo intervalo [1,10] e indica a importância de um documento, onde o valor 1 significa a maior importância;

- o *eixo* temporal é representado por uma tripla (a, m, d) indicando a data de criação da notícia.

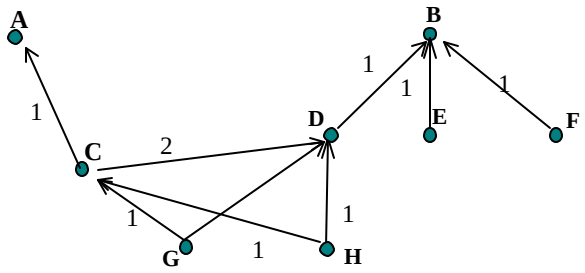


Figura 8: Parte do eixo semântico com os temas de um jornal eletrônico.

3.4.2 Especificação

Seja M o modelo de documentos com eixos de valoração $V_1,\ldots,V_n$.

Este modelo é parcialmente especificado como um *framework*, contendo os seguintes *hotspots*:

- item: representa a estrutura dos documentos. Por exemplo, para o *MyNews*, a classe notícia define os atributos: título, subtítulo, autores, resumo, imagem e texto.
- eixos de valoração: correspondem diretamente aos espaços $V_1,\ldots,V_n$. Por exemplo, para o *MyNews*, eixo semântico (ontologia), eixo de importância, eixo temporal (atualidade).
- métricas de cada eixo: definem como associar valor ao conteúdo, baseado-se nos eixos de valoração. Por exemplo, no eixo conceitual, dado pela ontologia, a relação entre duas notícias pode ser dada pela distância (número de arcos) entre elas na ontologia

A Figura 9 exibe o diagrama de classes do *framework*, ilustrando uma possível instanciação para a aplicação *MyNews*. As classes correspondentes ao modelo de documentos são aquelas delimitadas pela linha pontilhada.

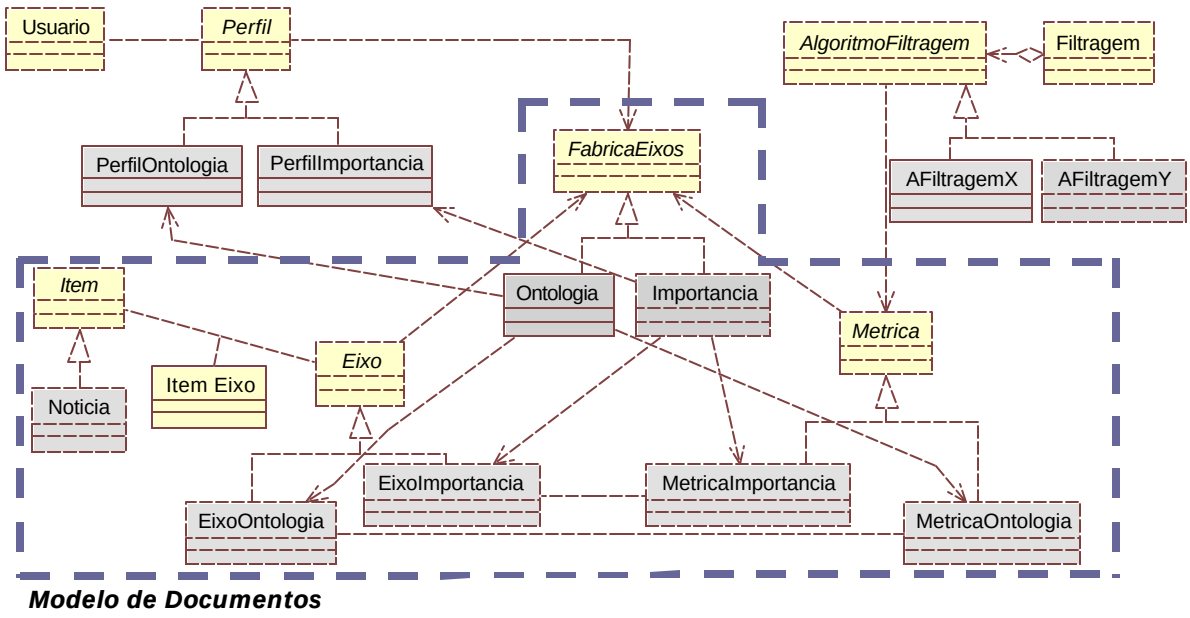


Figura 9: Modelo de classes para o modelo de documentos.

O documento *MetaModeloDocumento.xsd* captura a definição dessas classes da seguinte forma:

- O documento XML deve conter o elemento <item> como tag raiz. Este elemento contém o atributo "nome" e dois elementos: uma lista de atributos e uma lista de eixos de valoração. Esta definição é apresentada na Figura 10.

  O atributo "nome" deve conter o nome da classe resultante da instanciação do *hotSpot* Item do *framework* de filtragem.

```
<xs:element name="item" type="itemType" />
<xs:complexType name="itemType">
 <xs:sequence>
   <xs:element name="atributoItem" type="atributoItemType"
    minOccurs="1" maxOccurs="unbounded" />
   <xs:element name="eixoValoracao" type="eixoValoracaoType"
    minOccurs="1" maxOccurs="unbounded" />
 </xs:sequence>
 <xs:attribute name="nome" type="xs:string" use="required"/>
</xs:complexType>
```

Figura 10: Definição da tag raiz do modelo de documento e sua composição

- O elemento <atributoItem> possui os atributos “nome”, “tipo”, “obrigatório” e “multivalorado”. Cada elemento <atributoItem> corresponde a um atributo da classe que instancia o *hotspot* Item.

```
<xs:complexType name="atributoItemType">
  <xs:attribute name="nome" type="xs:string" use="required"/>
  <xs:attribute name="tipo" type="tipoType" use="required"/>
  <xs:attribute name="obrigatorio" type="obrigatorioType"
   use="required"/>
  <xs:attribute name="multivalorado" type="multivaloradoType"
   use="required"/>
</xs:complexType>
```

Figura 11: Especificação da lista de atributos do item

- O elemento <eixoValoracao> é composto pelos atributos “nome”, “domínio” e “métrica”. O domínio do eixo pode ser “enumerável” ou “infinito”. O conteúdo dos atributos “nome” e “métrica” deve conter os nomes das classes que instanciam os *hotspots* Eixo e Métrica, respectivamente

```
<xs:complexType name="eixoValoracaoType">
  <xs:attribute name="nome" type="xs:string" use="required"/>
  <xs:attribute name="dominio" type="dominioType" use="required"/>
  <xs:attribute name="metrica" type="xsd:string" use="required"/>
</xs:complexType>

<xs:simpleType name="dominioType">
  <xs:restriction base="xs:NMTOKEN">
   <xs:enumeration value="enumeravel"/>
   <xs:enumeration value="infinito"/>
  </xs:restriction>
</xs:simpleType>
```

Figura 12: Especificação da estrutura de um eixo de valoração

A especificação completa do documento *metaModeloDocumento.xsd* se encontra no Apêndice I. Este documento formaliza o processo de definição do modelo de documento e será utilizado para validação no momento da instanciação do modelo de documento do *framework de personalização*. Apenas os modelos que seguirem esse esquema serão aceitos e tratados pelo *framework*.

3.4.3 Exemplo de Instanciação do Modelo de Documentos

Considerando a possível instanciação do *framework* do modelo de documento, ilustrado na Figura 9, e o esquema detalhado na seção **Error! Reference source not found.**, a

estrutura do modelo de documentos do *MyNews* é representada pelo documento XML *Modelodocumento.xml* (Figura 13).

Este documento será utilizado pelo processo de empacotamento, discutido na seção 3.8, e deve ser necessariamente definido pelo instanciador do *framework*.

ModeloDocumento.xml

```
<item nome= “Noticia”>

 <atributoItem nome =”id” tipo=”number” obrigatorio=”sim”
  multivalorado=”nao”/>
 <atributoItem nome =”titulo” tipo=”text” obrigatorio=”sim”
  multivalorado=”nao”/>
 <atributoItem nome =”subtitulo” tipo=”text” obrigatorio=”sim”
  multivalorado=”nao”/>
 <atributoItem nome =”resumo” tipo=”text” obrigatorio=”sim”
  multivalorado=”nao”/>
 <atributoItem nome =”autor” tipo=”text” obrigatorio=”nao”
  multivalorado=”sim”/>
 <atributoItem nome =”texto” tipo=”text” obrigatorio=”sim”
  multivalorado=”nao”/>
 <atributoItem nome =”imagem” tipo=”text” obrigatorio=”nao”
  multivalorado=”nao”/>

 <eixoValoracao nome=”EixoOntologia” dominio=”Enumeravel”
  metrica=”MetricaOntologia”/>
 <eixoValoracao nome=”EixoImportancia” dominio=”Infinito”
  metrica=”MetricaImportancia”/>

</item>
```

Figura 13: Parte do Modelo de Documentos do MyNews

A definição completa do modelo de documento do *MyNews* (*modeloDocumento.xml*) encontra-se no Apêndice II.

É importante ressaltar que a partir dos documentos *metaModeloDocumento.xsd* e *modeloDocumento.xml* é possível gerar automaticamente um documento XML Schema, *modeloDocumento*.xsd, que define o esquema do item instanciado para a aplicação. Para o *MyNews*, por exemplo, este documento define o esquema de uma notícia. Dessa forma, os subsistemas auxiliares responsáveis pelo cadastro ou busca de conteúdo poderão utilizar este esquema para validar e armazenar os documentos. A definição do *modeloDocumento.xsd* se encontra no Apêndice I.

3.5 Modelo de Usuário

3.5.1 Descrição Informal

O modelo de usuário deve conter informações sobre as preferências e objetivos de um usuário na utilização do sistema. Estes dados podem ser informados diretamente pelo próprio usuário através de um questionário, ou inferidos pelo sistema a partir da monitoração de seu comportamento na aplicação.

Seja M um modelo de documentos (específico para a aplicação) com eixos de valoração $V_1,\ldots,V_n$. Um modelo de usuário U para M é um par $U=((U_1,\ldots,U_n), K)$, onde U_i define uma restrição sobre os valores do eixo de valoração V_i.

No exemplo do *MyNews* apresentado na seção anterior, onde são considerados os eixos semântico, de importância e temporal, o modelo do usuário poderia ser definido pelo par U = ((S, I, T), K), onde:

- S é um conjunto de temas de interesse do usuário, obtidos dos temas contidos no eixo semântico;
- I indica a importância mínima dos documentos que devem ser retornados;
- T é um conjunto de restrições relacionadas à obsolescência dos documentos;
- K é o limite mínimo do rank dos documentos a serem retornados (a função de rank será definida na seção 3.6.1 .

Considere, por exemplo, que o usuário esteja interessado pelo tema A e que deseje visualizar todas as notícias (qualquer importância entre 1 e 10) que foram inseridas nos últimos 3 dias, e que o rank mínimo das notícias seja 6. Este modelo poderia ser representado pelo par U = (({A}, 10, {*today* – t < 3}), 6), onde *today* denota a data corrente.

3.5.2 Especificação

O modelo de usuários é parcialmente especificado como um *framework*, contendo o seguinte *hotspot*:

- perfil do usuário: o perfil do usuário deve possibilitar a definição de restrições em relação a cada eixo definido.

A Figura 14 exibe o diagrama de classes do *framework*, ilustrando uma possível instanciação para a aplicação *MyNews*. As classes correspondentes ao modelo de usuário são aquelas delimitadas pela linha pontilhada.

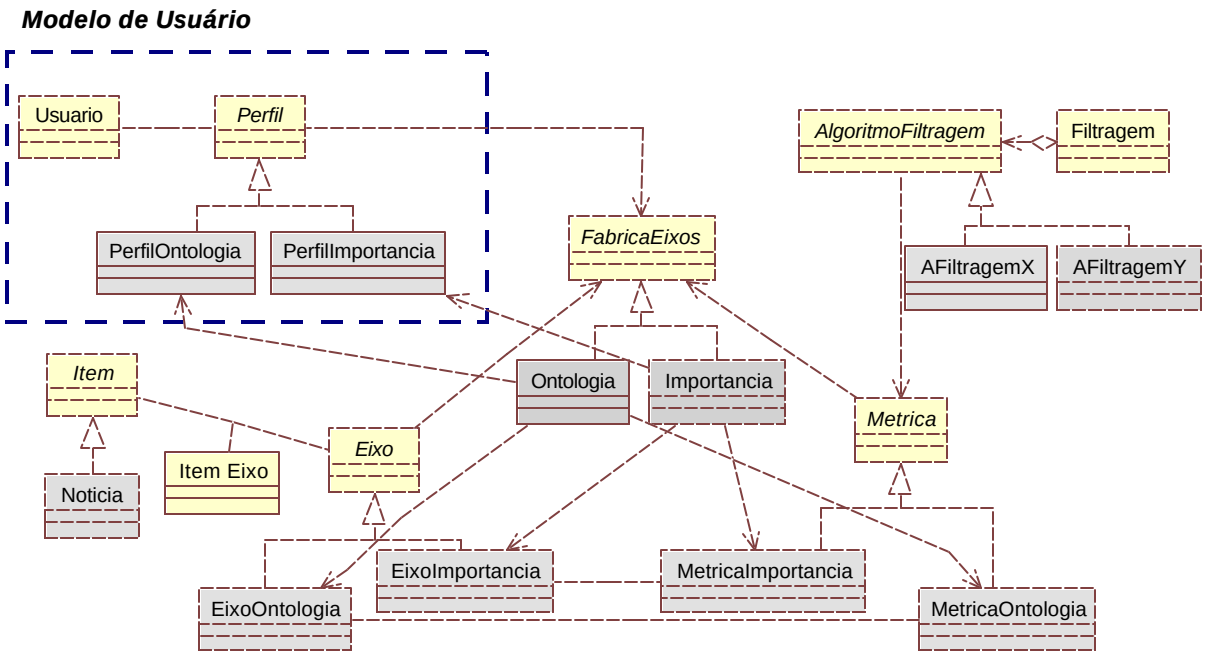


Figura 14: Modelo de classes para o modelo de usuário.

Para fins de completude da especificação do *framework*, será definido o meta-modelo de usuário, representado por um documento XML Schema, que especifica o modelo de usuário, apesar de o *framework* não utilizá-lo diretamente.

O documento *MetaModeloUsuario.xsd* captura a definição da classe Perfil da seguinte forma:

- O documento XML deve conter o elemento <perfil> como tag raiz. Este elemento contém uma lista de elementos <eixoValoracao>. Cada um destes elementos representam uma especialização da classe Perfil;

```
<xs:element name="perfil" type="perfilType" />
<xs:complexType name="perfilType">
 <xs:sequence>
   <xs:element name="eixoValoracao" type="eixoValoracaoType"
    minOccurs="1" maxOccurs="unbounded" />
 </xs:sequence>
</xs:complexType>
```

Figura 15: Definição da tag raiz do modelo de usuário e sua composição

- Cada elemento <eixoValoracao> possui o atributo “nome”, que deve conter o nome da classe que estende a classe Perfil. Este elemento deve conter o elemento <enumeravel> ou o elemento <infinito>. Estas tags representam os domínios de metadados tratados pelo *framework*, possibilitando a definição de restrições utilizando um conjunto (dominio enumerável) ou um intervalo (domínio infinito);

```
<xs:complexType name="eixoValoracaoType">
 <xs:sequence>
  <xs:choice>
   <xs:element name="enumeravel" type="enumeravelType" minOccurs="1"
    maxOccurs="1"/>
   <xs:element name="infinito" type="infinitoType" minOccurs="1"
    maxOccurs="1"/>
  </xs:choice>
 </xs:sequence>
 <xs:attribute name="nome" type="xs:string" use="required"/>
</xs:complexType>
```

Figura 16: Especificação da restrição definida no perfil do usuário.

A especificação completa do documento *MetaModeloUsuario.xsd* se encontra no Apêndice I. Este documento formaliza o processo de definição do modelo de usuário.

3.5.3 Exemplo de Instanciação do Modelo de Usuário

Considerando a instanciação do *framework* do modelo de usuário, ilustrado na Figura 14, e o esquema detalhado na seção **Error! Reference source not found.**, a estrutura do modelo de documentos do *MyNews* é representada pelo documento XML *MetaModeloUsuario.xml* (Figura 17).

ModeloUsuario.xml

```
<perfil>
 <eixoValoracao nome =”Ontologia”/>
  <enumeravel>
   <valor>A</valor>
   <valor>B</valor>
  </enumeravel>
 </eixoValoracao>

 <eixoValoracao nome =”Importancia”/>
  <infinito ini =”1” fim=”3”/>
 </eixoValoracao>

 <eixoValoracao nome =”Atualidade”/>
  <infinito ini =”15/05/2002” fim=”15/05/2002”/>
 </eixoValoracao>
</perfil>
```

Figura 17: Documento que representa o modelo de usuários do MyNews

A definição completa do modelo de usuário do *MyNews* (*modeloDocumento.xml*) encontra-se no Apêndice II.

3.6 Subsistema de Filtragem

3.6.1 Descrição Informal

O algoritmo de filtragem cria um subconjunto ordenado do conjunto de documentos disponíveis, baseando-se em:

- uma *função de rank F,* que computa o valor do *rank* de cada documento, de acordo com seus valores em cada eixo de valoração, e
- os objetivos e interesses representados no modelo de usuário.

Intuitivamente, o algoritmo de filtragem correlaciona os eixos de valoração dos documentos com o modelo do usuário, obtendo um conjunto ordenado de documentos que supostamente são de interesse do usuário.

Mais precisamente, seja M um modelo de documentos com eixos de valoração V_1, ..., V_n e U=((U_1,…,U_n), K) um modelo de usuários para M.

Uma função de rank para U é uma função F[U]: V_1 X … X V_n → R.

O algoritmo de filtragem recupera do repositório de documentos todos os documentos *x* cujos valores nos eixos de valoração satisfazem às restrições definidas em U e são tais que F[U](x) > K. O algoritmo de filtragem ordena ainda os documentos por ordem de rank.

Retornando à aplicação de exemplo *MyNews*, considere o modelo de usuário definido na seção 3.5.1 representado pelo par U = (({A}, 10, {*today* – t < 3}), 6). Considere ainda a função de ranking que combina os eixos semântico e de importância e é definida como $F[U](x) = \sqrt{(D_s[U](x))^2 + (D_i[U](x))^2}$, onde $D_s[U](x)$ corresponde à menor distância entre o tema do documento *x* e os temas de interesse do usuário, e o valor $D_i[U](x)$ representa a importância do documento *x*.

A Tabela 1 mostra um resultado possível para o processo de filtragem, onde a coluna de ranking contém o valor de relevância do documento computado pela função apresentada acima, e as outras colunas correspondem aos eixos de valoração e componentes do documento (veja também a Figura 8 para a descrição do eixo semântico).

Documento	Ranking	Tema	Importância	Obsolescência	Título
Doc2	2.24	C	1	3-Dec-2001	n2
Doc1	3.16	A	3	5-Dec-2001	n1
Doc4	3.16	A	3	3-Dec-2001	n4
Doc53	5.10	A	5	3-Dec-2001	n53
Doc18	7.21	D	6	4-Dec-2001	n18
Doc7	10.05	A	10	1-Dec-2001	n7

Tabela 1: Exemplo de um conjunto de documentos resultantes do processo de filtragem.

3.6.2 Modelagem

O subsistema de filtragem é modelado como um *framework*, contendo o seguinte *hotspot*:

- algoritmo de filtragem: se refere à forma de combinação dos eixos. Este algoritmo deve trazer o conjunto de documentos mais relevantes para o usuário.

A Figura 18 combina as figuras 8 e 13 com o hotspot do algoritmo de filtragem, ilustrando uma possível instanciação para o filtro da aplicação *MyNews*. As classes de cores mais escuras representam as instanciações dos *hotspots*. Por exemplo, as classes Ontologia e Importancia são instanciações do *hotSpot* FabricaEixos. Cada uma é responsável por instanciar o eixo, métrica e perfil correspondente. As classes AFiltragemX e AFiltragemY instanciam o *hotSpot* AlgoritmoFiltragem implementando os algoritmos de filtragem que fazem a combinação das métricas relacionadas a cada eixo.

O subsistema de filtragem é representado pelas classes Filtragem e AlgoritmoFiltragem.

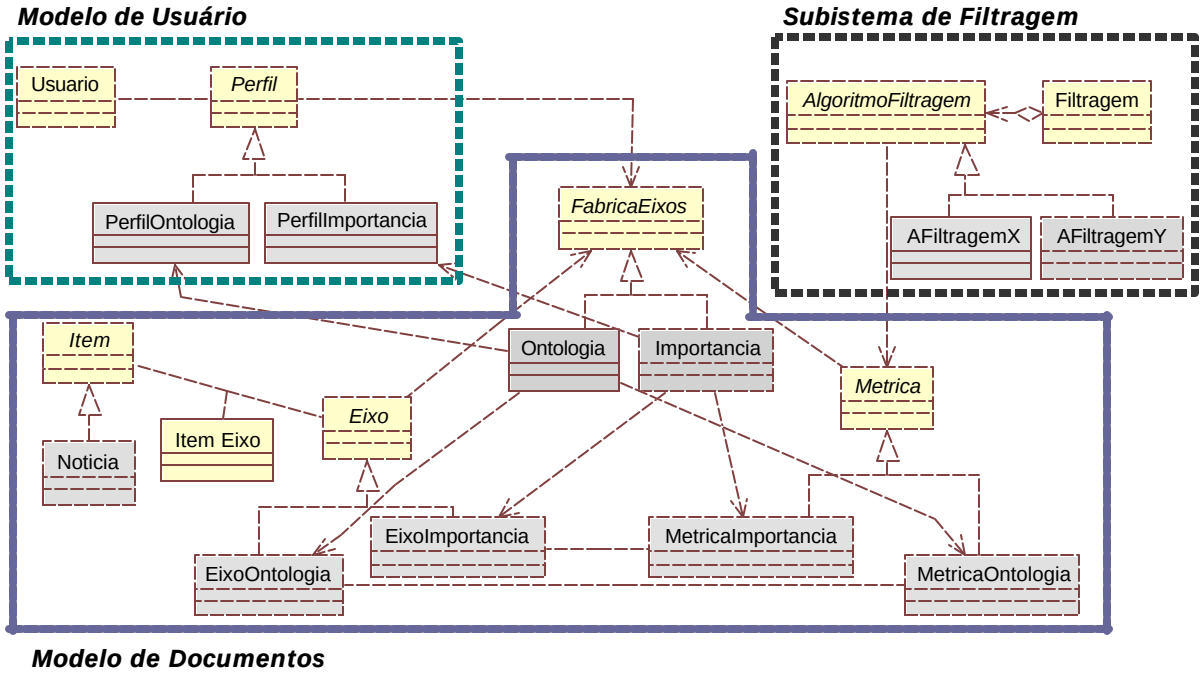


Figura 18: Modelo de classes do *framework* de filtragem de documentos

Os *design patterns* [Pree95] utilizados para definição do *framework* de filtragem foram o *Abstract Factory*, *Strategy* e *Template Method*

O *pattern Abstract Factory* oferece uma interface para a criação de famílias de objetos relacionados sem a necessidade de especificar suas classes concretas. Este *design pattern* é utilizado no *hotspot* FabricaEixos, como ilustrado na Figura 19. Assim, sempre que um novo eixo de valoração for definido, as classes Eixo, Métrica e Perfil serão necessariamente instanciadas. A métrica define como comparar os documentos tendo como base o novo eixo. O perfil irá possibilitar que o usuário indique suas preferências definindo restrições quanto ao seu domínio.

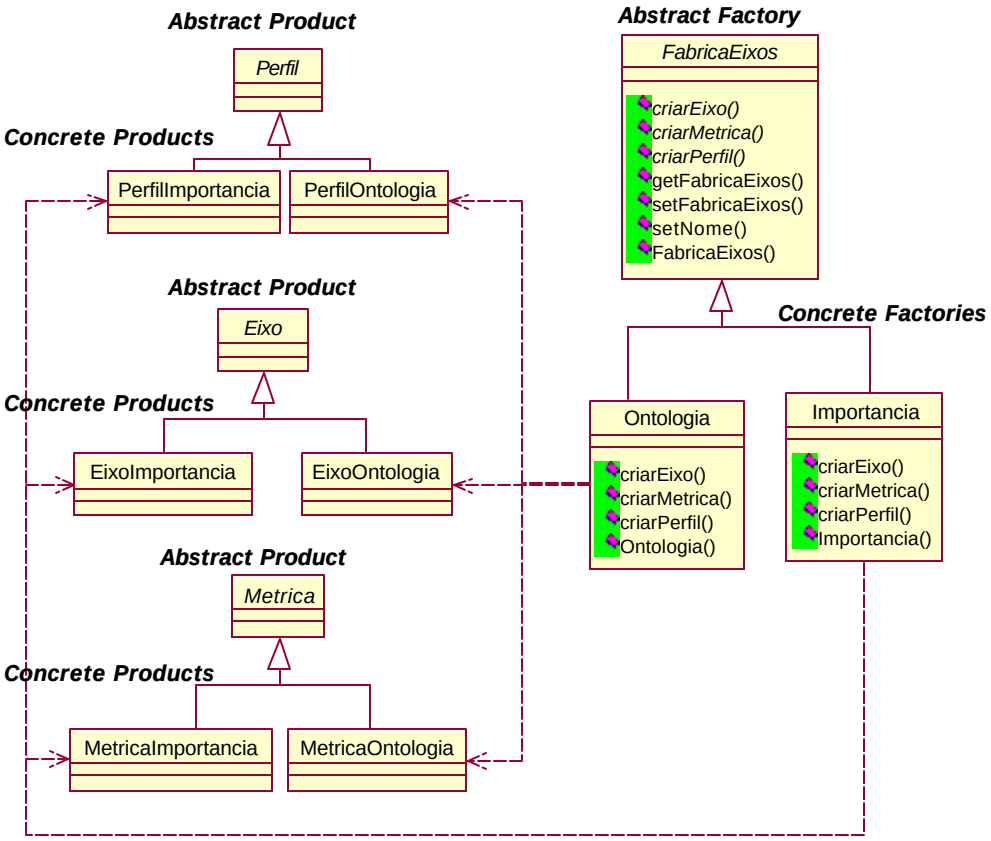


Figura 19: Estrutura do Design Pattern Abstract Factory

O *pattern Strategy* é utilizado no *hotspot* AlgoritmoFiltragem. O objetivo foi permitir a inclusão de novos algoritmos para seleção de conteúdo. A classe Filtragem, dessa forma, pode definir uma família de algoritmos e permitir sua variação de forma independente.

O *pattern Template Method* foi utilizado para facilitar a inclusão de um novo algoritmo de filtragem, generalizando o processo de execução das métricas.

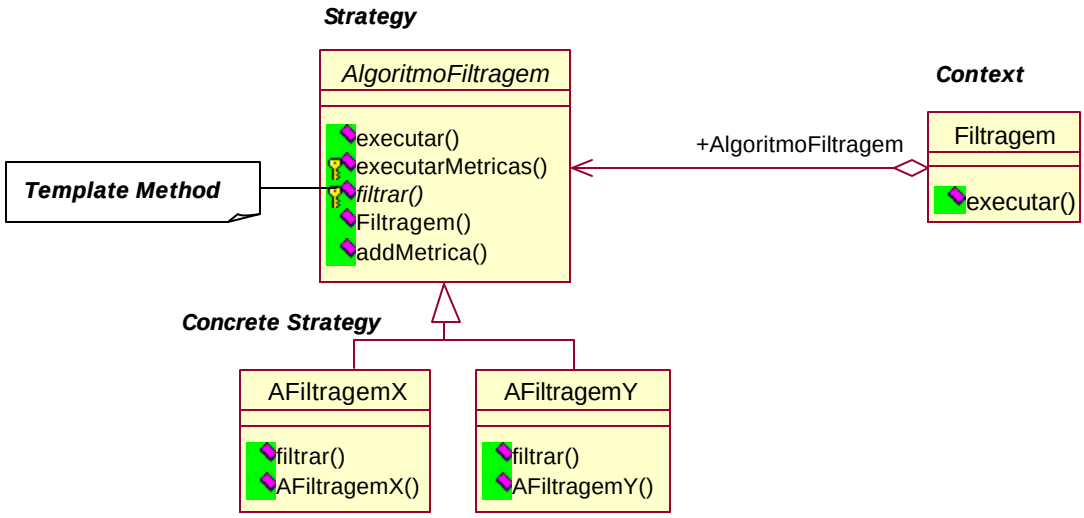


Figura 20: Estrutura do Design Pattern Strategy

A abordagem de filtragem proposta torna o domínio da aplicação flexível. É possível lidar tanto com notícias quanto com produtos (como a Amazon.com) redefinindo apenas a ontologia de domínio e algumas métricas específicas.

Os diagramas de seqüência do *framework* de filtragem se encontram no Apêndice III.

3.6.3 Fluxo de Execução

Quando um algoritmo de filtragem, responsável pela geração do conjunto de documentos de interesse do usuário (*Documentos.xml*), é acionado, os seguintes passos são executados:

- O método *executar* do algoritmo de filtragem recebe, como parâmetro de entrada, o usuário que está acessando a aplicação. Este método executa as métricas referentes a cada eixo de valoração (*executarMetricas*), no caso do *MyNews* MetricaOntologia, MetricaAtualidade, MetricaImportancia, calculando o valor de cada documento nos respectivos eixos, e então aciona o método *filtrar*;

- O método *filtrar* recebe, como entrada, os valores das notícias em cada eixo; realiza a combinação dos mesmos, de acordo com a função de rank; e então, retorna um conjunto de notícias com os respectivos valores de relevância (*Documentos.xml*).

O diagrama de seqüência que detalha o fluxo de execução do Subsistema de Filtragem se encontra no Apêndice III.

3.7 Modelo de Empacotamento

3.7.1 Descrição Informal

O modelo de empacotamento contém a definição os padrões que irão organizar o resultado da filtragem. Cada padrão, por sua vez, define os seguintes elementos:

- os grupos nos quais o conjunto de documentos filtrados será particionado;
- a ordem dos grupos e a ordem dos documentos em cada grupo;
- as visões de documentos, que definem os componentes que serão utilizados e passados para o processo de formatação, além das referências ou *hyperlinks* para outras visões de documentos.

A Figura 21 ilustra pictoricamente o modelo de empacotamento definido para o *MyNews*.

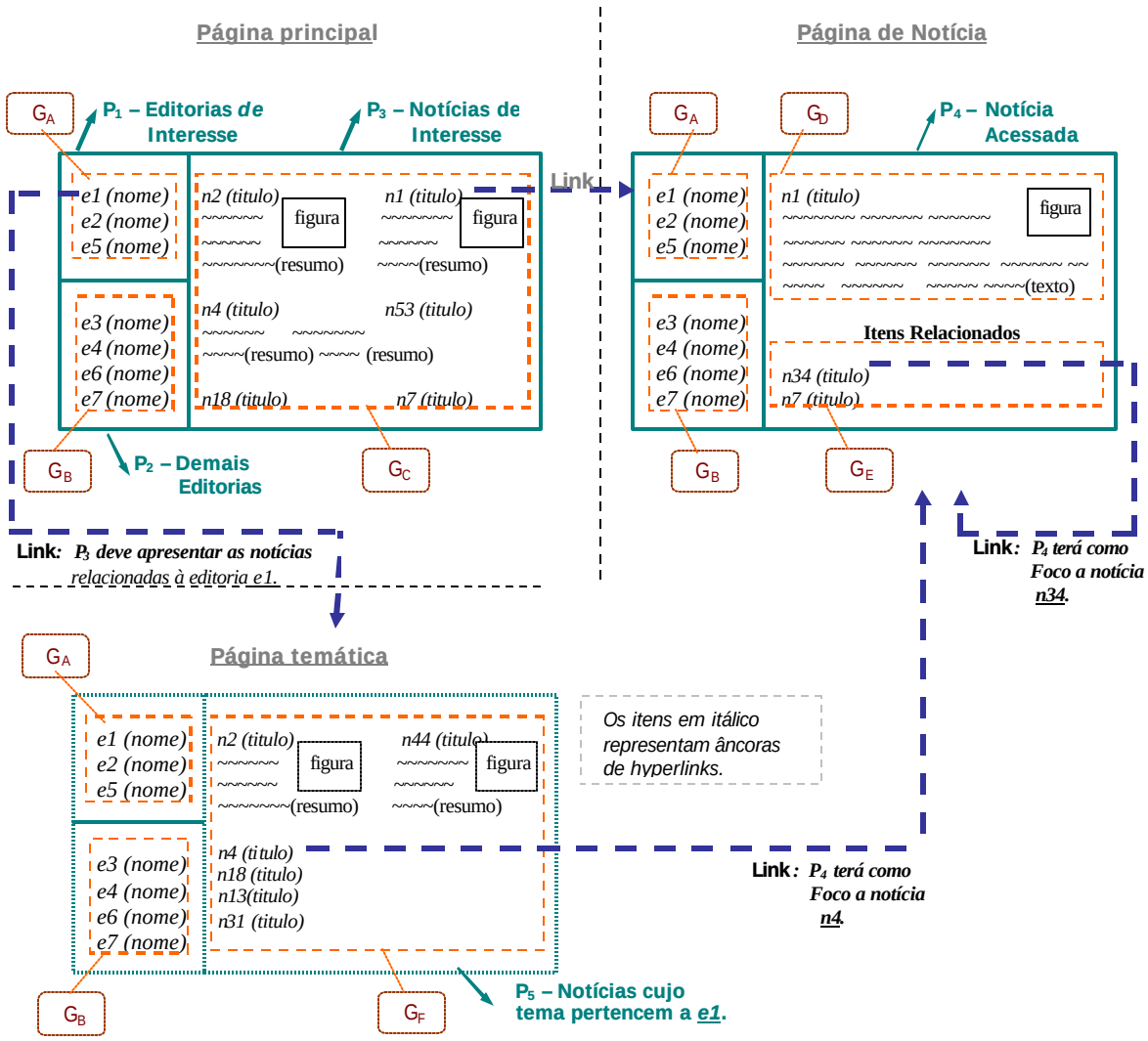


Figura 21: Modelo de Empacotamento do *MyNews*

Mais detalhadamente, o modelo de empacotamento do *MyNews* é definido da seguinte forma:

- As três estruturas de nós apresentadas na Figura 21 representam os contextos nos quais as notícias poderão ser apresentadas. Cada uma dessas estruturas deve conter um ou mais padrões que, por sua vez, devem definir a existência ou não de *hyperlinks*, delineando a navegação entre elas. As estruturas definidas para o *MyNews* são:

- *página principal*: contém dois conjuntos de índices de acesso - "Editorias de Interesse" (P_1) e "Demais Editorias" (P_2) - e um conjunto de notícias consideradas relevantes para usuário, que devem ser apresentadas seguindo a ordem do ranking obtido pelo processo de filtragem (P_3).

  O *hyperlink* referente a um índice de acesso, a editoria *e1* (em P_1), por exemplo, aponta para uma *página temática.*

  Já os *hyperlinks* referentes às notícias apontam para a própria notícia no contexto de *página de notícia*.

- *página temática*: contém uma estrutura semelhante à estrutura da *página principal*. Contudo, as notícias apresentadas em P_5 são referentes apenas aos temas relacionados à editoria *e1*.

- *página de notícia*: detalha uma notícia foco (G_D) e apresenta um conjunto contendo os títulos das notícias relacionadas à notícia foco (G_E). Cada item desse conjunto representa um *hyperlink* para a própria notícia no contexto de *página de notícia*. Esta estrutura é representada pelo conjunto { P_1, P_2 e P_4}.

  Para o *MyNews*, uma notícia é considerada diretamente relacionada a uma notícia foco se o seu tema for próximo (distância ≤ 1) ao tema da notícia foco na ontologia.

- P_1 contém a lista de editorias de interesse do usuário (representada pelo grupo G_A);
- P_2 contém uma lista com as editorias restantes (representada pelo grupo G_B);
- P_3 contém o grupo G_C que é formado por um conjunto de notícias em diferentes níveis de detalhe. Este conjunto é gerado a partir do ranking e deve incluir as notícias cujo valor do ranking é menor que 6 (veja seção 3.6 para a definição do ranking);
- P_4 apresenta o detalhamento de uma notícia foco (grupo unitário G_D) e uma lista com os títulos das notícias diretamente relacionadas a ela, a partir de uma nova filtragem (grupo G_E);

- P_5 contém o grupo G_F que é formado por um conjunto de notícias em diferentes níveis de detalhe. G_F é gerado a partir de uma editoria. Ele apresenta as notícias cujos temas correspondem a uma das editorias relacionadas nos grupos G_A ou G_B.

- Cada notícia apresentada em P_3, P_4 ou P_5 contém um *hyperlink* para o detalhamento da própria notícia (contexto de *página de notícia*).

Observa-se que os grupos podem ser compostos tanto por documentos, como em G_C, G_D, G_E e G_F, quanto por metadados, como em G_A e G_B, representando um índice para um novo grupo de documentos.

3.7.2 Especificação

A forma de especificação do modelo de empacotamento é descrito pelo documento XML Schema *MetaModeloEmpacotamento.xsd*. O meta-modelo de empacotamento definido pelo *framework* não considera relacionamentos espaciais ou temporais.

Como ilustrado na Figura 21, cada página é formada por um conjunto de padrões que, por sua vez, são subdivididos em grupos. O documento *MetaModeloEmpacotamento.xsd* formaliza a definição dessas estruturas da seguinte forma:

- O documento XML especificado pelo instanciador deverá ter <modeloEmpacotamento> como tag raiz, sob a qual serão definidos os padrões de apresentação da aplicação (<padrao>). Estas definições são apresentadas na Figura 22.

```
<xs:element name="modeloEmpacotamento"type="modeloEmpacotamentoType">
</xs:element>
<xs:complexType name="modeloEmpacotamentoType">
 <xs:sequence>
  <xs:element name="padrao" type="padraoType" minOccurs="1"
   maxOccurs="unbounded" />
  </xs:sequence>
</xs:complexType>
```

Figura 22: Definição da tag raiz e sua composição.

- Cada padrão deverá conter pelo menos um grupo. Na especificação esta condição é definida pelo identificador minOccurs do elemento "*grupo*". Além dos grupos, como apresentado na Figura 23, cada padrão deve conter um nome (atributo "*nomePadrao*").

```
<xs:complexType name="padraoType">
 <xs:sequence>
  <xs:element name="grupo" type="grupoType" minOccurs="1"
   maxOccurs="unbounded" />
 </xs:sequence>
 <xs:attribute name="nomePadrao" type="xs:string" use="required"/>
</xs:complexType>
```

Figura 23: Especificação da estrutura de um padrão.

- A definição dos grupos, como especificado na Figura 24, deve conter:
  - a natureza das informações a serem apresentadas (lista de editorias ou conjunto de documentos): representado pelo grupo “fonteConteúdo”;
  - um algoritmo de ordenação opcional: <ordenação>;
  - a especificação do *hyperlink*: <hyperlink>;
  - dois atributos indicando o nome e o título do grupo;

```
<xs:complexType name="grupoType">
  <xs:sequence>
   <xs:group ref="fonteConteudo"/>
   <xs:element name="ordenacao" type="ordenacaoType" minOccurs="0"
    maxOccurs="1" />
   <xs:element name="hyperlink" type="hyperlinkType" minOccurs="0"
    maxOccurs="1" />
  </xs:sequence>
  <xs:attribute name="nomeGrupo" type="xs:string" use="required"/>
  <xs:attribute name="tituloGrupo" type="xs:string" use="required"/>
 </xs:complexType>
```

Figura 24: Especificação da estrutura de um grupo.

Observa-se, pela especificação da Figura 24, que o escopo do *hyperlink* abrange todos os documentos pertencentes ao grupo em questão. Esta alternativa foi adotada para simplificação do problema.

- O conteúdo de um grupo, como discutido anteriormente, pode ser gerado de duas formas:
  - a partir do algoritmo especificado na tag <ranking>, apresentando um conjunto de notícias. Nesta alternativa, deve-se definir o nível de detalhe dos documentos (<visao>) e um documento foco opcional (<foco>). A execução do algoritmo deve

resultar em um conjunto ordenado de documentos XML que seguem o XML Schema modeloDocumento.xsd;

O elemento <foco> corresponde à variável, a partir da qual, todo o conjunto de documentos retornado pela filtragem é analisado e reestruturado.

Quando a tag <ranking> possui o atributo "param", o algoritmo de filtragem é executado considerando um novo modelo de usuário induzido pelo documento foco, indicado no atributo "paramRanking".

- a partir de um metadado especificado pela composição da tag <modelo>. O elemento "modelo" é composto por outras tags que indicam o atributo que deve ser apresentado como *hyperlink* ("atributo") e o limite de índices ("limite").

  Este elemento também deve conter o eixo de valoração e o tipo de procura, que indica se os atributos do eixo devem corresponder àqueles contidos no modelo do usuário (IN) ou não (OUT).

```
<xs:group name="fonteConteudo">
  <xs:choice>
   <xs:sequence>
    <xs:element name="ranking" type="rankingType" minOccurs="0"
     maxOccurs="1" />
    <xs:element name="visao" type="visaoType" minOccurs="0"
     maxOccurs="unbounded" />
    <xs:element name="foco" type="focoType" minOccurs="0"
     maxOccurs="1" />
   </xs:sequence>
   <xs:element name="modelo" type="modeloType" />
  </xs:choice>
</xs:group>


<!-- o elemento modelo eh utilizado quando o conteudo do grupo eh
metadado -->
<xs:complexType name="modeloType">
  <xs:sequence>
   <xs:element name="atributo" type="xs:string" minOccurs="1"
    maxOccurs="unbounded" />
   <xs:element name="limite" type="xs:string" />
  </xs:sequence>
  <xs:attribute name="nomeModelo" type="xs:string" use="required"/>
  <xs:attribute name="metadado" type="xs:string" use="required"/>
  <xs:attribute name="tipoProcura" type="tipoProcuraType"
   use="required"/>
</xs:complexType>
```

Figura 25: Especificação da composição do grupo "fonteConteúdo"

A Figura 26 ilustra o documento XML correspondente às especificações dos elementos <ranking> (em P_3) e <modelo> (em P_1):

```
<padrao nomePadrao="P3">
  <grupo nomeGrupo="Gc" tituloGrupo="Notícias de Interesse">
   <ranking paramRanking="ObterDocFoco">AFiltragemX</ranking>
   <visao nome="dv3">
    ...
   </visao>
   ....
   </grupo>
</padrão>

<padrao nomePadrao="P1">
  <grupo nomeGrupo="Ga" tituloGrupo="Editorias de Interesse">
   <modelo nomeModelo = "ModeloUsuario" metadado="Ontologia"
    tipoProcura="IN">
    <atributo> Tema </atributo>
    <limite> 10 </limite>
   </modelo>
   ...
  </grupo>
 </padrao>
```

Figura 26: Exemplo da definição da fonte de conteúdo

- A visão de um documento é representada pelo elemento <visão> e deve conter o atributo “nome”, que irá referenciar uma visão definida no documento *ModeloVisoes.xml*. Como especificado na Figura 27, este elemento pode ser composto por uma das seguintes tags: <numDoc>, <intervaloRank> ou <eixo>.

```
<xs:complexType name="intervaloRankType">
  <xs:attribute name="ini" type="xs:string" use="required"/>
  <xs:attribute name="fim" type="xs:string" use="required"/>
</xs:complexType>
<xs:complexType name="eixoType">
  <xs:attribute name="nome" type="xs:string" use="required"/>
  <xs:attribute name="valor" type="xs:string" use="required"/>
</xs:complexType>

<xs:complexType name="visaoType">
  <xs:choise>
    <xs:element name="numDoc" type="xs:string" minOccurs="1"
     maxOccurs="1" />
    <xs:element name="intervaloRank" type="intervaloRankType"
     minOccurs="1" maxOccurs="1" />
    <xs:element name="eixo" type="eixoType" minOccurs="1"
     maxOccurs="1" />
  </xs:choise>
  <xs:attribute name="nome" type="xs:string" use="required"/>
</xs:complexType>
```

Figura 27: Especificação de uma visão.

Dessa forma, o instanciador pode definir o conteúdo da visão indicando apenas o número de documentos, considerando a ordenação dada pelo rank; ou indicando o intervalo de rank referenciando, por exemplo, todos os documentos cujo valor de rank estejam entre 1 e 3; ou ainda indicar o eixo de valoração, onde um novo agrupamento é realizado com base neste metadado. Abaixo são apresentados alguns exemplos para definição de uma visao:

<visao nome="dv3"><numDoc>2</numDoc></visao>, ou

<visao nome="dv3"><intervaloRank ini="1" fim="3" /></visao>, ou

<visao nome="dv3"><eixo nomeEixo="Ontologia">C</eixo></visao>

- O *hyperlink* de um grupo define qual o contexto que as notícias alvo do *hyperlink* serão apresentadas, indicando os padrões destino.

  Inicialmente, como citado anteriormente, a definição de *hyperlink* é idêntica para todo o grupo de documentos do grupo em questão.

```
<xs:complexType name="hyperlinkType">
  <xs:sequence>
   <xs:element name="link" type="xs:string" minOccurs="0"
    maxOccurs="unbounded" />
  </xs:sequence>
 </xs:complexType>
```

Figura 28: Especificação do *hyperlink* de um grupo.

- Para garantir, ainda na fase de validação do modelo, que o usuário tenha indicado apenas nomes de padrões que estejam definidos no modelo, a definição apresentada na Figura 29 deve ser acrescentada.

```
<!-- Link deve ser um padrao ja definido-->
  <xs:key name="idPadrao" >
   <xs:selector xpath="padrao" />
   <xs:field xpath="@nomePadrao" />
  </xs:key>
  <xs:keyref name="refIdPadrao" refer="idPadrao" >
   <xs:selector xpath="padrao" />
   <xs:field xpath="link" />
  </xs:keyref>
```

Figura 29: Definição de chave para o atributo "nomePadrao" do elemento Padrao.

É importante ressaltar que o instanciador do modelo deve garantir a consistência entre o *ModeloEmpacotamento.xml* e os seguintes documentos: *ModeloVisoes.xml*, a tag <visões> do modelo deve referenciar uma visão definida no documento *ModeloVisoes.xml*; e

ModeloDocumento.xml, o atributo “metadado” deve referenciar um eixo de vabração previamente definido no documento *ModeloDocumento.xml*.

A especificação completa do documento *ModeloEmpacotamento.xsd* se encontra no Apêndice I. Este documento formaliza o processo de definição do modelo de empacotamento e será utilizado para validação no momento da instanciação do *framework*. Apenas os modelos que seguirem esse esquema serão aceitos e tratados pelo *framework*.

3.7.3 Exemplo de Instanciação do Modelo para o MyNews

Para a instanciação do modelo de empacotamento, as visões dos documentos devem ser previamente definidas.

Uma *visão de documento* especifica uma estrutura diferente para um documento, que pode reordenar ou omitir seus componentes, e é definido através de um documento XML, como ilustrado na Figura 30 para o *MyNews*.

ModeloVisoes.xml

```
<visoes>
 <visao nome = “dv1”>
  <atributoItem nome = “id”>
  <atributoItem nome = “titulo”>
 </visao>
 <visao nome = “dv2”>
  <atributoItem nome = “id”>
  <atributoItem nome = “titulo”>
  <atributoItem nome = “resumo”>
 </visao>
 <visao nome = “dv3”>
  <atributoItem nome = “id”>
  <atributoItem nome = “titulo”>
  <atributoItem nome = “resumo”>
  <atributoItem nome = “imagem”>
 </visao>
 <visao nome = “dv4”>
  <atributoItem nome = “id”>
  <atributoItem nome = “titulo”>
  <atributoItem nome = “resumo”>
  <atributoItem nome = “imagem”>
  <atributoItem nome = “texto”>
  <atributoItem nome = “autor”>
</visao>
</visoes>
```

Figura 30: Definição das visões de documentos para o *MyNews*.

O documento *ModeloVisoes.xml* é validado pelo documento XML Schema *MetaModeloVisoes.xsd* na fase de instanciação do *framework*.

O documento *MetaModeloVisoes.xsd*, apresentado na Figura 31, faz parte do *framework* e formaliza a definição das visões da seguinte forma:

- o documento *ModeloVisoes.xml*, definido pelo instanciador, deverá ter <visoes> como tag raiz. Sob esta tag serão definidas as visões da aplicação. Cada elemento <visao> deve conter uma lista de atributos e um nome.

- cada elemento <atributoItem> deve conter o nome de um atributo do item instanciado (Noticia, para o *MyNews*).

MetaModeloVisoes.xsd

```
<xs:complexType name="atributoItemType">
 <xs:attribute name="nome" type="xs:string" use="required"/>
</xs:complexType>
<xs:complexType name="visaoType">
 <xs:sequence>
   <xs:element name=”atributoItem” type=”atributoItemType”
    minOccurs="1" maxOccurs="unbounded" />
 </xs:sequence>
 <xs:attribute name="nome" type="xs:string" use="required"/>
</xs:complexType>
<xs:element name=”visoes” type=”visaoType” minOccurs="1"
 maxOccurs="unbounded" />
```

Figura 31: MetaModeloVisoes.xsd

Considerando o modelo de empacotamento ilustrado na Figura 21 e o meta-modelo detalhado na seção **Error! Reference source not found.**, o documento referente ao modelo de Empacotamento do *MyNews* – *modeloEmpacotamento.xml* – define o padrão P_1, por exemplo, da seguinte forma:

```
<padrao nomePadrao="P1">
  <grupo nomeGrupo="Ga" tituloGrupo="Editorias de Interesse">
   <modelo nomeModelo = "ModeloUsuario" metadado="Ontologia"
    tipoProcura="IN">
    <atributo> Tema </atributo><limite> 10 </limite>
   </modelo>
   <ordenacao> ordemAlfabetica </ordenacao>
   <hyperlink> <link>P1</link><link>P2</link><link>P5</link>
   </hyperlink>
  </grupo>
 </padrao>
```

Figura 32: Definição do padrão P_1.

Pela definição apresentada na Figura 32 , observa-se que:

- o padrão P_1 contém apenas um grupo – Editorias de Interesse:

  <grupo nomeGrupo="Ga" tituloGrupo="Editorias de Interesse">

- este grupo é composto por índices referentes a um conjunto de documentos cujos temas estão no modelo do usuário:

  <modelo nomeModelo = "ModeloUsuario" metadado="Ontologia" tipoProcura="IN">
  <atributo> Tema </atributo>

- é permitida a apresentação de no máximo dez índices:

  <limite> 10 </limite>

- as editorias são listadas em ordem alfabética:

  <ordenacao> ordemAlfabetica </ordenacao>

- os *hyperlinks* das editorias apontam para o conjunto de padrões {P1, P2, P5}. Esta estrutura corresponde à página temática.

A definição do padrão P_2 é semelhante à do P_1, porém, sua condição de procura deve ser igual a “OUT”, representando as editorias que não estão na lista de preferências do usuário.

Para exemplificar a utilização do elemento <ranking> , a Figura 33 apresenta a definição do padrão P_3 do *MyNews*.

```
<padrao nomePadrao="P3">
  <grupo nomeGrupo="Gc" tituloGrupo="Notícias de Interesse">
   <ranking> AFiltragemX </ranking>
   <visao nome="dv3">
    <numDoc>2</numDoc>
   </visao>
   <visao nome="dv2">
    <numDoc>2</numDoc>
   </visao>
   <visao nome="dv1">
    <numDoc>-1</numDoc>
   </visao>
   <hyperlink>
      <link>P1</link><link>P2</link><link>P4</link>
   </hyperlink>
  </grupo>
 </padrao>
```

Figura 33: Definição do padrão P_3

O padrão P_3 contém apenas um grupo, o G_C. Os dois primeiros documentos deste grupo retêm os componentes definidos na visão dv3 da Figura 30; os dois subseqüentes retêm os componentes da visão dv2; para todos os outros documentos do padrão P_3, a visão dv1 será utilizada. Os *hyperlinks* apontam para as notícias correspondentes no contexto *de página de notícia*, ou seja, o conjunto de padrões {P1, P2, P5}.

A definição completa do modelo de empacotamento do MyNews - *ModeloEmpacotamento.xml* - se encontra no Apêndice II.

3.8 Subsistema de Empacotamento

3.8.1 Descrição Informal

O subsistema de empacotamento irá delinear a navegação e estruturação dos documentos, independente das questões de formatação, a partir do conjunto de documentos resultante do processo de filtragem e do modelo de empacotamento. Ele engloba as tarefas de agrupamento, ordenação e reestruturação dos documentos, e é guiado pelo modelo de empacotamento.

O esquema funcional do processo de empacotamento definido pelo *framework* é ilustrado pela Figura 34.

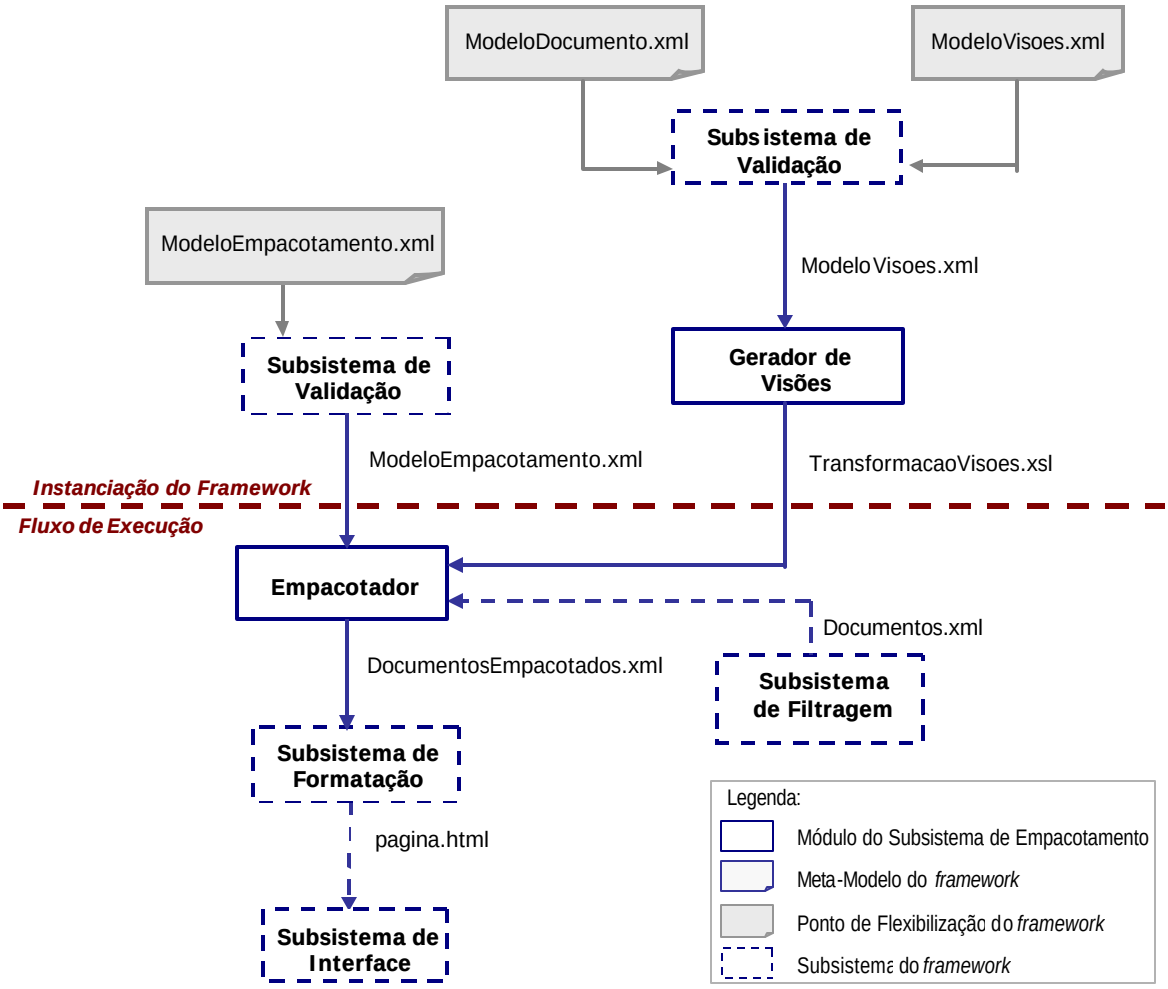


Figura 34 : Esquema Funcional do Processo de Empacotamento

Os componentes sombreados correspondem aos documentos que devem ser definidos pelo instanciador do *framework* . Os demais componentes fazem parte do *frozenspot* do *framework* .

Relembrando, o modelo de empacotamento é um template que define visões sobre o resultado da filtragem ou sobre os metadados. O Empacotador, por sua vez, é responsável pela materialização dessas visões.

Ainda na fase de instanciação, o módulo *Gerador de Visões* cria o documento *TransformacaoVisoes.xsl* a partir do documento *ModeloVisoes.xml*. O documento gerado será utilizado pelo módulo Empacotador a fim de transformar o arquivo XML resultante do processo de filtragem em um novo documento XML contendo apenas os componentes desejados [XSLT99]. O documento *TransformacaoVisoes.xsl* gerado para a aplicação *MyNews* é descrito no Apêndice II.

A função do módulo *Empacotador* é interpretar os elementos definidos pelo modelo de empacotamento e gerar um documento XML contendo o conteúdo e as diretivas de organização do mesmo. Este módulo realiza os seguintes passos:

- interpreta a requisição de empacotamento, identificando os padrões que devem ser gerados;
- para cada padrão, realiza a ordenação dos documentos dentro de cada grupo. Esta ordenação pode seguir a ordem de ranking dos documentos ou o valor obtido por uma nova função que combina o ranking com alguns atributos dos documentos. Por exemplo, os documentos podem ser apresentados na ordem cronológica reversa, indicando a atualidade dos eventos, independente de sua importância relativa. Este passo é resultante da interpretação da diretiva <ranking> do modelo de empacotamento.
- define quais componentes de cada documento serão passados para o processo de formatação. Este passo é resultante da interpretação da diretiva <visão> que utiliza a definição contida em *TransformacaoVisoes.xsl*.

Dessa forma, o processo de empacotamento resulta em um outro arquivo XML, *DocumentosEmpacotados.xml*, contendo, além das diretivas de organização, o conteúdo que deve ser apresentado.

No *MyNews*, usando o modelo de empacotamento descrito na seção 3.7, os documentos mostrados na Tabela 1 serão organizados, considerando o contexto da *página principal*, como na Figura 35.

DocumentosEmpacotados.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<documentos tipo=”noticia”>

 <padrao nomePadrao=”P3”>
  <grupo nomeGrupo=”G3” titulo=”Noticias de Interesse”>

   <item>
    <atributo nome="id"> 2 </atributo>
    <atributo nome="titulo"> n2 </atributo>
    <atributo nome="resumo"> resumo2 </atributo>
    <atributo nome="imagem"> endImagem2 </atributo>

    <eixoValoracao nome="EixoOntologia> C </eixoValoracao>
    <eixoValoracao nome="EixoImportancia> 1 </eixoValoracao>
    <eixoValoracao nome="EixoAtualidade> 03/12/2001 </eixoValoracao
   </item>

   <item>
    <atributo nome="id"> 1 </atributo>
    <atributo nome="titulo"> n1 </atributo>
    <atributo nome="resumo"> resumo1 </atributo>
    <atributo nome="imagem"> endImagem1 </atributo>

    <eixoValoracao nome="EixoOntologia> A </eixoValoracao>
    <eixoValoracao nome="EixoImportancia> 3 </eixoValoracao>
    <eixoValoracao nome="EixoAtualidade> 05/12/2001 </eixoValoracao
   </item>
      ...
   ...

   <hyperlink>
     <link>P1</link>
     <link>P2</link>
     <link>P4</link>
   </hyperlink>
  </grupo>
 </padrao>
 ...
 ...
</documentos>
```

Figura 35 : Resultado do Processo de Empacotamento

3.8.2 Modelagem

A Figura 36 exibe o diagrama de classes referentes a implementação do subsistema de empacotamento e indica a correspondência entre este subsistema e o esquema apresentado na Figura 34. Estas classes compõem o *frozenspot* do *framework.*

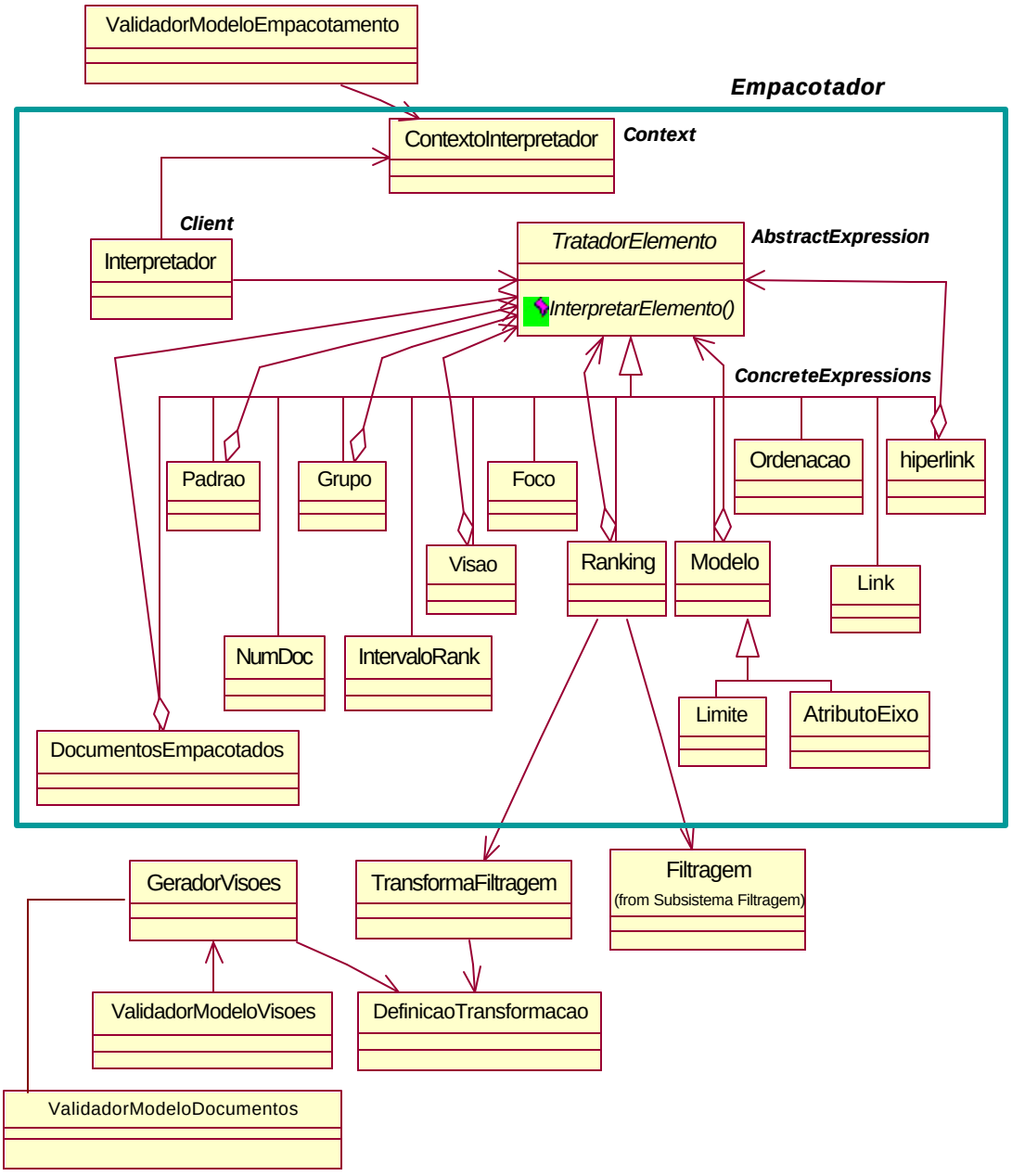


Figura 36: Diagrama de Classes do Subsistema de Empacotamento

O modelo de empacotamento, como discutido anteriormente, é representado por um documento XML e consequentemente não implica a criação de novas classes. Ao instanciador confere a indicação da localização dos documentos XML que contêm os modelos (nas classes ValidadorModeloEmpacotamento, ValidadorModeloVisoes e ValidadorModeloDocumento).

O módulo Empacotador implementa o *pattern Interpreter*. A opção pela utilização desse *design pattern* tem como objetivo preparar o *framework* para uma futura evolução. Sua utilização facilita a mudança e extensão da gramática. Como ele utiliza classes para representar as regras da gramática, pode-se utilizar a herança para mudar ou estender a gramática.

A classe TratadorElemento declara a operação abstrata *interpretar*, que é comum a todos os nós da árvore sintática. A classe ContextoInterpretador contém as informações globais do interpretador. A classe Interpretador obtém a árvore sintática representando uma instância da gramática. Esta árvore é interpretada pelas subclasses de TratadorElemento.

As classe que herdam TratadorElemento definem a semântica das tags contidas no modelo de empacotamento.

3.8.3 Fluxo de Execução

Os passos executados pelo subsistema de empacotamento para a geração do documento *DocumentosEmpacotados.xml* são:

- Primeiramente o *ModeloEmpacotamento*.xml é validado pela classe ValidadorModeloEmpacotamento, que armazena a árvore resultante da análise sintática na classe ContextoInterpretador.

- O segundo passo é realizado pela classe Interpretador, responsável pela interpretação da árvore correspondente ao modelo de empacotamento. O interpretador aciona a subclasse de TratadorElemento correspondente à tag raiz do modelo de empacotamento (classe DocumentosEmpacotados) através do método *executar*, passando o objeto Contexto como parâmetro. Esta classe inicializa a interpretação do modelo, executando as regras correspondentes e redirecionando para a classe correspondente a classe filha, caso a tag em questão não seja terminal. O estado atual do resultado da interpretação é armazenado na classe ContextoInterpretador.

O diagrama de seqüência que detalha o fluxo de execução do Subsistema de Empacotamento se encontra no Apêndice III.

4. Exemplos de Instanciação do Framework

Este capítulo apresenta duas instanciações para o *framework* proposto: *MyNews* e *MyRefs*. Estas instâncias implementam um jornal eletrônico e um repositório de referências adaptativos, respectivamente.

A instanciação do *framework*, considerando apenas os módulos discutidos no capítulo anterior, prevê duas etapas. A primeira etapa é a instanciação da parte do *framework* que realiza a filtragem dos documentos, isto é, a instanciação do *Subsistema de Filtragem.* A segunda etapa da instanciação corresponde ao *Subsistema de Empacotamento*. Estas etapas serão descritas para cada uma das instanciações.

Dentre as duas propostas de instanciação do *framework*, apenas o *MyNews* foi implementado.

Na Seção 4.1.4 será discutido o *Subsistema de Modelagem de Usuários* implementado para o *MyNews*, à guisa de ilustração, já que não faz parte do escopo desta dissertação.

4.1 MyNews

4.1.1 Motivação

A possibilidade de oferecer acesso às notícias pela Web vem atraindo uma crescente atenção nos últimos anos. Uma pesquisa conduzida pela *Content Intelligence* em Junho de 2001 [Content01] mostrou que os jornais nos EUA estão perdendo leitores para a Web. A pesquisa mostra que, quanto mais acostumada a navegar está uma pessoa, mais ela tende a substituir a leitura de jornais impressos pelas informações obtidas online.

Esta é, de fato, uma oportunidade interessante para todas as companhias que operam na Web, não só aquelas do setor de comunicação (como jornais, rádios ou companhias de TV). Muitas companhias oferecem serviços de notícias, geralmente relativos ao seu campo comercial ou aos interesses dos seus clientes. O objetivo freqüente desses serviços é, além de atrair navegadores, assegurar que acessem regularmente o site. Um dos fatores de sucesso para este tipo de serviço é permitir a personalização no acesso às notícias.

Para que os jornais eletrônicos se tornem tão difundidos como as suas versões impressas, eles devem oferecer características atrativas, além de serem meramente fáceis de ler. Eles podem fornecer um serviço personalizado permitindo que o usuário encontre facilmente a informação que o interessa.

Considerando este panorama e o fato de as aplicações desta natureza pertencerem ao domínio dos problemas solucionados pelo *framework* proposto, a aplicação instanciada, e que vem sendo discutida ao longo da dissertação, consiste em um jornal eletrônico adaptativo – o *MyNews*.

4.1.2 Instanciação do Subsistema de Filtragem

Este subsistema destina-se a selecionar as notícias do jornal considerando as informações contidas no modelo de usuários e nas próprias notícias. Dessa forma, ao instanciar este módulo, os *hotspots* do *framework* de filtragem devem ser implementados de acordo com a natureza dos metadados definidos para o conteúdo da aplicação, ou seja, para uma notícia.

4.1.2.1 Modelo de Documentos

O passo inicial para a instanciação do *framework* consiste na definição da estrutura do conteúdo. O conteúdo do *MyNews* representa a estrutura de uma notícia e contém os seguintes atributos: título, subtítulo, autores, resumo, corpo (texto), imagens (opcional). Esta estrutura é utilizada para a criação da classe Noticia, correspondendo à instanciação do *hotspot* Item.

A classe FabricaEixos, implementação do *hotspot* FabricaEixos, tem por objetivo permitir a instanciação do *hotspot* Eixo mediante a instanciação do *hotspot* Perfil e Metrica correspondente.

A implementação deste *hotspot* deve fornecer uma classe que possua o relacionamento de herança com a classe FabricaEixos. A classe proveniente da instanciação do *hotspot* deve herdar os métodos CriarEixo, CriarMetrica e CriarPerfil, e implementá-los. Durante a instanciação, de acordo com os eixos de valoração definidos na seção 3.4, as classes implementadas foram: Ontologia, Importancia e Atualidade. Cada uma dessas classes instanciam os *hotspots* Eixo, Perfil e Métrica correspondentes.

O *hotspot* Eixo é implementado, a partir do relacionamento de herança, pelas classes EixoOntologia, EixoImportancia e EixoAtualidade. Estas classes foram definidas de forma que a associação das notícias com cada eixo de valoração ocorra da seguinte forma:

- eixo semântico: neste eixo o valor da notícia é o tema em que está classificada na ontologia;
- eixo de importância: corresponde ao valor de importância da notícia dada pelo editor;
- eixo de atualidade: corresponde à data de criação da notícia.

O objetivo da classe Metrica, implementação do *hot-spot* Metrica, é computar um valor para o conteúdo em um eixo de valoração específico. A instanciação da classe Metrica deve ser feita pelo relacionamento de herança. A classe implementada deve herdar a classe Metrica e implementar o método *executar*. O método *executar* desta classe abstrata recebe como parâmetro o perfil do usuário e um vetor com parâmetros necessários para sua execução. O resultado deste método é um objeto da classe ItemValor contendo os valores das notícias considerando o eixo em questão. Na instanciação do *hotspot* Metrica foram definidas, a partir do relacionamento de herança, as classes MetricaOntologia, MetricaImportancia e MetricaAtualidade.

A métrica utilizada para valoração das notícias a partir da ontologia corresponde à forma de caminhamento no grafo, como discutido anteriormente. Para o *MyNews*, serão consideradas as notícias contidas nos temas de distância de até 2 arcos. Na navegação que é feita para termos mais genéricos, o valor de cada arco percorrido é incrementado do valor 2.

A Figura 37 ilustra a instanciação do modelo de documentos do *MyNews*.

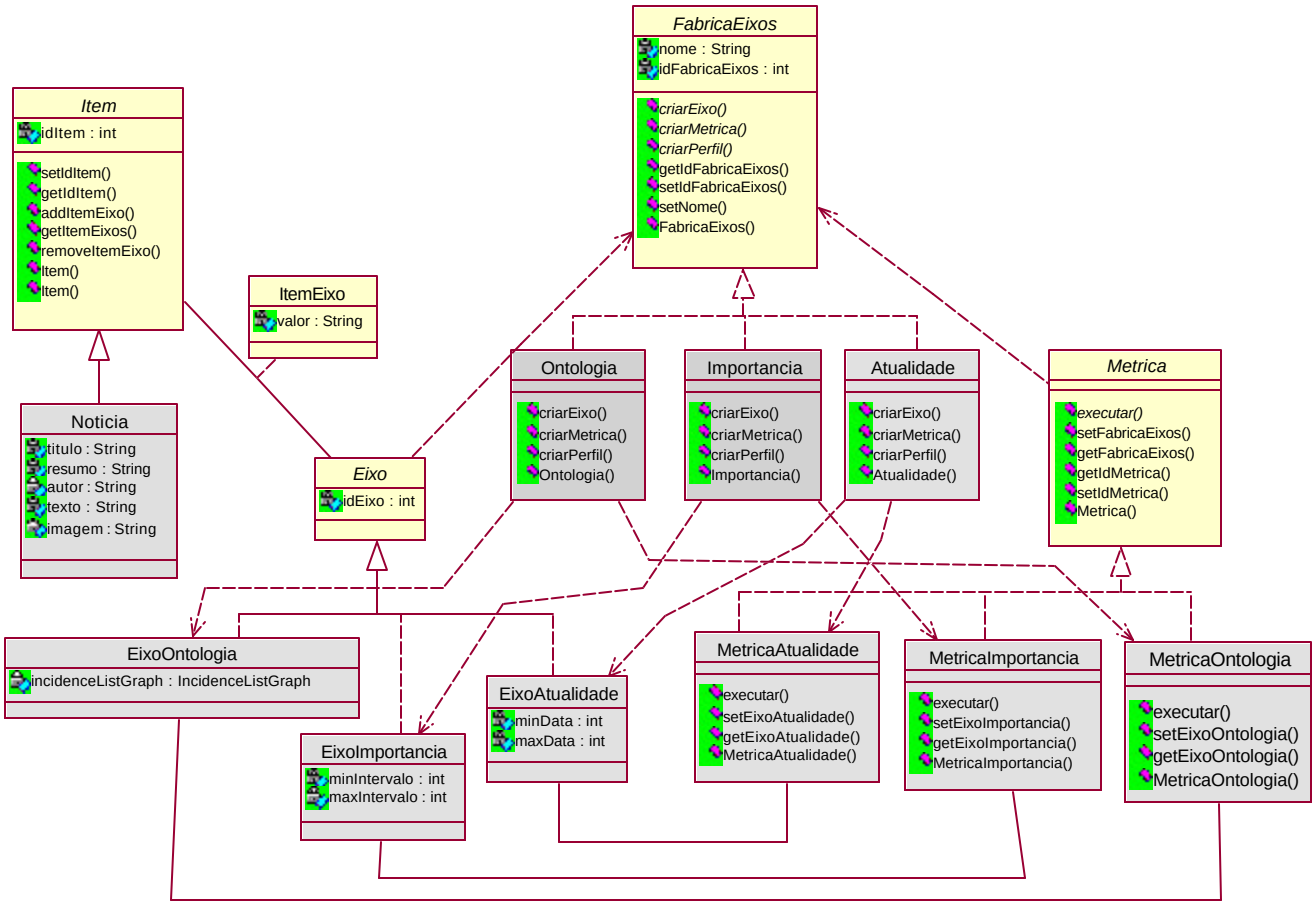


Figura 37 : Instanciação do Modelo de Documentos do MyNews

O conteúdo do *MyNews* é contextualizado pelo eixo semântico – representado por uma ontologia de domínio. Contudo, relacionar um item da ontologia com uma notícia específica pode apresentar algumas dificuldades. Na abordagem que está sendo utilizada na implementação do *MyNews*, as notícias se inter-relacionam, indiretamente, pela ligação entre os temas em que estão classificadas na ontologia. Uma desvantagem desta alternativa é o fato de os relacionamentos serem fixos, ou seja, não serem definidos no momento da criação da notícia e, desta forma, possivelmente não capturarem todos os relacionamentos entre os temas. Além disso, está sendo considerado que uma notícia deve estar classificada em apenas um tema na ontologia.

A ontologia adotada pelo *MyNews* corresponde a uma parte da ontologia definida pelo IPTC (Conselho de Telecomunicações Internacional), cujas atividades focam na definição de

padrões para o intercâmbio de notícias. O documento XML que define esta ontologia é formalizado em [IPTC00].

Além da estrutura definida pelo ITPC, cada item deve conter o atributo “nivelBase”. Este atributo indica se o tema faz parte do nível base de categorização e se refere a abrangência do tema. Um tema de nível base não deve ser muito específico nem muito genérico. Estes itens serão apresentados aos usuários no momento da criação do perfil.

A ontologia do *MyNews* ainda permite a inclusão de pesos para cada relacionamento entre temas. A utilização de rótulos nos arcos é uma justificativa para a tentativa de se associar semântica ao grafo. O grafo, além de conter os nós e suas associações, deve indicar quais itens estão mais fortemente relacionados. No caso específico do *MyNews*, esse tipo de conhecimento sobre a ontologia torna o processo de filtragem mais preciso.

Estas definições são utilizadas para a criação da classe EixoOntologia, que instancia o *hotspot* Eixo. Para a implementação da ontologia, foi utilizada uma biblioteca de grafos (JDSL 2.0).

4.1.2.2 Modelo de Usuários

O modelo do usuário do *MyNews* corresponde às instanciações do *hotspot* Perfil, como ilustrado na Figura 38. As classes que implementam este *hotspot* – PerfilOntologia, PerfilImportancia, PerfilAtualidade – devem permitir a definição de restrições quanto ao eixo correspondente, como discutido na seção 3.5.1. Estas classes devem implementar os métodos *setConteudoPerfil* e *getConteudoPerfil*.

Dessa forma o usuário pode informar seus interesses com relação:

- aos temas vinculados às notícias que deseja ler (dentre os itens da ontologia definidos como de nível base). Por questões de simplicidade, o usuário somente pode indicar seu interesse ou não por determinado tema;
- à atualidade das notícias (eixo de atualidade);
- à importância das notícias (eixo de importância).

As restrições definidas pelas implementações deste *hotspot* podem se aplicar a um dos dois domínios: infinito (restringido por um intervalo) ou enumerável (restringido por um conjunto).

A classe EixoOntologia define um domínio enumerável e as classes EixoImportancia e EixoAtualidade definem o domínio infinito.

Além de dados relacionados ao interesse do usuário, os dados de identificação do usuário, como login e senha também devem constar no modelo. Este tipo de dados é representado pela classe Usuário.

O processo de modelagem de usuário implementado para o *MyNews* é discutido na seção 4.1.4.

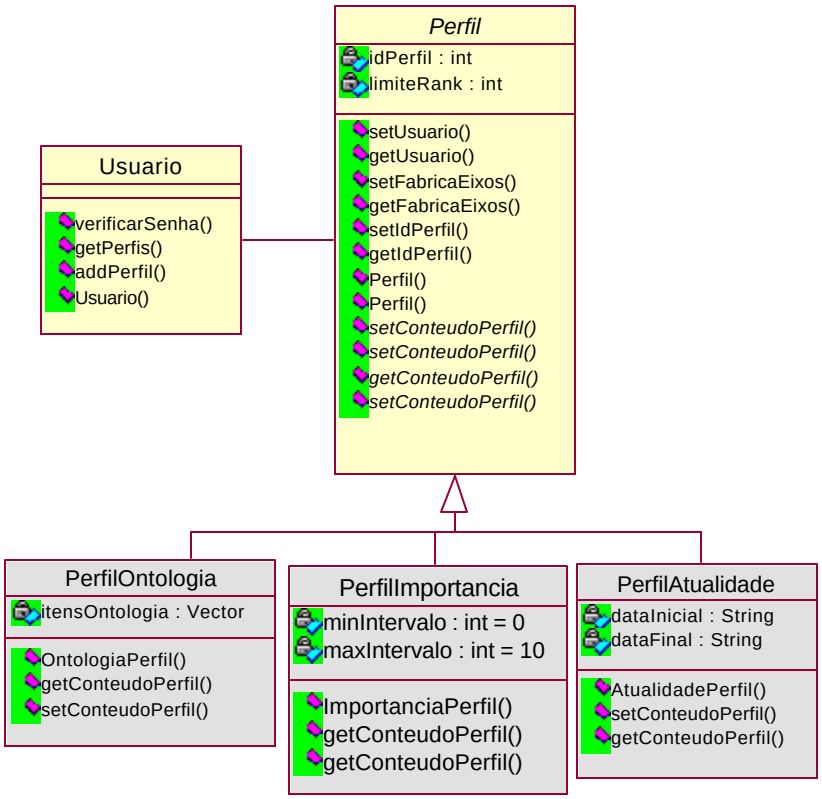


Figura 38: Instanciação do Modelo de Usuários do MyNews.

4.1.2.3 Filtragem dos Documentos

Como o próprio nome sugere, o objetivo da classe AlgoritmoFiltragem, implementação do *hotspot* AlgoritmoFiltragem, é realizar a filtragem do conteúdo da aplicação, baseando-se no modelo do usuário. O método *executar* desta classe recebe, como parâmetro de entrada, a identificação do usuário que está acessando a aplicação. Este método executa as métricas de cada eixo (*executarMetricas*) e então aciona o método *filtrar*. Este método recebe, como

entrada, os valores das notícias em cada eixo, realiza a combinação dos mesmos e retorna um conjunto de notícias com os respectivos valores de relevância. A Figura 39 ilustra esta etapa da instanciação.

A implementação deste *hotspot* para o *MyNews* gerou as classes AFiltragemX e AFiltragemY, que herdam a classe abstrata AlgoritmoFiltragem e implementam o método *filtrar*.

A classe AFiltragemX é utilizada para a geração do grupo G_c (notícias de interesse) da página principal do *MyNews*. O método *filtrar* utiliza uma função de combinação, discutida na seção 3.6.1, para gerar o ranking das notícias.

A classe AFiltragemY é utilizada para a geração do grupo G_F (notícias de interesse relacionadas a uma editoria específica) da página de notícia do *MyNews*. O método filtrar aplica a função de combinação considerando apenas as notícias da editoria em questão.

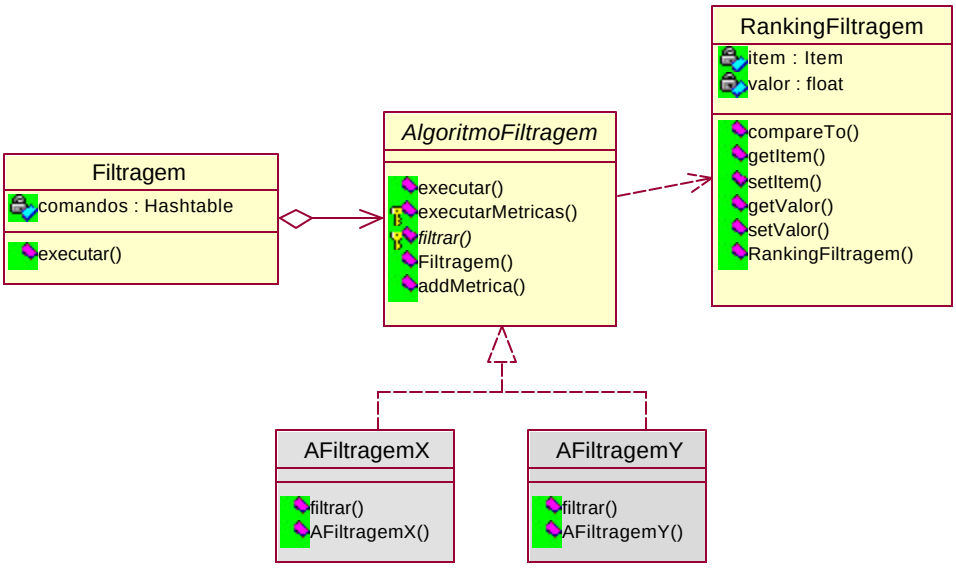


Figura 39: Instanciação do Algoritmo de Filtragem do MyNews.

4.1.2.4 Modelagem

Na Figura 40, encontra-se o diagrama de classes completo deste subsistema com as devidas implementações dos *hotspots*.

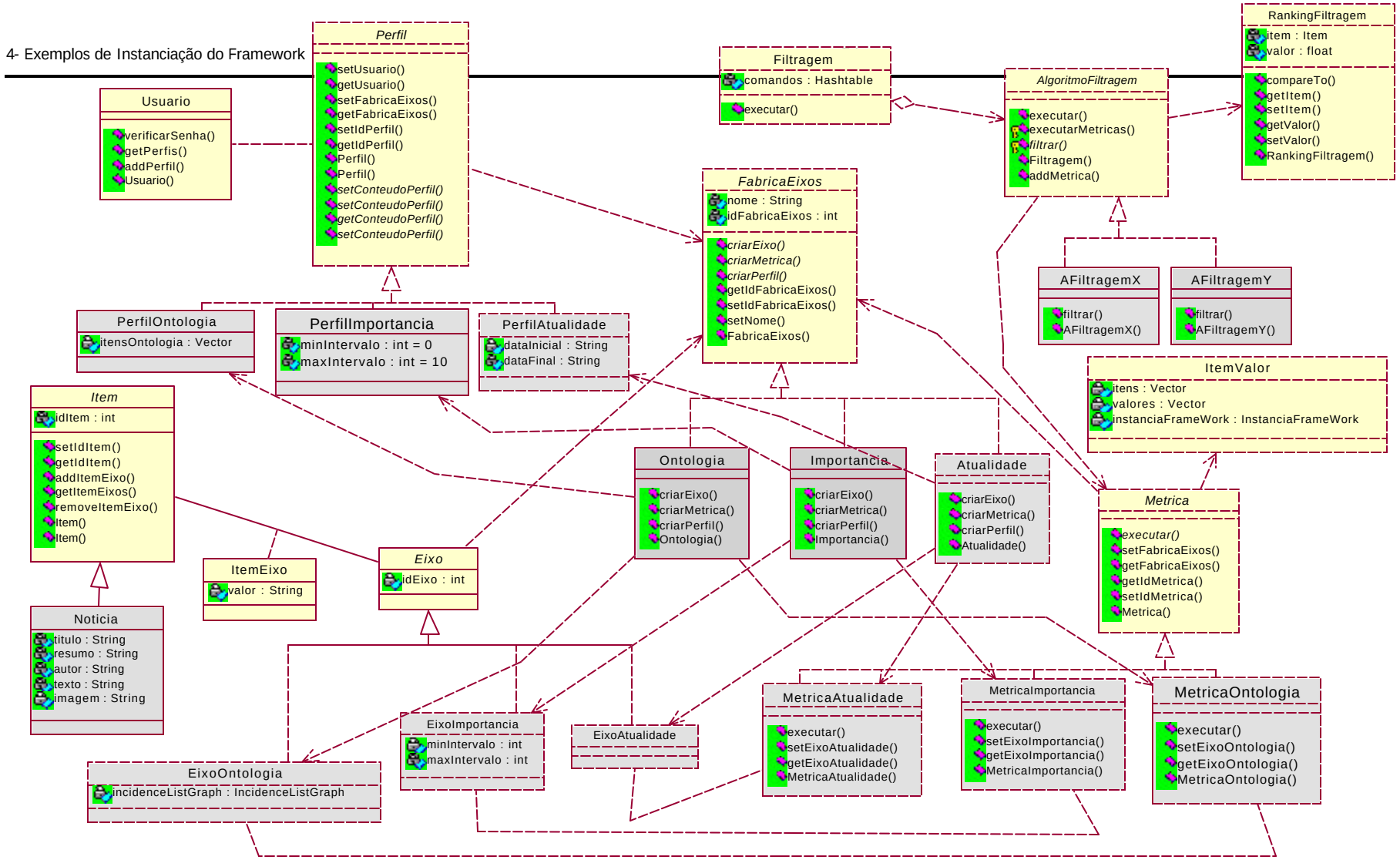


Figura 40: Diagrama de Classes do Subsistema de Filtragem do *MyNews*

4.1.3 Instanciação do Subsistema de Empacotamento

A instanciação deste subsistema é mais simples do que a do Subsistema de Filtragem, no que se refere aos *hotspots*. Os pontos de flexibilização deste subsistema devem ser implementados por documentos XML, como ilustrado na Figura 34. A validação destes documentos é realizada pelo subsistema de validação. Os documentos que foram gerados para o *MyNews* foram: *ModeloEmpacotamento.xml, ModeloVisoes.xml* e *ModeloDocumento.xml*. Estes documentos se encontram no Apêndice II.

A instanciação do subsistema de empacotamento para o *MyNews* envolveu dois passos:

- definição dos documentos XML que representam o modelo de empacotamento, de visões e de documento. Este último, representa as instanciações realizadas para os *hotspots* Item, Eixo e Métrica;
- Atualização das classes que implementam o subsistema de validação:
    - atualização da classe ValidadorModeloDocumento, indicando o caminho do arquivo ModeloDocumento.xml;
    - atualização da classe ValidadorModeloVisoes, indicando o caminho do arquivo ModeloVisoes.xml;
    - atualização da classe ValidadorModeloEmpacotamento, indicando o caminho dos arquivos ModeloEmpacotamneto.xml.

4.1.4 Subsistema de Modelagem de Usuários

4.1.4.1 Inicialização e Evolução do Modelo de Usuário

Na maior parte dos sistemas atuais, o comportamento do usuário não só é observado como também modelado como uma base direta para a personalização do sistema.

A abordagem utilizada no *MyNews* para a inicialização do modelo de usuários faz uso de questionários padrão.

Quando um novo usuário acessa o *MyNews*, um questionário, contendo uma lista de opções representando os temas base da ontologia, além das opções que possibilitam indicação de restrições quanto à importância e atualidade dos documentos, é apresentado ao usuário.

O modelo do usuário é configurado conforme o preenchimento do questionário. Contudo, se o usuário, por algum motivo, não responder ao questionário, seu perfil será automaticamente associado a um perfil padrão. Este perfil deve refletir, no caso do *MyNews*, o usuário que se interessa um pouco por cada tema.

Geralmente, agentes de interface são utilizados para, a partir da monitoração do usuário, aprenderem as correlações entre as situações que o usuário encontra e as ações que executa. Estes dados são utilizados para prever o comportamento do usuário em situações futuras, sugerir as ações apropriadas, e talvez executar automaticamente algumas ações em nome do usuário.

A abordagem utilizada para evolução do perfil no desenvolvimento do *MyNews* é baseada na análise do *log* (histórico) de navegação do usuário. O processo de análise é semelhante ao processo implementado pelo sistema ELFI, discutido na seção 2.3.2, considerando apenas a primeira forma de exploração do *log*, onde os arquivos de *log* são escaneados por seqüências de chaves que provavelmente indicam o interesse do usuário em um tópico de pesquisa específico. Para este propósito são necessários dois agentes: um agente de monitoração e um agente de modelagem.

O agente de monitoração é o encarregado de monitorar todas as ações do usuário no sistema criando um histórico de suas interações, por exemplo, guardando todos os temas que acessou e quantas vezes acessou em cada dia aquele tema.

O agente de modelagem, por sua vez, acessa o histórico de navegação do usuário e atualiza o modelo considerando o número de acessos em cada tema por dia, num período de tempo *t*. No entanto, para garantir a efetividade do algoritmo de inferência deve-se ter o cuidado na definição de quais informações levar em consideração, em que espaço de tempo e sobre qual número de interações.

4.1.4.2 Modelagem

O modelo de classes do subsistema de modelagem de usuário é apresentado na Figura 41. Os agentes utilizam um *blackboard* para trocar informações entre eles. Este subsistema, apesar de utilizar uma fachada para acesso ao modelo de usuário, ainda possui um acoplamento médio ao subsistema de filtragem.

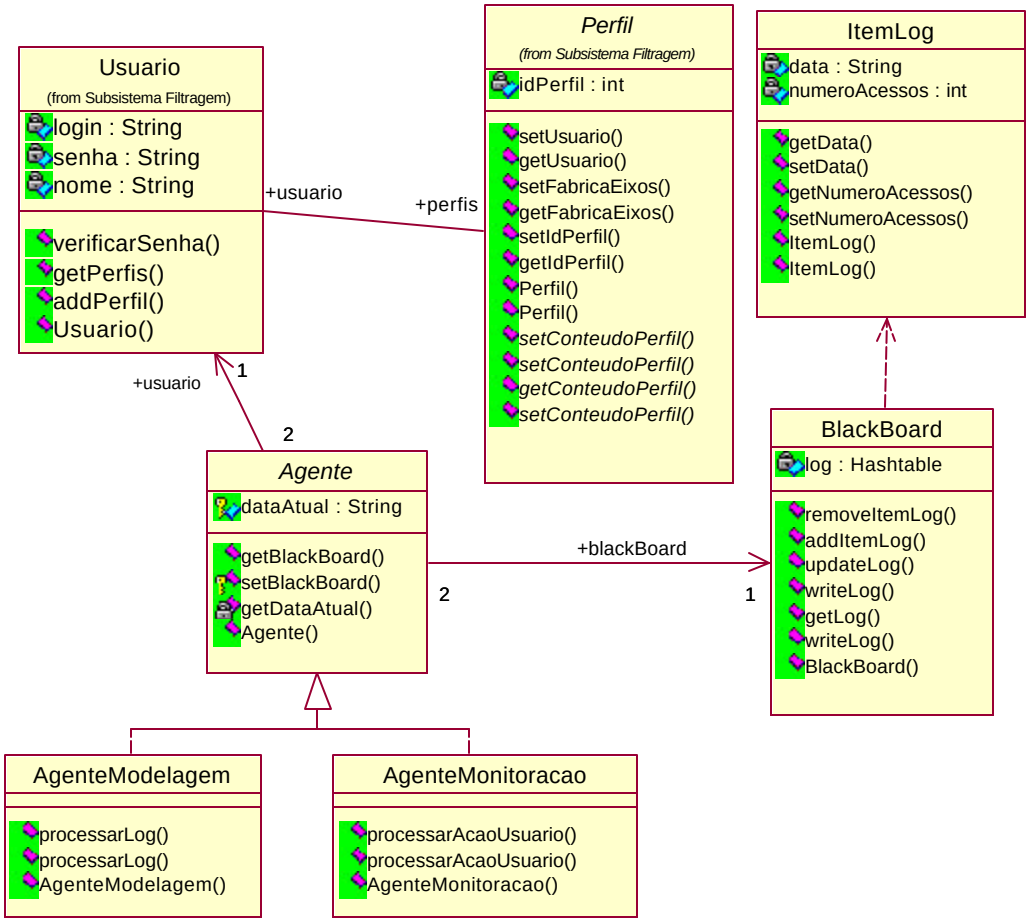


Figura 41 : Diagrama de Classes do Subsistema de Modelagem de Usuários

O agente de monitoração é inicializado pelo subsistema de interface no momento em que o usuário entra no sistema. Quando a seção do usuário é finalizada, o agente de modelagem é acionado. Este, baseando-se nos dados contidos na classe BlackBoard, pode inferir o interesse ou desinteresse do usuário em algum tema em particular, utilizando regras predefinidas. No caso do *MyNews*, o modelo do usuário será atualizado apenas no próximo acesso mediante a confirmação do usuário.

Os diagramas de seqüência que detalham a atuação dos agentes de modelagem e monitoração são apresentados no Apêndice III.

O Subsistema de Interface implementado para o *MyNews* também é apresentado no Apêndice III. A formatação dos documentos é acionada pelas classes de comando de interface e é representado pelo parser XSL.

4.2 MyRefs

4.2.1 Motivação

O volume de literatura científica disponibilizada na Web tipicamente excede a capacidade de os cientistas identificarem e utilizarem todas as informações relevantes sobre sua pesquisa.

Algumas melhorias vêm sendo realizadas no que se refere ao acesso da literatura científica, permitindo a localização de documentos com maior facilidade a partir de bibliotecas eletrônicas, por exemplo. Contudo, os pesquisadores ainda perdem bastante tempo procurando por novas publicações em suas áreas de interesse.

Pode-se observar, portanto, que este tipo de aplicação pode ser bastante beneficiado com a adoção de algumas técnicas simples de personalização. Como esta abordagem pertence ao domínio dos problemas solucionados pelo *framework* proposto, optou-se pela definição do *MyRefs* como segundo exemplo de instanciação do *framework* .

O *MyRefs* é um repositório de referências adaptativo cujo objetivo é promover a disseminação de artigos científicos em uma comunidade específica. Ele realiza esta tarefa, fazendo uma constante mineração da base de dados, com base no modelo do usuário, apresentando os documentos mais recentes. A apresentação dos documentos pode ser organizada baseando-se nas datas, citações ou na quantidade de acesso entre outros.

4.2.2 Instanciação do Subsistema de Filtragem

Como discutido durante a definição do *MyNews*, a instanciação deste módulo abrange a definição de três *hotspots*: modelo de documentos, modelo de usuários e algoritmo de filtragem.

4.2.2.1 Modelo de Documentos

A instanciação deste *hotspot* corresponde às seguintes definições:

- Estrutura do Conteúdo: O conteúdo do *MyRefs* é composto por artigos científicos. Cada artigo segue a seguinte estrutura: título, autores, resumo, texto, publicação. Esta estrutura é utilizada para a criação da classe Artigo, correspondendo à instanciação do *hotspot* Item.

- A implementação do *hotspot* FabricaEixos define os eixos de valoração da aplicação. As classes que devem ser definidas são: Ontologia, Citacao e Atualidade. Estas classes correspondem ao tópico que o artigo se refere, às suas citações e ao ano de publicação, respectivamente. Estas classes, como discutido anteriormente, são utilizadas na instanciação do Perfil, Metrica e Eixo correspondente.

- As classes que instanciam o hotspot Eixo – EixoOntologia, EixoCitacao e EixoAtualidade – são definidas da seguinte forma:

    - Eixo ontologia: neste eixo o valor do artigo é o tópico em que está classificado na ontologia de domínio da aplicação.

    - Eixo citação: corresponde às referências feitas no artigo (bibliografia);

    - Eixo atualidade: corresponde ao ano de publicação do artigo.

  Além dos eixos definidos, para fins de simplificação, existem outras possibilidades como: tipo de artigo, número de acessos entre outros.

Uma distinção desta com a instanciação *MyNews* se refere ao eixo semântico, com a definição de uma nova ontologia. Outra distinção se refere a métrica utilizada para valoração dos artigos a partir da ontologia. Para o *MyRefs*, os documentos obtidos devem ser apenas aqueles relacionados aos tópicos de interesse dos usuário ou aos seus subtópicos.

4.2.2.2 Modelo de Usuários

O modelo do usuário do *MyRefs* corresponde às instanciações do *hotspot* Pefil. As classes que implementam este *hotspot* – PerfilOntologia, PerfilCitacao, PerfilAtualidade – devem permitir a definição de restrições quanto ao eixo correspondente, como discutido na seção 3.5.1. Estas classes devem implementar os métodos *setConteudoPerfil* e *getConteudoPerfil*.

4.2.2.3 Filtragem dos Documentos

Como descrito no processo de instanciação do *MyNews*, este *hotspot* é representado pela classe AlgoritmoFiltragem, cujas instanciações são responsáveis pela filtragem dos documentos, baseando-se no modelo do usuário.

A implementação deste *hotspot* para o *MyRefs* pode ser as classes AFiltragemI, AFiltragemH e AFiltragemC, que herdam a classe abstrata AlgoritmoFiltragem e implementam o método *filtrar*.

A classe AFiltragemI poderia ser utilizada para a geração de um grupo de artigos de interesse. O método *filtrar* utiliza uma função de combinação, que também deve ser definida com base nas características do domínio da aplicação.

A classe AFiltragemH poderia ser utilizada para a geração de um grupo de *hyperlinks* para os documentos citados em artigo específico.

A classe AfiltragemC poderia ser utilizada para a geração de um grupo de artigos mais citados, relacionados a um tópico específico.

4.2.3 Instanciação do Subsistema de Empacotamento

A instanciação deste subsistema, como discutido anteriormente, é mais simples do que a do Subsistema de Filtragem, no que se refere aos *hotspots*. Os pontos de flexibilização deste subsistema devem ser implementados por documentos XML, como ilustrado na Figura 34. A validação destes documentos é realizada pelo subsistema de validação. Os documentos que devem ser gerados para o *MyRefs* são: *ModeloEmpacotamento.xml, ModeloVisoes.xml* e *ModeloDocumento.xml*. Estes documentos se encontram no Apêndice II.

5. Conclusões e Trabalhos Futuros

5.1 Conclusões

Nesta dissertação foi definido um *framework* cujo objetivo é dar suporte ao desenvolvimento de aplicações adaptativas e/ou adaptáveis. A principal contribuição deste trabalho foi a organização do *framework* em três camadas que tratam os processos de filtragem, empacotamento e formatação utilizando os conceitos de *separation of concerns*. Esta abordagem facilita o processo de personalização, permitindo a reutilização de código e o mínimo de reprojeto.

Mais especificamente, esta dissertação propôs a desassociação dos processos de filtragem e formatação, de forma que a estruturação dos documentos e da aplicação em si fosse tratada isoladamente por uma camada intermediária, a camada de empacotamento.

Considerando o processo de filtragem, a abordagem utilizada é similar a outras abordagens utilizadas na literatura. Seu objetivo final, considerando o *MyNews*, é obter o conjunto de notícias mais relevantes para cada usuário. O *MyRefs*, por sua vez, obtém o conteúdo científico mais recente e que mais se aproxima da área de pesquisa do usuário em questão.

Já a camada de empacotamento, introduzida pelo *framework* de personalização proposto, permite a definição da estrutura abstrata da aplicação. Esta estrutura é gerada a partir de modelos definidos pelo instanciador do *framework* e irá delinear a navegação e estruturação dos documentos, independente das questões de formatação.

Implementando os *hotspots* do *framework* para o *MyNews*, por exemplo, pôde-se observar que a flexibilidade oferecida pelo *framework* facilita o refinamento dos algoritmos e configurações, permitindo a reutilização de código e o mínimo de reprojeto na criação de uma nova aplicação.

Apesar de este trabalho não ter tratado das questões de inicialização e evolução dos modelos, é importante ressaltar que boas técnicas de modelagem de usuários [ArdGoy99, Ardissono99a, Ardissono00a, Billsus99, Billsus00, Brusilovsky99, Kamba95, Fink97] são essenciais para o sucesso ou não de uma aplicação adaptativa.

Para a utilização das informações contidas nos modelos, o *framework* necessitou estabelecer meta-modelos para definição dos modelos. Dessa forma, desde que os módulos de atualização dos modelos formate as informações seguindo o meta-modelo definido, pode-se utilizar quaisquer técnicas de modelagem.

Este *framework* pode ser utilizado como um recurso para analisar as abordagens existentes para geração de sistemas adaptativos. O sistema de geração automática de apresentação multimídia *Cuypers* [Ossenbruggen01, Geurts01] decompõe o que o *framework* define como camada de formatação de acordo com os seus algoritmos para geração automática de apresentações multimídia. Se forem definidos algoritmos que gerem uma estrutura semântica adequada, baseada em intenções comunicativas, o processo de empacotamento seria capaz de prover a entrada para este protótipo. Dessa forma, o processo de empacotamento estaria posicionado em um nível anterior ao *Cuypers.*

Este trabalho se assemelha à proposta do IMMPS [Bordegoni97], que define um Padrão de Modelo de Referência para Sistemas de Apresentação Multimídia Inteligentes. A camada de *design*, contudo, encapsula a camada de empacotamento e algumas funcionalidades da camada de formatação do *framework*.

A abordagem descrita em [Ardissono00] propõe um jornal eletrônico (*OtaOnline*) que realiza a personalização do conteúdo e do detalhe da notícia se baseando na receptividade do usuário. Correspondendo ao processo de empacotamento do *framework*, a abordagem do *OtaOnline* oferece apenas dois tipos de páginas para a organização da informação: páginas de índices e páginas de notícias. A instanciação do *framework*, o *MyNews*, mantém a distinção entre os processos de formatação e empacotamento, permitindo aos editores e projetistas de interface experimentarem soluções alternativas com maior facilidade.

Atualmente, a implementação se concentra no acoplamento dos subsistemas de filtragem e empacotamento, que já estão implementados.

5.2 Trabalhos Futuros

Como continuidade do trabalho realizado no decorrer desta dissertação, sugerimos:

- finalizar a implementação dos demais subsistemas do *framework* (formatação e interface);
- implementar os módulos auxiliares de cadastramento e busca de conteúdo na Web.
- Implementar módulos para inicialização e evolução do modelo de usuário utilizando técnicas mais elaboradas como as apresentadas na 2.3.1.
- *design* do *framework* , com relação a seus *hotspots* e seu *frozenspot*.
- melhorar a especificação do modelo de empacotamento:
    - definir *hyperlinks* mais elaborados;
    - tornar o interpretador mais genérico, verificando a viabilidade de utilização de linguagens de consulta como, por exemplo, XQL, XSL, entre outras;
    - implementar um mecanismo que permita a mudança do modelo de empacotamento de um usuário específico, baseando-se, por exemplo, na receptividade do usuário ou preferências de apresentação. Este mecanismo conteria a definição de regras para degradação das visões ou substituição de padrões.

      Dessa forma, cada usuário teria um modelo de empacotamento distinto, ou seja, sua própria visão da aplicação.
- realização de testes, já que a satisfação do usuário deve ser empiricamente avaliada a partir de testes da qualidade de uso (usabilidade e comunicabilidade), ditando o sucesso ou não dos algoritmos e configurações utilizadas.
- verificar a facilidade de se fazer ajustes nos modelos e algoritmos, com base nos resultados de avaliações e testes com usuários.

6. Referências

[ArdGoy99] Ardissono, L., and A. Goy. *Tailoring the Interaction with Users in Electronic Shops*. In J. Kay, ed.: UM99 User Modeling: Proceedings of the Seventh International Conference. Wien New York: Springer-Verlag, 35-44, 1999.

[ArdGoy00] Ardissono, L., and A. Goy. *Dynamic generation of adaptive Web catalogs*. In: Lecture Notes in Computer Science n. 1892:Adaptive Hypermedia and Adaptive Web-Based Systems. Berlin: Springer Verlag, pp. 5-16, 2000.

[Ardissono99] Ardissono, L., Abarbero, C., Goy, A., and G. Petrone. *An agent Architecture for Personalized Web Stores*.In Proc. 3rd Int. Conf. on Autonomous Agents (Agents '99), pages 182--189, Seattle, WA, 1999.

[Ardissono99a] Ardissono, L., Console, L., and I. Torre. *On the application of personalization fechiques to news servers on the WWW*. In: Lecture Notes in Artificial Intelligence N. 1792. Berlin: Springer Verlag, pp. 261—272, 1999.

[Ardissono00] Ardissono, L., Console, L., and I. Torre. *Strategies for personalizing the access to news servers*. Working Notes of the Adaptive User Interfaces. Spring Symposium of AAAI (Technical Report SS-00-01), pp. 7-12, Stanford, CA, 2000, AAAI Press.

[Ardissono00a] Ardissono, L., and P. Torasso. *Dynamic user modeling in a Web store shell*. In Proc. 14th Conf. ECAI, to appear, Berlin, 2000.

[Benaki97] Benaki, E., Karkaletsis, V. A., and Constantine D. Spyrolpoulos. *Integrating User Modeling Into Information Extraction: The UMIE Prototype*. In Proceedings of the Sixth International Conference, 1997.

[Billsus99] Billsus, D. and M. Pazzani. *A Hybrid User Model for News Story Classification.* In: J.Kay, ed. UM99 User Modeling: Proceedings of the Seventh International Conference. Wien New York, Springer-Verlag:99-108, 1999.

[Billsus00] Billsus, D. and M. Pazzani. *User Modeling for Adaptive News Access*. User Modeling and User-Adapted Interaction 10(2-3), 147-180, 2000.

[Billsus00a] Billsus, D., Pazzani, J., and J. Chen. *A Learning Agent for Wireless News Access*. Proceedings of the 2000 International Conference on Intelligent User Interfaces , pages 33 - 36, 2000.

[Billsus02] Billsus, D. Brunk, C. A., Evans, G., Gladish , and M. Pazzani. *Adaptive Interfaces for Ubiquitous Web Acess*. Communications of The ACM, volume 45, pages 34 - 38, May, 2002.

[Bordegoni97] Bordegoni, M., Faconti, G., Maybury, M.T., Rist, T., Ruggieri, S., Trahanias, P., and M. Wilson. *A Standard Reference Model for Intelligent Multimedia Presentation Systems*. Computer Standards & Interfaces, 18(6-7):477-496, December, 1997.

[Bradley00] Bradley, K., Rafter, R. and Smyth, B. *Case-Based User Profiling for Content Personalization*. Book: AH. 2000

[Brusilovsky94] Brusilovsky, P. *Adaptive Hypermedia: An Attempt to Analyze and Generalize*. In Workshop on Adaptive hypertext and hypermedia at UM'94, Hyannis, Cape Cod, MA, USA, 1994.

[Brusilovsky96] Brusilovsky, P., Pesin, L. *Methods and Techniques of Adaptive Hypermedia*. User Modeling and User-Adapted Interaction, 6(2-3), 87-129, 1996.

[Brusilovsky97] Brusilovsky, P.. *Efficient Techniques for Adaptive Hypermedia*. In C. Nicolas and J. Mayfield, editors, Intelligent hypertext: advanced

techniques for the World Wide Web, volume 1326 of LNCS, pages 12--30. Springer, Berlin Heidelberg, 1997.

[Brusilovsky98] Brusilovsky, P. *Adaptive educational systems on the world-wide web: A review of available technologies*. In Proceedings of Workshop "WWW-Based Tutoring" at 4th International Conference on Intelligent Tutoring Systems (ITS'98), San Antonio, TX, 1998.

[Brusilovsky99] Brusilovsky, P. and Cooper, D. W. *ADAPTS: Adaptive hypermedia for Web-based performance support system*. Proceedings of the 2nd Workshop on Adaptive Systems and User Modeling on the WWW. 1999.

[Bushey99] Bushey, R., Mauney, J. M., and Thomas Deelman. *The development of Behavior-Based User Models for a Computer System*. In J. Kay, ed.: UM99 User Modeling: Proceedings of the Seventh International Conference. Wien New York: Springer, 1999.

[Carolis99] Carolis, Berardina. D., Rosis, Fiorella D., Andreoli, V. Cavallo, and M.L. De Cicco. *The Dynamic Generation of Hypertext Presentations of Medical Guidelines*. New Review of Hypermedia and Multimedia, 4, 1999

[Cingil00] Cingil, I., Dogac, A., and A. Azgin. *A broader approach to personalization*. Communications of the ACM 43(8):136--141, August 2000.

[Content01] Content Intelligence Group, Lyra Research, Inc. *Study Reveals 52 Percent of People over 55 Feel Web Is More Important Than Newspapers*. Research is featured in June 2001 issue of Content Intelligence.

[DeBra99] De Bra, P., Houben, G.J., and H. Wu. *AHAM: A Dexter-based Reference Model for Adaptive Hypermedia*. Proceedings of the ACM Conference on Hypertext and Hypermedia, pp. 147-156, Darmstadt, Germany, 1999.

[DeBra99a] De Bra, P., *Design Issues in Adaptive Hypermedia Application Development*, Proceedings of the Second Workshop on Adaptive Systems and User Modeling on the World Wide Web, pp. 29-39, Toronto and Banff, Canada, 1999.

[DeBra99b] De Bra, P., Brusilovsky, P., Houben, G.J., Adaptive Hypermedia, From Systems to Framework, ACM Computing Surveys, 1999.

[DeBra00] De Bra, P., Aerts, A., Houben, G.J., Wu, H., *Making General-Purpose Adaptive Hypermedia Work*. Proceedings of the WebNet Conference, pp. 117-123, 2000.

[Dorneles00] Dorneles, C. F. *Extração de Dados semi-estruturados com base em uma Ontologia*. Dissertação de mestrado, Universidade Federal do Rio Grande do Sul, maio 2000.

[Embley98] Embley, D., Campbell, D., Smith, and S. Liddle. *Ontology-based extraction and structuring of information from data-rich unstructured documents.* Proceedings of Conference on Information and Knowledge Management, 1998.

[Embley98a] Embley, D., Campbell, D., Smith, and S. Liddle. *A Conceptual-modeling approach to extracting data from the Web*. Proceedings of 17th International Conference on Conceptual Modeling, 1998.

[Encarnação99] Encarnação, L. M. and S. L. Stoev. *An Application-Independent User Support System Exploiting Action-Sequence Based User Modeling*. In: J. Kay, ed.: UM99 User Modeling: Proceedings of the Seventh International Conference. Wien New Yourk: Springer-Velarg, 145-254, 1999.

[Erdmann99] Erdmann, M., and R. Studer. *Ontologies as conceptual models for XML documents*. In KAW, October, 1999.

[Fayad99] Fayad, M., Schmidt, D., and Johnson, R. *Building Application Frameworks*, Wiley Computer Publishing, 1999.

[Fink96] Fink, J., Kobsa, A., and A. Nill. *User Oriented Adaptivity and Adaptability in the AVANTI Project*, in Designing for the Web: Empirical Studies, Microsoft Usability Group, Redmond WA, 1996.

[Fink97] Fink, J., Kobsa, A., and J. Schreck. *Personalized Hypermedia Information Provision through Adaptive and Adaptable System Features: User Modeling, Privacy and Security Issues*. Proc. of the Workshop Adaptive Systems and User Modeling on the World Wide Web of the 6th Int. Conf. on User Modeling, Chia Laguna, Sardinia, June 1997.

[Fink99] Fink, J., Kobsa, A., and A. Nill, *Adaptable and Adaptive Information Provision for All Users, Including Disabled and Elderly People* , The New Review of Hypermedia and Multimedia, Vol 4, 1999.

[Fink00] Fink, J., and A. Kobsa. *A Review and Analysis of Commercial User Modeling Serves for Personalization on the World Wide Web*. User Modeling and User-Adapted Interaction 9, Special Issue on Deployed User Modeling, 2000.

[Fink01] Fink, J. *User Modeling Servers: Requirements, Design, and Evaluation*. Department of Mathematics and Computer Science, University of Essen, Germany, 2001.

[Geurts01] Geurts, j., Ossenbruggen, J. V., and Lynda Hardman. *Application-Specific Constraints for Multimedia Presentation Generation*. (technical report INS-R0107), 2001

[Haym00] Haym Hirsh. Chumki Basu and Brian D. Davison. *Learning to Personalize.* Communications of ACM, August, Vol. 43 No. 8, 2000.

[Henze00] Henze, N. and W. Nejdl. *Extensible Adaptive Hypermedia Courseware: Integrating Different Courses and Web Material*. In Adaptive Hypermedia and Adaptive Web-Based Systems. Berlin, Springer, 109-120, 2000.

[Hof98] Hof, R. D., Green, H., and Himmelstein, L. *Now it's YOU WEB*. Business Week, pages 68-74, October 5, 1998.

[IPTC00] IPTC, *International Press Telecommunications Council*, 2000. Disponível em: http://www.iptc.org/.

[Jameson01] Jameson, A. *Modeling both the Context and the User*. Personal Technologies 5(1), 2001.

[Joerding98] Joerding, T., and K., Meissner. *Intelligent Multimedia Presentations in the Web: Fun without Annoyance*. Proccedings of the Seventh International World Wide Web Conference (WWW7), Brisbane, Australia, 1998.

[Kamba95] Kamba, T., Bharat, K., and M. C. Albers. *The Krakatoa Chronicle – An Interactive, Personalized, Newspaper on the Web*. In Proc. of the Fourth International World Wide Web Conference, pp. 159-170, Nov 1995.

[Kamba95a] Kamba, T., Bharat, K., and M. C. Albers. *An Interactive, Personalized, Newspaper on WWW*. Proc. 4th Intl. World Wide Web Conference, 1995.

[Khan00] Khan, L., and McLeod, D. *Audio Structuring and Personalized Retrieval Using Ontologies*. In Proceedings of IEEE Advances in Digital Libraries, Library of Congress, Washington, DC,May 2000.

[Kobsa94] Kobsa, A. *Special Issue on User Modeling Shell Systems*. User Modeling and User Adapted Interaction, 4(2) and 4(3), 1994.

[Kobsa95] Kobsa, A., and Pohl, W. *The user modeling shell system BGP-MS*. User Modeling and UserAdapted Interaction 4(2):59—106, 1995.

[kobsa01] Kobsa, A. Koenemann, J. and Pohl, W. *Personalized Hypermedia Presentation Techniques for Improving Online Customer Relationships*. The Knowledge Engineering Review, forthcoming, 2001.

[Martins02] Martins, C. L., Coelho, A. S. T., Barbosa, S. D. J., Casanova, M. A., Lucena, C. J. P. *A Framework for Filtering and Packaging Hypermedia Documents*. Second International Conference on Adaptive Hypermedia and Adaptive Web Based Systems. Spain, May, 2002.

[Ossenbruggen01] Ossenbruggen, J.V., Geurts, J., Cornelissen, F., Rutledge, L., and Lynda Hardman. *Towards Second and Third Generation Web-Based Multimedia*. In The Tenth International World Wide Web Conference, pages 479-488, Hong Kong, May 1-5, 2001.

[Parsaye00] Parsaye, Kamran. PQ: *The Personalization Quotient of a Website*. Published by personalization.com. NovuWeb, Inc. 2000.

[Perkowitz00] Perkowitz, Mike and Oren Etzioni. *Adaptive Web Sites*. Communications of ACM, August 2000, Vol. 43 No. 8.

[Petrelli99] Petrelli, D., De Angeli, A., and Gregorio Convertino. *A User-Centered Approach to User Modelling*. Proceedings of the Seventh International Conference, UM´99, 255-264. 1999

[Pohl99] Pohl, W. and Achim Nick. Machine *Learning and Knowledge Representation in the LaboUr Approach to User Modeling*. In Proc. 7th Int. Conf. on User Modeling, pp. 179--188, Banff, Canada, 1999.

[Pree95] Pree, W. *Design Patterns for Object-Oriented Software Development*. Addison Wesley, 1995.

[RDF99] W3C. *Resource Description Framework (RDF) Model and Syntax Specification*. W3C Recommendation, February de 1999.

[RDF00] W3C. *Resource Description Framework (RDF) Schema Specification*. W3C Recommendation, March, 2000.

[Rutledge98] Rutledge, L., Hardman, L., Ossenbruggen, J.V. and Dick C. A. Bulterman. *Implementing Adaptability in the Standard Reference Model for Intelligent Multimedia Presentation Systems*. In International Conference on Multimedia Modeling, pages 12-30, 12-15 October 1998

[Rutledge00] Rutledge, L, Bailey, Brian, Ossembruggen, J.V., Hardman, L., and Joost Geurts. *Generating Presentation Constraints from Rhetorical Structure.* In Proceedings of 11th ACM conference on Hypertext and Hypermedia, pages 19-28, May, 2000.

[Rutledge00a] Rutledge, L., Davis, J., Ossenbruggen, J.V. and Lynda Hardman. *Inter-dimensional Hypermedia Communicative Devices for Rhetorical Structure*. In Proceedings of International Conference on Multimedia Modeling, pages 89-105, Japan, November 13-15, 2000.

[Saarela96] Saarela, J., Turpeinen, M., and R. Sulonen. *Requirements for an electronic Newspaper*. Proc. of the Second Int. Baltic Workshop on DB and IS. pp. 84-92. 1996

[Samulowitz00] Samulowitz, M. *Designing a Hierarchy of User Models for Context-Aware Applications*. Position Paper. Workshop on 'Situated Interaction in Ubiquitous Computing' at CHI 2000, 3rd of April 2000.

[Schema01] W3C. XML Schema Part 1: Structures, W3C Recommendation, 2001.

[SMIL98] W3C. *Synchronized Generalized Markup Language (SMIL) 1.0 Specification*. W3C Recommendation, June, 1998.

[Turpeinen96] Turpeinen, M., Saarela, J., and T. Puskala. 1996. *Architecture for Agent Mediated Personalised News Services*. In PAAM 1996.

[XML00] W3C. *Extensible Markup Language (XML) 1.0 (Second Edition)*. W3C Recommendation, October, 2000.

[XSLT99] W3C. XSL Transformations (XSLT) 1.0. W3C Recommendation, November, 1999.

[Wu99] Wu, H., Houben, G.J., De Bra, P., *User Modeling in Adaptive Hypermedia Applications*, Proceedings of the "Interdisciplinaire Conferentie Informatiewetenschap", pp. 10-21, Amsterdam, 1999.

[Wu99a] Wu, H., Houben, G.J., De Bra, P., *Authoring Support for Adaptive Hypermedia Applications*, Proceedings of the ED-MEDIA Conference, pp. 364-369, AACE, Seattle, 1999.

[Wu00] Wu, H., De Bra, P., Aerts, A., Houben, G.J., *Adaptation Control in Adaptive Hypermedia Systems*. Proceedings of the AH2000 Conference, pp. 250-259, 2000. Lecture Notes in Computing Science, Vol. 1892, Springer.

[Tarr99] Tarr, P., Ossher, H., Harrison, W., and S. Sutton. *N Degrees of Separation: Multi-Dimensional Separation of Concerns*. Proceedings 21st International Conference on Software Engineering (ICSE'99), May 1999.

[Uschold96] Uschold, M., and Gruninger, M. *Ontologies: principles, methods, and applications.* Knowledge Engineering Review, 11(2), 93—155, 1996.

Apêndice I

Modelos do Framework

6.1 Esquema do Modelo de Empacotamento

MetaModeloEmpacotamento.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

 <!-- definicao de tipos simples-->
 <xs:simpleType name="focoType">
  <xs:restriction base="xs:string"/>
 </xs:simpleType>
 <xs:simpleType name="expressaoType">
  <xs:restriction base="xs:string"/>
 </xs:simpleType>
 <xs:simpleType name="linkType">
  <xs:restriction base="xs:string"/>
 </xs:simpleType>
 <xs:simpleType name="paramRankingType">
  <xs:restriction base="xs:string"/>
 </xs:simpleType>
 <xs:simpleType name="paramOrdenacaoType">
  <xs:restriction base="xs:string"/>
 </xs:simpleType>
 <xs:simpleType name="paramLinkAlgType">
  <xs:restriction base="xs:string"/>
 </xs:simpleType>
 <xs:simpleType name="tipoProcuraType">
  <xs:restriction base="xs:NMTOKEN">
   <xs:enumeration value="IN"/>
   <xs:enumeration value="OUT"/>
  </xs:restriction>
 </xs:simpleType>

 <!-- definicao de tipos complexos-->
 <xs:complexType name="linkAlgType">
  <xs:simpleContent>
   <xs:extension base="xs:string">
     <xs:attribute name="paramLinkAlg" type="paramLinkAlgType"/>
   </xs:extension>
  </xs:simpleContent>
 </xs:complexType>

 <xs:complexType name="itemLinkType">
```

```
  <xs:sequence>
    <xs:element name="link" type="linkType" minOccurs="1" maxOccurs="1"
/>
    <xs:element name="linkAlg" type="linkAlgType" minOccurs="0"
maxOccurs="1" />
   </xs:sequence>
  </xs:complexType>

 <xs:complexType name="hiperlinkType">
   <xs:sequence>
    <xs:element name="itemLink" type="itemLinkType" minOccurs="0"
maxOccurs="unbounded" />
   </xs:sequence>
  </xs:complexType>

 <xs:complexType name="rankingType">
  <xs:simpleContent>
   <xs:extension base="xs:string">
     <xs:attribute name="paramRanking" type="paramRankingType"/>
   </xs:extension>
  </xs:simpleContent>
 </xs:complexType>

 <xs:complexType name="ordenacaoType">
  <xs:simpleContent>
   <xs:extension base="xs:string">
     <xs:attribute name="paramOrdenacao" type="paramOrdenacaoType"/>
   </xs:extension>
  </xs:simpleContent>
 </xs:complexType>


 <!-- o elemento modelo eh utilizado quando o conteudo do grupo eh
metadado -->
 <xs:complexType name="modeloType">
  <xs:sequence>
   <xs:element name="atributo" type="xs:string" minOccurs="1"
maxOccurs="unbounded" />
   <xs:element name="limite" type="xs:string" />
  </xs:sequence>
  <xs:attribute name="nomeModelo" type="xs:string" use="required"/>
  <xs:attribute name="metadado" type="xs:string" use="required"/>
  <xs:attribute name="tipoProcura" type="tipoProcuraType"
use="required"/>
 </xs:complexType>

 <xs:complexType name="intervaloRankType">
  <xs:attribute name="ini" type="xs:string" use="required"/>
  <xs:attribute name="fim" type="xs:string" use="required"/>
 </xs:complexType>

 <xs:complexType name="visaoType">
  <xs:choise>
    <xs:element name="numDoc" type="xs:string" minOccurs="1"
maxOccurs="1" />
    <xs:element name="intervaloRank" type="intervaloRankType"
minOccurs="1" maxOccurs="1" />
```

```
    <xs:element name="nomeEixo" type="xs:string" minOccurs="1"
maxOccurs="1" />
  </xs:choise>
  <xs:attribute name="nome" type="xs:string" use="required"/>
 </xs:complexType>

 <xs:group name="fonteConteudo">
  <xs:choice>
   <xs:sequence>
    <xs:element name="ranking" type="rankingType" minOccurs="0"
maxOccurs="1" />
    <xs:element name="visao" type="visaoType" minOccurs="0"
maxOccurs="unbounded" />
    <xs:element name="foco" type="focoType" minOccurs="0" maxOccurs="1"
/>
   </xs:sequence>
   <xs:element name="modelo" type="modeloType" />
  </xs:choice>
 </xs:group>

 <xs:complexType name="grupoType">
  <xs:sequence>
   <xs:group ref="fonteConteudo"/>
   <xs:element name="ordenacao" type="ordenacaoType" minOccurs="0"
maxOccurs="1" />
   <xs:element name="hiperlink" type="hiperlinkType" minOccurs="0"
maxOccurs="1" />
  </xs:sequence>
  <xs:attribute name="nomeGrupo" type="xs:string" use="required"/>
  <xs:attribute name="tituloGrupo" type="xs:string" use="required"/>
 </xs:complexType>

 <xs:complexType name="padraoType">
  <xs:sequence>
   <xs:element name="grupo" type="grupoType" minOccurs="1"
maxOccurs="unbounded" />
  </xs:sequence>
  <xs:attribute name="nomePadrao" type="xs:string" use="required"/>
 </xs:complexType>

 <xs:complexType name="modeloEmpacotamentoType">
  <xs:sequence>
   <xs:element name="padrao" type="padraoType" minOccurs="1"
maxOccurs="unbounded" />
  </xs:sequence>
 </xs:complexType>

 <xs:element name="modeloEmpacotamento" type="modeloEmpacotamentoType" >
 <!-- Link deve ser um padrao ja definido-->
  <xs:key name="idPadrao" >
   <xs:selector xpath="padrao" />
   <xs:field xpath="@nomePadrao" />
  </xs:key>
  <xs:keyref name="refIdPadrao" refer="idPadrao" >
   <xs:selector xpath="padrao" />
   <xs:field xpath="link" />
  </xs:keyref>
 </xs:element>
```

```
</xs:schema>
```

6.2 Esquema do Modelo de Documento

MetaModeloDocumento.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<!-- definicao de tipos simples-->

 <xs:simpleType name="tipoType">
  <xs:restriction base="xs:NMTOKEN">
   <xs:enumeration value="text"/>
   <xs:enumeration value="number"/>
  </xs:restriction>
 </xs:simpleType>

 <xs:simpleType name="obrigatorioType">
  <xs:restriction base="xs:NMTOKEN">
   <xs:enumeration value="sim"/>
   <xs:enumeration value="nao"/>
  </xs:restriction>
 </xs:simpleType>

 <xs:simpleType name="multivaloradoType">
  <xs:restriction base="xs:NMTOKEN">
   <xs:enumeration value="sim"/>
   <xs:enumeration value="nao"/>
  </xs:restriction>
 </xs:simpleType>

 <xs:simpleType name="dominioType">
  <xs:restriction base="xs:NMTOKEN">
   <xs:enumeration value="enumeravel"/>
   <xs:enumeration value="infinito"/>
  </xs:restriction>
 </xs:simpleType>

<!-- definicao dos tipos complexos-->


 <xs:complexType name="atributoItemType">
  <xs:attribute name="nome" type="xs:string" use="required"/>
  <xs:attribute name="tipo" type="tipoType" use="required"/>
  <xs:attribute name="obrigatorio" type="obrigatorioType"
use="required"/>
  <xs:attribute name="multivalorado" type="multivaloradoType"
use="required"/>
 </xs:complexType>

 <xs:complexType name="eixoValoracaoType">
```

```
  <xs:attribute name="nome" type="xs:string" use="required"/>
  <xs:attribute name="dominio" type="dominioType" use="required"/>
  <xs:attribute name="metrica" type="xsd:string" use="required"/>
 </xs:complexType>

<xs:complexType name="itemType">
 <xs:sequence>
  <xs:element name="atributoItem" type="atributoItemType" minOccurs="1"
maxOccurs="unbounded" />
    <xs:element name="eixoValoracao" type="eixoValoracaoType"
minOccurs="1" maxOccurs="unbounded" />
  </xs:sequence>
  <xs:attribute name="nome" type="xs:string" use="required"/>
 </xs:complexType>

<xs:element name="item" type="itemType" /></xs:schema>
```

6.3 Esquema das Visões

MetaModeloVisoes.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

 <xs:complexType name="atributoItemType">
  <xs:attribute name="nome" type="xs:string" use="required"/>
 </xs:complexType>

 <xs:complexType name="visaoType">
  <xs:sequence>
    <xs:element name=”atributoItem” type=”atributoItemType” minOccurs="1"
maxOccurs="unbounded" />
  </xs:sequence>
  <xs:attribute name="nome" type="xs:string" use="required"/>
 </xs:complexType>

 <xs:element name=”visoes” type=”visaoType” minOccurs="1"
maxOccurs="unbounded" />

</xs:schema>
```

6.4 Esquema do modelo de usuário

MetaModeloUsuario.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:complexType name="enumeravelType">
 <xs:sequence>
  <xs:element name="valor" type="xs:string" minOccurs="1"
   maxOccurs="unbounded"/>
 </xs:sequence>
</xs:complexType>

<xs:complexType name="infinitoType">
  <xs:simpleContent>
   <xs:extension base="xs:string">
     <xs:attribute name="ini" type="xs:string" use=”required”/>
     <xs:attribute name="fim" type="xs:string" use=”required”/>
   </xs:extension>
  </xs:simpleContent>
 </xs:complexType>

<xs:complexType name="eixoValoracaoType">
 <xs:sequence>
  <xs:choice>
   <xs:element name="enumeravel" type="enumeravelType" minOccurs="1"
    maxOccurs="1"/>
   <xs:element name="infinito" type="infinitoType" minOccurs="1"
    maxOccurs="1"/>
  </xs:choice>
 </sequence>
 <xs:attribute name="nome" type="xs:string" use="required"/>
</xs:complexType>

<xs:complexType name="perfilType">
 <xs:sequence>
  <xs:element name="eixoValoracao" type="eixoValoracaoType"
   minOccurs="1" maxOccurs="unbounded" />
 </xs:sequence>
</xs:complexType>

<xs:element name="perfil" type="perfilType" />

</xs:schema>
```

6.5 Esquema do resultado do empacotamento

DocumentosEmpacotados.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

 <xs:complexType name="valorEixoType">
  <xs:attribute name="valorIni" type="xs:string" use=”required”/>
  <xs:attribute name="valorFim" type="xs:string"/>
</xs:complexType>

<xs:complexType name="linkAlgType">
  <xs:simpleContent>
   <xs:extension base="xs:string">
     <xs:attribute name="paramLinkAlg" type="xs:string"/>
   </xs:extension>
  </xs:simpleContent>
 </xs:complexType>

 <xs:complexType name="itemLinkType">
  <xs:sequence>
    <xs:element name="link" type="xs:string" minOccurs="1" maxOccurs="1"
/>
    <xs:element name="linkAlg" type="xs:string" minOccurs="0"
maxOccurs="1" />
   </xs:sequence>
  </xs:complexType>

 <xs:complexType name="hiperlinkType">
   <xs:sequence>
    <xs:element name="itemLink" type="xs:string" minOccurs="0"
maxOccurs="unbounded" />
   </xs:sequence>
  </xs:complexType>

<xs:complexType name="atributoType">
  <xs:simpleContent>
   <xs:extension base="xs:string">
     <xs:attribute name="nome" type="xs:string" use=”required”/>
   </xs:extension>
  </xs:simpleContent>
 </xs:complexType>

<xs:complexType name="eixoValoracaoType">
  <xs:simpleContent>
   <xs:extension base="xs:string">
     <xs:attribute name="nome" type="xs:string" use=”required”/>
   </xs:extension>
  </xs:simpleContent>
 </xs:complexType>

<!-- o elemento item eh utilizado quando o conteudo do grupo e um
conjunto de documentos xml -->
 <xs:complexType name="itemType">
```

```
  <xs:sequence>
   <xs:element name="atributo" type="atributoType" minOccurs="1"
maxOccurs="unbounded" />
   <xs:element name="eixoValoracao" type="eixoValoracaoType"
minOccurs="1" maxOccurs="unbounded" />
  </xs:sequence>
</xs:complexType>

<!-- o elemento modelo eh utilizado quando o conteudo do grupo eh
metadado -->
 <xs:complexType name="modeloType">
  <xs:choice>
   <xs:element name="valorEixo" type="valorEixoType" />
  </xs:choice>
  <xs:attribute name="nomeModelo" type="xs:string" use="required"/>
  <xs:attribute name="eixoValoracao" type="xs:string" use="required"/>
</xs:complexType>

 <xs:group name="conteudo">
  <xs:choice>
   <xs:element name="documentos" type="itemType" minOccurs="1"
maxOccurs="1" />
   <xs:element name="modelo" type="modeloType" minOccurs="1"
maxOccurs="1" />
  </xs:choice>
 </xs:group>

 <xs:complexType name="grupoType">
  <xs:sequence>
   <xs:group ref="conteudo"/>
   <xs:element name="hiperlink" type="hiperlinkType" minOccurs="0"
maxOccurs="1" />
  </xs:sequence>
  <xs:attribute name="nomeGrupo" type="xs:string" use="required"/>
  <xs:attribute name="tituloGrupo" type="xs:string" use="required"/>
 </xs:complexType>

 <xs:complexType name="padraoType">
  <xs:sequence>
    <xs:element name="grupo" type="grupoType" minOccurs="1"
maxOccurs="unbounded" />
  </xs:sequence>
  <xs:attribute name="nomePadrao" type="xs:string" use="required"/>
 </xs:complexType>

 <xs:complexType name="documentosEmpacotadosType">
  <xs:sequence>
    <xs:element name="padrao" type="padraoType" minOccurs="1"
maxOccurs="unbounded" />
  </xs:sequence>
  <xs:attribute name="tipoItem" type="xs:string" use="required"/>
 </xs:complexType>
 <xs:element name="documentosEmpacotados"type="documentosType" />

</xs:schema>
```

Apêndice II

1 Modelos do MyNews

1.1 Modelo de Empacotamento

ModeloEmpacotamento.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<modeloEmpacotamento
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation=
      "d:\\temp\\MetaModeloEmpacotamento.xsd">

<padrao nomePadrao="P1">
  <grupo nomeGrupo="Ga" tituloGrupo="Editorias de Interesse">
   <modelo nomeModelo = "ModeloUsuario" metadado="EixoOntologia"
tipoProcura="IN">
    <atributo> Tema </atributo>
    <limite> 10 </limite>
   </modelo>
   <ordenacao> OrdemAlfabetica </ordenacao>
   <hiperlink>
     <link>P1</link>
     <link>P2</link>
     <link>P5</link>
   </hiperlink>
  </grupo>
 </padrao>

 <padrao nomePadrao="P2">
  <grupo nomeGrupo="Gb" tituloGrupo="Demais Editorias">
   <modelo nomeModelo = "ModeloUsuario" metadado="EixoOntologia"
tipoProcura="OUT">
    <atributo> Tema </atributo>
    <limite> 10 </limite>
   </modelo>
   <ordenacao> OrdemAlfabetica </ordenacao>
   <hiperlink>
     <link>P1</link>
     <link>P2</link>
     <link>P5</link>
   </hiperlink>
  </grupo>
 </padrao>

 <padrao nomePadrao="P3">
  <grupo nomeGrupo="Gc" tituloGrupo="Notícias de Interesse">
   <ranking> AFiltragemX </ranking>
```

```
   <visao nome=”dv3”>
    <numDoc> 2 </numDoc>
   </visao>
   <visao nome=”dv2”>
    <numDoc> 2 </numDoc>
   </visao>
   <visao nome=”dv1”>
    <numDoc> -1 </numDoc>
   </visao>
   <hiperlink>
     <link>P1</link>
     <link>P2</link>
     <link>P4</link>
   </hiperlink>
  </grupo>
 </padrao>

 <padrao nomePadrao="P4">
  <grupo nomeGrupo="Gd" tituloGrupo="Descrição da Notícia">
   <visao nome=”dv4”>
    <numDoc> 1 </numDoc>
   </visao>
   <foco> ObterDocFoco </foco>
  </grupo>

  <grupo nomeGrupo="Ge" tituloGrupo="Notícias relacionadas">
   <ranking paramRanking="ObterDocFoco"> AFiltragemY </ranking>
   <visao nome=”dv1”>
    <numDoc> 6 </numDoc>
   </visao>
   <ordenacao> OrdemAlfabetica </ordenacao>
   <hiperlink>
     <link>P1</link>
     <link>P2</link>
     <link>P4</link>
   </hiperlink>
  </grupo>
 </padrao>

<padrao nomePadrao="P5">
  <grupo nomeGrupo="Gf" tituloGrupo="Notícias">
   <ranking paramRanking=”ObterDocFoco”>AFiltragemX</ranking>
   <visao nome=”dv3”>
    <numDoc> 2 </numDoc>
   </visao>
   <visao nome=”dv2”>
    <numDoc> -1 </numDoc>
   </visao>
   <hiperlink>
     <link>P1</link>
     <link>P2</link>
     <link>P4</link>
   </hiperlink>
  </grupo>
 </padrao>

</modeloEmpacotamento>
```

1.2 Modelo de Documento

ModeloDocumento.xml

```
<?xml version=”1.0” encoding=”utf-8”?>

<item nome= “Noticia”
    xslns:xsi=”http://www.w3.org/2001/XMLSchema-instance”
....xsi:noNamespaceSchemaLocation=
       ”c:\\xml\\modelos\\metaModeloDocumento.xsd”>

 <atributoItem nome =”id” tipo=”number” obrigatorio=”sim”
multivalorado=”nao”/>
 <atributoItem nome =”titulo” tipo=”text” obrigatorio=”sim”
multivalorado=”nao”/>
 <atributoItem nome =”subtitulo” tipo=”text” obrigatorio=”sim”
multivalorado=”nao”/>
 <atributoItem nome =”resumo” tipo=”text” obrigatorio=”sim”
multivalorado=”nao”/>
 <atributoItem nome =”autor” tipo=”text” obrigatorio=”nao”
multivalorado=”sim”/>
 <atributoItem nome =”texto” tipo=”text” obrigatorio=”sim”
multivalorado=”nao”/>
 <atributoItem nome =”imagem” tipo=”text” obrigatorio=”nao”
multivalorado=”nao”/>

 <eixoValoracao nome=”EixoOntologia” dominio=”Enumeravel”
metrica=”MetricaOntologia”/>
 <eixoValoracao nome=”EixoImportancia” dominio=”Infinito”
metrica=”MetricaImportancia”/>
 <eixoValoracao nome=”EixoAtualidade” dominio=”Infinito”
metrica=”MetricaImportancia”/>

</item>
```

1.3 Modelo de Usuário

ModeloUsuario.xml

```
<?xml version=”1.0” encoding=”utf-8”?>

<perfil
    xslns:xsi=http://www.w3.org/2001/XMLSchema-instance
....xsi:noNamespaceSchemaLocation=
       ”c:\\xml\\modelos\\MetaModeloUsuario.xsd”>

 <eixoValoracao nome =”Ontologia”/>
  <enumeravel>
   <valor>A</valor>
   <valor>B</valor>
  </enumeravel>
 </eixoValoracao>

 <eixoValoracao nome =”Importancia”/>
  <infinito ini =”1” fim=”3”/>
 </eixoValoracao>

 <eixoValoracao nome =”Atualidade”/>
  <infinito ini =”15/05/2002” fim=”15/05/2002”/>
 </eixoValoracao>

</perfil>
```

1.4 Esquema do Modelo de Documento

ModeloDocumento.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

 <xs:complexType name="atributoType">
  <xs:sequence>
    <xs:element name="id" type="xs:string" minOccurs="1"
maxOccurs="1" />
    <xs:element name="titulo" type="xs:string" minOccurs="1"
maxOccurs="1" />
    <xs:element name="subtitulo" type="xs:string" minOccurs="1"
maxOccurs="1" />
    <xs:element name="autor" type="xs:string" minOccurs="0"
maxOccurs="unbounded" />
    <xs:element name="texto" type="xs:string" minOccurs="1"
maxOccurs="1" />
    <xs:element name="imagem" type="xs:string" minOccurs="0"
maxOccurs="1" />
 </xs:complexType>

 <xs:complexType name="eixoValoracaoType">
  <xs:sequence>
    <xs:element name="EixoOntologia" type="xs:string" minOccurs="1"
maxOccurs="1" />
    <xs:element name="EixoAtualidade" type="xs:string" minOccurs="1"
maxOccurs="1" />
    <xs:element name="EixoImportancia" type="xs:string" minOccurs="1"
maxOccurs="1" />
 </xs:complexType>

 <xs:complexType name="noticiaType">
  <xs:sequence>
    <xs:element name="atributo" type="atributoType" minOccurs="1"
maxOccurs="1" />
    <xs:element name="eixoValoracao" type="eixoValoracaoType"
minOccurs="1" maxOccurs="1" />
 </xs:complexType>

<xs:complexType name="documentosType">
  <xs:sequence>
    <xs:element name="noticia" type="noticiaType" minOccurs="1"
maxOccurs="unbounded" />
 </xs:complexType>

 <xs:element name="documentos" type="documentosType" />

</xs:schema>
```

1.5 Visões de Documentos

ModeloVisoes.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<visoes
    xslns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation=
        "d:\\xml\\modelos\\MetaModeloVisoes.xsd">

 <visao nome = “dv1”>
  <atributoItem nome = “id”>
  <atriutoitem nome = “titulo”>
 </visao>
 <visao nome = “dv2”>
  <atributoItem nome = “id”>
  <atributoItem nome = “titulo”>
  <atributoItem nome = “resumo”>
 </visao>
 <visao nome = “dv3”>
  <atributoItem nome = “id”>
  <atributoItem nome = “titulo”>
  <atributoItem nome = “resumo”>
  <atributoItem nome = “imagem”>
 </visao>
 <visao nome = “dv4”>
  <atributoItem nome = “id”>
  <atributoItem nome = “titulo”>
  <atributoItem nome = “resumo”>
  <atributoItem nome = “imagem”>
  <atributoItem nome = “texto”>
  <atributoItem nome = “autor”>
 </visao>

</visoes>
```

1.6 Resultado da Filtragem dos Documentos

Documentos.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<documentos tipo="noticia">

 <item rank="2,24">

  <atributoItem nome="id">2</atributoItem>
  <atributoItem nome="titulo">n2</atributoItem>
  <atributoItem nome="resumo">resumo2</atributoItem>
  <atributoItem nome="imagem">endImagem2</atributoItem>
  <atributoItem nome="texto">texto2</atributoItem>
  <atributoItem nome="autor">autor2_1</atributoItem>
  <atributoItem nome="autor">autor2_2</atributoItem>

  <eixoValoracao nome="EixoOntologia">C</eixoValoracao>
  <eixoValoracao nome="EixoImportancia">1</eixoValoracao>
  <eixoValoracao nome="EixoAtualidade">03/12/2001</eixoValoracao>

 </item>

 <item rank="3,14">

  <atributoItem nome="id">1</atributoItem>
  <atributoItem nome="titulo">n1</atributoItem>
  <atributoItem nome="resumo">resumo1</atributoItem>
  <atributoItem nome="imagem">endImagem1</atributoItem>
  <atributoItem nome="texto">texto1</atributoItem>
  <atributoItem nome="autor">autor1</atributoItem>

  <eixoValoracao nome="EixoOntologia">A</eixoValoracao>
  <eixoValoracao nome="EixoImportancia">3</eixoValoracao>
  <eixoValoracao nome="EixoAtualidade">05/12/2001</eixoValoracao>

 </item>

...
...
</documentos>
```

1.7 Resultado do Empacotamento dos Documentos

DocumentosEmpacotados.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<documentosEmpacotados tipo=”noticia”>

 <padrao nomePadrao=”P3”>
  <grupo nomeGrupo=”G3” titulo=”Noticias de Interesse”>
   <documentos>
    <item>
     <atributo nome=”id”> 2 </atributo>
     <atributo nome=”titujo”> n2 </atributo>
     <atributo nome=”resumo”> resumo2 </atributo>
     <atributo nome=”imagem”> endImagem2 </atributo>

     <eixoValoracao nome=”EixoOntologia> C </eixoValoracao>
     <eixoValoracao nome=”EixoImportancia> 1 </eixoValoracao>
     <eixoValoracao nome=”EixoAtualidade> 03/12/2001 </eixoValoracao
    </item>

    <item>
     <atributo nome=”id”> 1 </atributo>
     <atributo nome=”titujo”> n1 </atributo>
     <atributo nome=”resumo”> resumo1 </atributo>
     <atributo nome=”imagem”> endImagem1 </atributo>

     <eixoValoracao nome=”EixoOntologia> A </eixoValoracao>
     <eixoValoracao nome=”EixoImportancia> 3 </eixoValoracao>
     <eixoValoracao nome=”EixoAtualidade> 05/12/2001 </eixoValoracao
    </item>

    ...
    ...

   </documentos>

   <hiperlink>
    <itemLink>
     <link>P1</link>
    </itemLink>
    <itemLink>
     <link>P2</link>
    </itemLink>
    <itemLink>
     <link>P4</link>
    </itemLink>
   </hiperlink>
  </grupo>
 </padrao>

 ...
 ...
 ...
 ...
```

```
 <padrao nomePadrao="P1">
  <grupo nomeGrupo="G1" tituloGrupo="Editorias de Interesse">
   <modelo nomeModelo="ModeloUsuario" eixoValoracao="EixoOntologia">
    <valorEixo valorIni=”A”/>
    <valorEixo valorIni=”B”/>
    <valorEixo valorIni=”C”/>
   </modelo>
   <hiperlink>
     <link>P1</link>
     <link>P2</link>
     <link>P3</link>
   </hiperlink>
  </grupo>
 </padrao>

 ...
 ...

</documentosEmpacotados>
```

1.8 Transformação do documento *Documentos.xml*

TransformacaoVisoes.xsl

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="1.0">
  <xsl:output method="xml" indent="yes"/>

  <xsl:template match="/">
     <documentos>
        <xsl:apply-templates/>
     </documentos>
  </xsl:template>

  <xsl:template match="item">

    <xsl:choose>
       <xsl:when test="@visao='dv1'">
          <xsl:call-template name="dv1"/>
       </xsl:when>

       <xsl:when test="@visao='dv2'">
       <xsl:call-template name="dv2"/>
       </xsl:when>

       <xsl:when test="@visao='dv3'">
       <xsl:call-template name="dv3"/>
       </xsl:when>

       <xsl:when test="@visao='dv4'">
       <xsl:call-template name="dv4"/>
       </xsl:when>

    </xsl:choose>

  </xsl:template>


  <xsl:template name="dv1">
  <item>
   <xsl:for-each select="atributoItem">

    <xsl:variable name="tipo" select="@nome"/>
    <xsl:choose>
     <xsl:when test='$tipo="id"'>
       <atributoItem nome="id">
      <xsl:value-of select="."/>
       </atributoItem>
     </xsl:when>
     <xsl:when test='$tipo="titulo"'>
      <atributoItem nome="titulo">
       <xsl:value-of select="."/>
      </atributoItem>
     </xsl:when>
    </xsl:choose>
```

```
  </xsl:for-each>
  <xsl:copy-of select="eixoValoracao"/>
   </item>

</xsl:template>


<xsl:template name="dv2">
<item>
 <xsl:for-each select="atributoItem">

  <xsl:variable name="tipo" select="@nome"/>
  <xsl:choose>
   <xsl:when test='$tipo="id"'>
    <atributoItem nome="id">
     <xsl:value-of select="."/>
    </atributoItem>
   </xsl:when>
   <xsl:when test='$tipo="titulo"'>
    <atributoItem nome="titulo">
     <xsl:value-of select="."/>
    </atributoItem>
   </xsl:when>
   <xsl:when test='$tipo="resumo"'>
    <atributoItem nome="resumo">
     <xsl:value-of select="."/>
    </atributoItem>
   </xsl:when>
  </xsl:choose>

 </xsl:for-each>
 <xsl:copy-of select="eixoValoracao"/>
  </item>
</xsl:template>

<xsl:template name="dv3">
<item>
 <xsl:for-each select="atributoItem">

  <xsl:variable name="tipo" select="@nome"/>
  <xsl:choose>
   <xsl:when test='$tipo="id"'>
    <atributoItem nome="id">
     <xsl:value-of select="."/>
    </atributoItem>
   </xsl:when>
   <xsl:when test='$tipo="titulo"'>
    <atributoItem nome="titulo">
     <xsl:value-of select="."/>
    </atributoItem>
   </xsl:when>
   <xsl:when test='$tipo="resumo"'>
    <atributoItem nome="resumo">
     <xsl:value-of select="."/>
    </atributoItem>
   </xsl:when>
   <xsl:when test='$tipo="imagem"'>
    <atributoItem nome="imagem">
```

```
       <xsl:value-of select="."/>
      </atributoItem>
     </xsl:when>
    </xsl:choose>

   </xsl:for-each>
   <xsl:copy-of select="eixoValoracao"/>
    </item>
  </xsl:template>

  <xsl:template name="dv4" >
   <item>
   <xsl:for-each select="atributoItem">
    <xsl:variable name="tipo" select="@nome"/>
    <xsl:choose>
     <xsl:when test='$tipo="id"'>
      <atributoItem nome="id">
       <xsl:value-of select="atributoItem"/>
      </atributoItem>
     </xsl:when>
     <xsl:when test='$tipo="titulo"'>
      <atributoItem nome="titulo">
       <xsl:value-of select="atributoItem"/>
      </atributoItem>
     </xsl:when>
     <xsl:when test='$tipo="resumo"'>
      <atributoItem nome="resumo">
       <xsl:value-of select="atributoItem"/>
      </atributoItem>
     </xsl:when>
     <xsl:when test='$tipo="imagem"'>
      <atributoItem nome="imagem">
       <xsl:value-of select="atributoItem"/>
      </atributoItem>
     </xsl:when>
     <xsl:when test='$tipo="texto"'>
      <atributoItem nome="texto">
       <xsl:value-of select="atributoItem"/>
      </atributoItem>
     </xsl:when>
     <xsl:when test='$tipo="autor"'>
      <atributoItem nome="autor">
       <xsl:value-of select="atributoItem"/>
      </atributoItem>
     </xsl:when>
    </xsl:choose>
   </xsl:for-each>
   <xsl:copy-of select="eixoValoracao"/>
    </item>
  </xsl:template>

  <xsl:template match="eixoValoracao"/>

</xsl:stylesheet>

</xsl:stylesheet>
```

2 Modelos do MyRefs

2.1 Modelo de Empacotamento

ModeloEmpacotamento.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<modeloEmpacotamento
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation=
      "d:\\temp\\MetaModeloEmpacotamento.xsd">

<padrao nomePadrao="P1">
  <grupo nomeGrupo="Ga" tituloGrupo="Áreas de Interesse">
   <modelo nomeModelo = "ModeloUsuario" metadado="EixoOntologia"
tipoProcura="IN">
    <atributo> Topico </atributo>
    <limite> -1 </limite>
   </modelo>
   <ordenacao> OrdemAlfabetica </ordenacao>
   <hiperlink>
     <link>P1</link>
     <link>P2</link>
     <link>P5</link>
   </hiperlink>
  </grupo>
 </padrao>

 <padrao nomePadrao="P2">
  <grupo nomeGrupo="Gb" tituloGrupo="Artigos de Interesse">
   <ranking> AFiltragemXX </ranking>
   <visao nome=”dv2”>
    <numDoc> -1 </numDoc>
   </visao>
   <hiperlink>
     <link>P1</link>
     <link>P2</link>
     <link>P4</link>
   </hiperlink>
  </grupo>
 </padrao>

 <padrao nomePadrao="P4">
  <grupo nomeGrupo="Gc" tituloGrupo="Artigo">
   <visao nome=”dv3”>
    <numDoc> 1 </numDoc>
   </visao>
   <foco> ObterDocFoco </foco>
  </grupo>

  <grupo nomeGrupo="Gd" tituloGrupo="Artigos Citados">
   <ranking paramRanking="ObterDocFoco"> AFiltragemYY </ranking>
   <visao nome=”dv1”>
```

```
    <numDoc> 6 </numDoc>
   </visao>
   <ordenacao> OrdemAlfabetica </ordenacao>
   <hiperlink>
     <link>P1</link>
     <link>P2</link>
     <link>P4</link>
   </hiperlink>
  </grupo>
 </padrao>

<padrao nomePadrao="P5">
  <grupo nomeGrupo="Ge" tituloGrupo="Artigos de Interesse">
   <ranking paramRanking=”ObterDocFoco”>AFiltragemZZ</ranking>
   <visao nome=”dv2”>
    <numDoc> -1 </numDoc>
   </visao>
   <hiperlink>
     <link>P1</link>
     <link>P2</link>
     <link>P4</link>
   </hiperlink>
  </grupo>
 </padrao>

</modeloEmpacotamento>
```

2.2 Modelo de Documento

ModeloDocumento.xml

```
<?xml version=”1.0” encoding=”utf-8”?>

<item nome= “Artigo”
    xslns:xsi=”http://www.w3.org/2001/XMLSchema-instance”
....xsi:noNamespaceSchemaLocation=
       ”c:\\xml\\modelos\\metaModeloDocumento.xsd”>

 <atributoItem nome =”id” tipo=”number” obrigatorio=”sim”
multivalorado=”nao”/>
 <atributoItem nome =”titulo” tipo=”text” obrigatorio=”sim”
multivalorado=”nao”/>
 <atributoItem nome =”resumo” tipo=”text” obrigatorio=”sim”
multivalorado=”nao”/>
 <atributoItem nome =”autor” tipo=”text” obrigatorio=”nao”
multivalorado=”sim”/>
 <atributoItem nome =”texto” tipo=”text” obrigatorio=”sim”
multivalorado=”nao”/>
 <atributoItem nome =”publicacao” tipo=”text” obrigatorio=”sim”
multivalorado=”nao”/>

 <eixoValoracao nome=”EixoOntologia” dominio=”Enumeravel”
metrica=”MetricaOntologia”/>
 <eixoValoracao nome=”EixoCitacao” dominio=”Enumeravel”
metrica=”MetricaImportancia”/>
 <eixoValoracao nome=”EixoAtualidade” dominio=”Infinito”
metrica=”MetricaImportancia”/>

</item>
```

2.3 Modelo de Usuário

ModeloUsuario.xml

```
<?xml version=”1.0” encoding=”utf-8”?>

<perfil
    xslns:xsi=http://www.w3.org/2001/XMLSchema-instance
....xsi:noNamespaceSchemaLocation=
       ”c:\\xml\\modelos\\MetaModeloUsuario.xsd”>

 <eixoValoracao nome =”Ontologia”/>
  <enumeravel>
   <valor>Hipermídia Adaptativa</valor>
   <valor>Frameworks</valor>
  </enumeravel>
 </eixoValoracao>

 <eixoValoracao nome =”Citacao”/>
  <valor>Hipermídia Adaptativa</valor>
 </eixoValoracao>

 <eixoValoracao nome =”Atualidade”/>
  <infinito ini =”15/05/2002” fim=”15/05/2002”/>
 </eixoValoracao>

</perfil>
```

2.4 Visões de Documentos

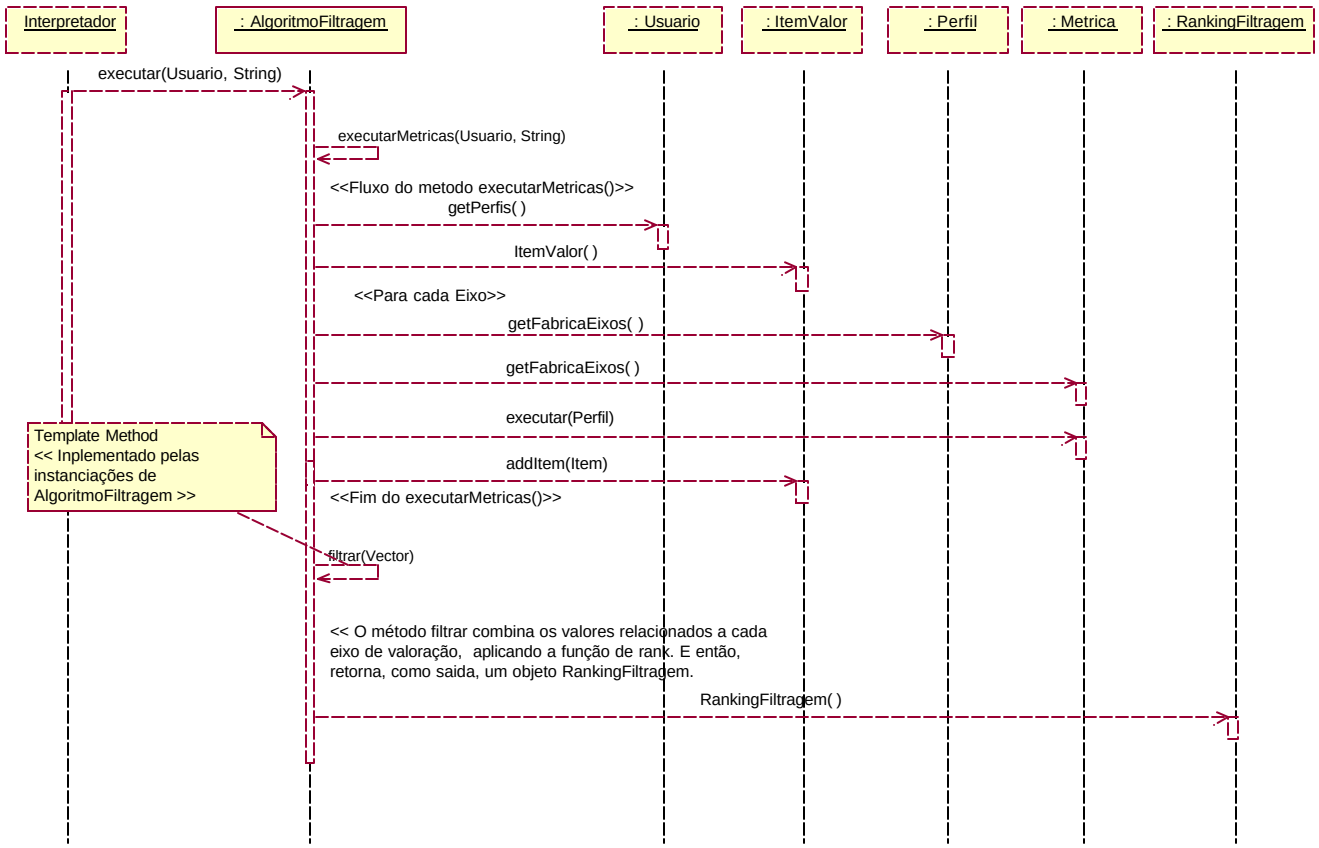
ModeloVisoes.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<visoes
    xslns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation=
        "d:\\xml\\modelos\\MetaModeloVisoes.xsd">

 <visao nome = “dv1”>
  <atributoItem nome = “id”>
  <atriutoitem nome = “titulo”>
 </visao>
 <visao nome = “dv2”>
  <atributoItem nome = “id”>
  <atributoItem nome = “titulo”>
  <atributoItem nome = “resumo”>
 </visao>
 <visao nome = “dv3”>
  <atributoItem nome = “id”>
  <atributoItem nome = “titulo”>
  <atributoItem nome = “resumo”>
  <atributoItem nome = “texto”>
  <atributoItem nome = “publicacao”>
 </visao>

</visoes>
```

Apêndice III

Diagramas de Seqüência: Filtragem

1.1 Executar Métricas



1.2 Executar Métrica Importância

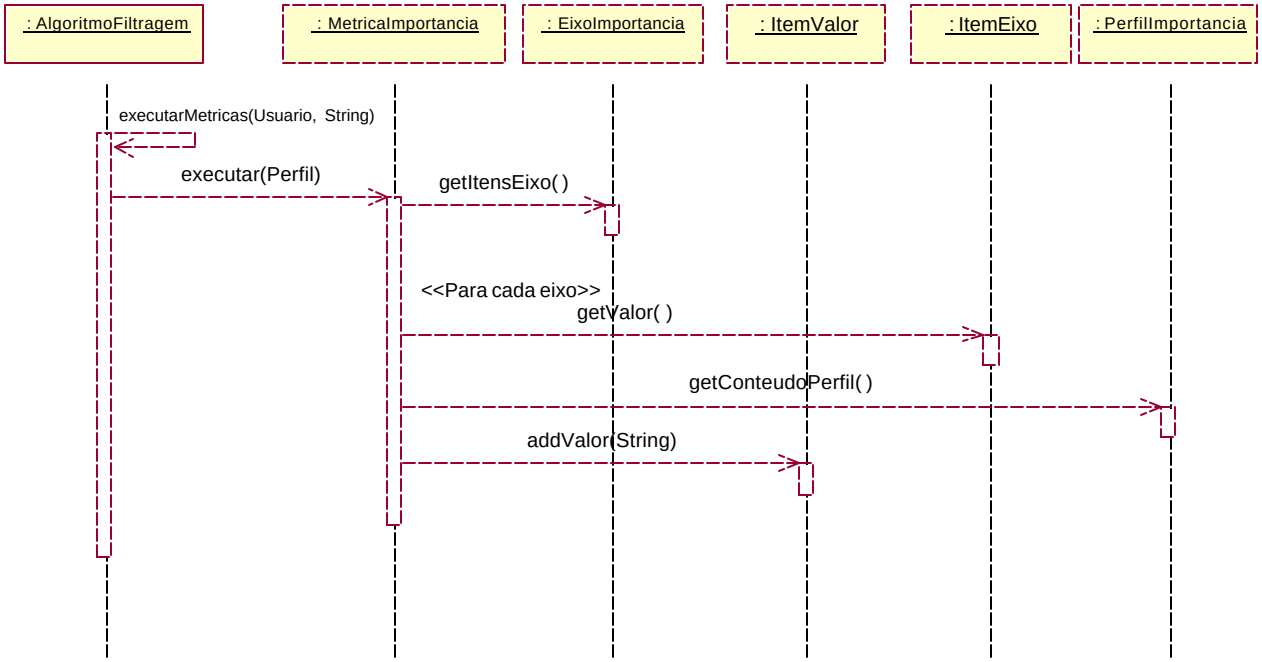
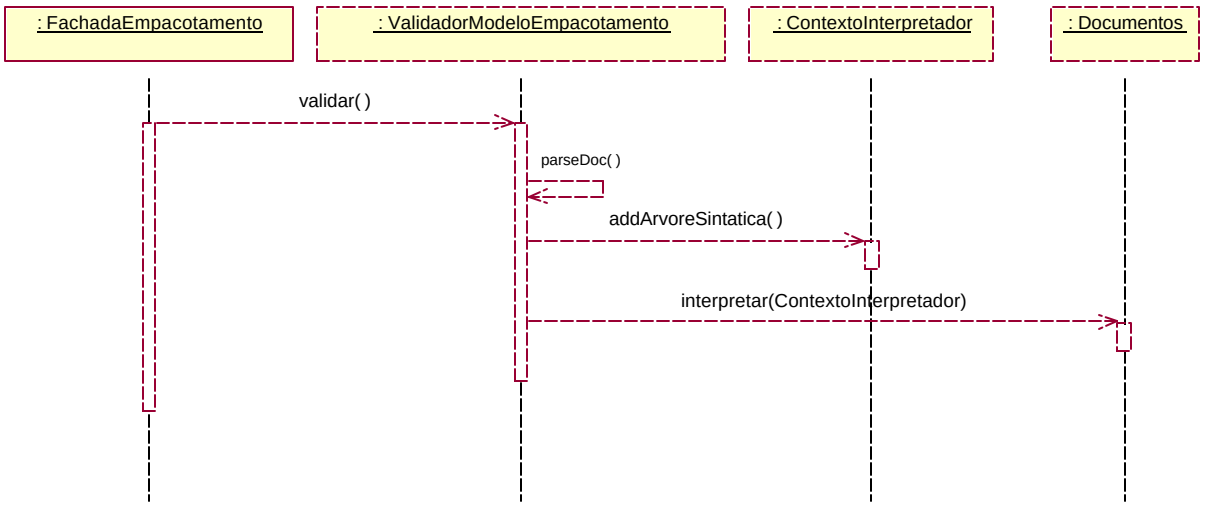


Diagrama de Seqüência: Empacotamento

1.3 Interpretar



1.4 Interpretar Elemento Ranking

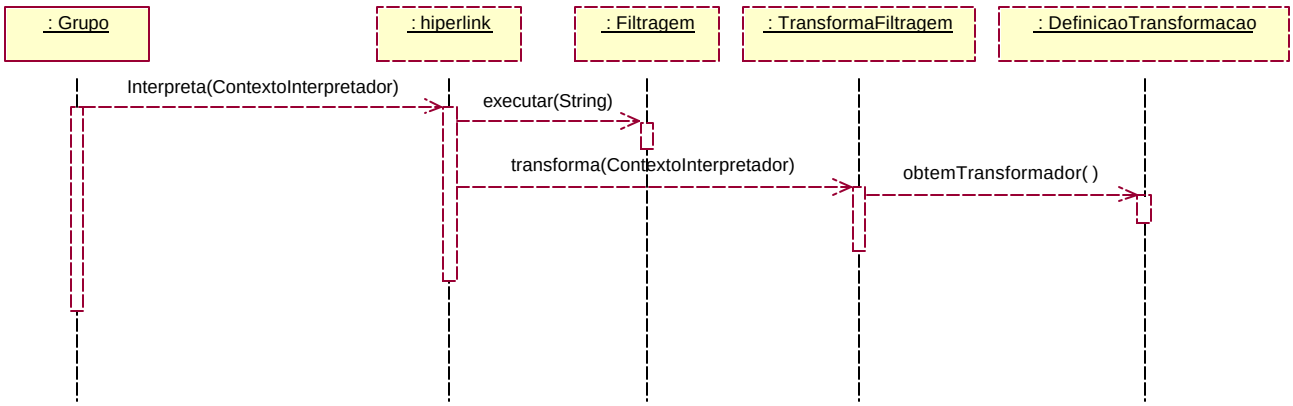
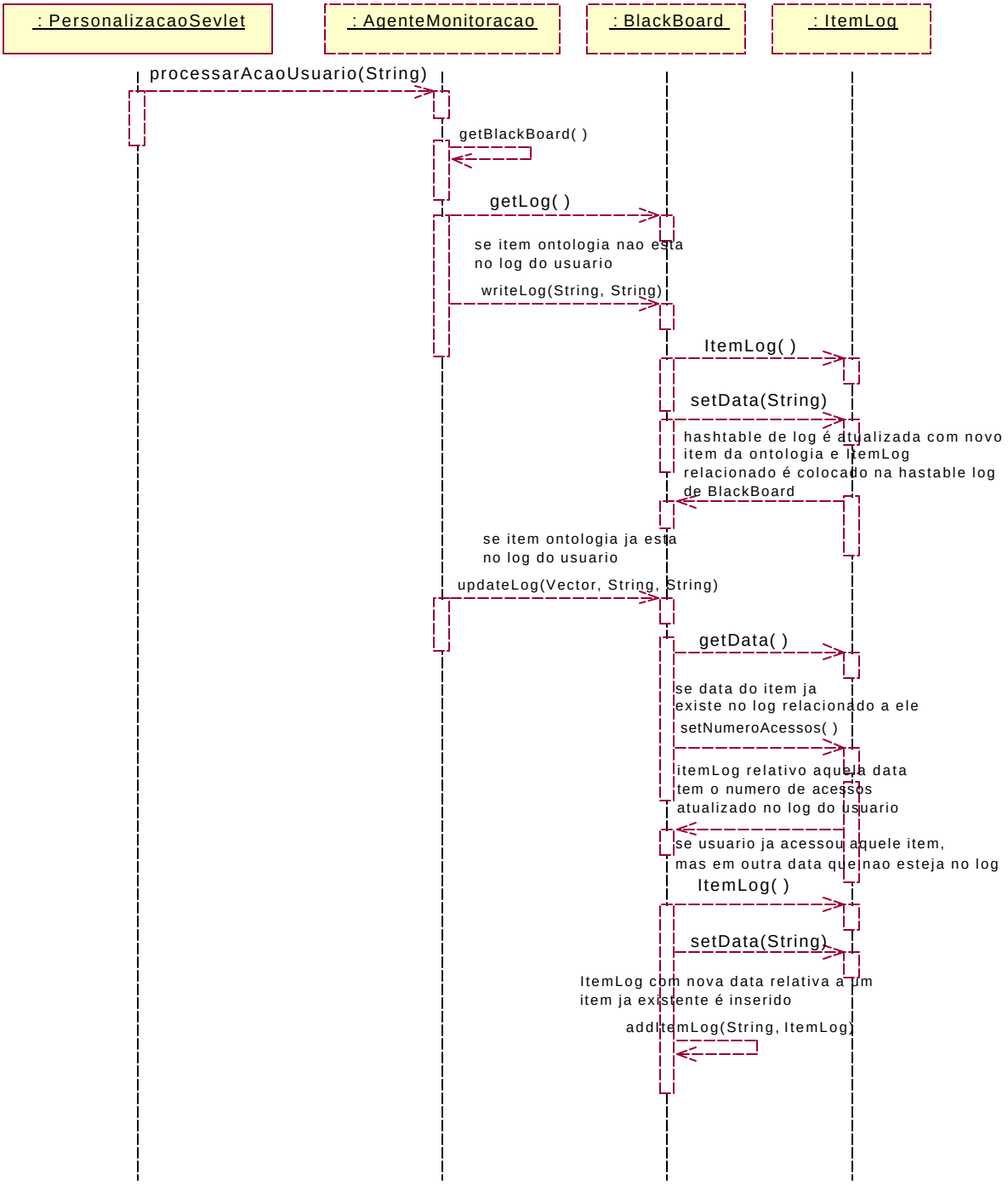


Diagrama de Sequência: Modelagem Usuários

1.5 Monitoração



1.6 Modelagem

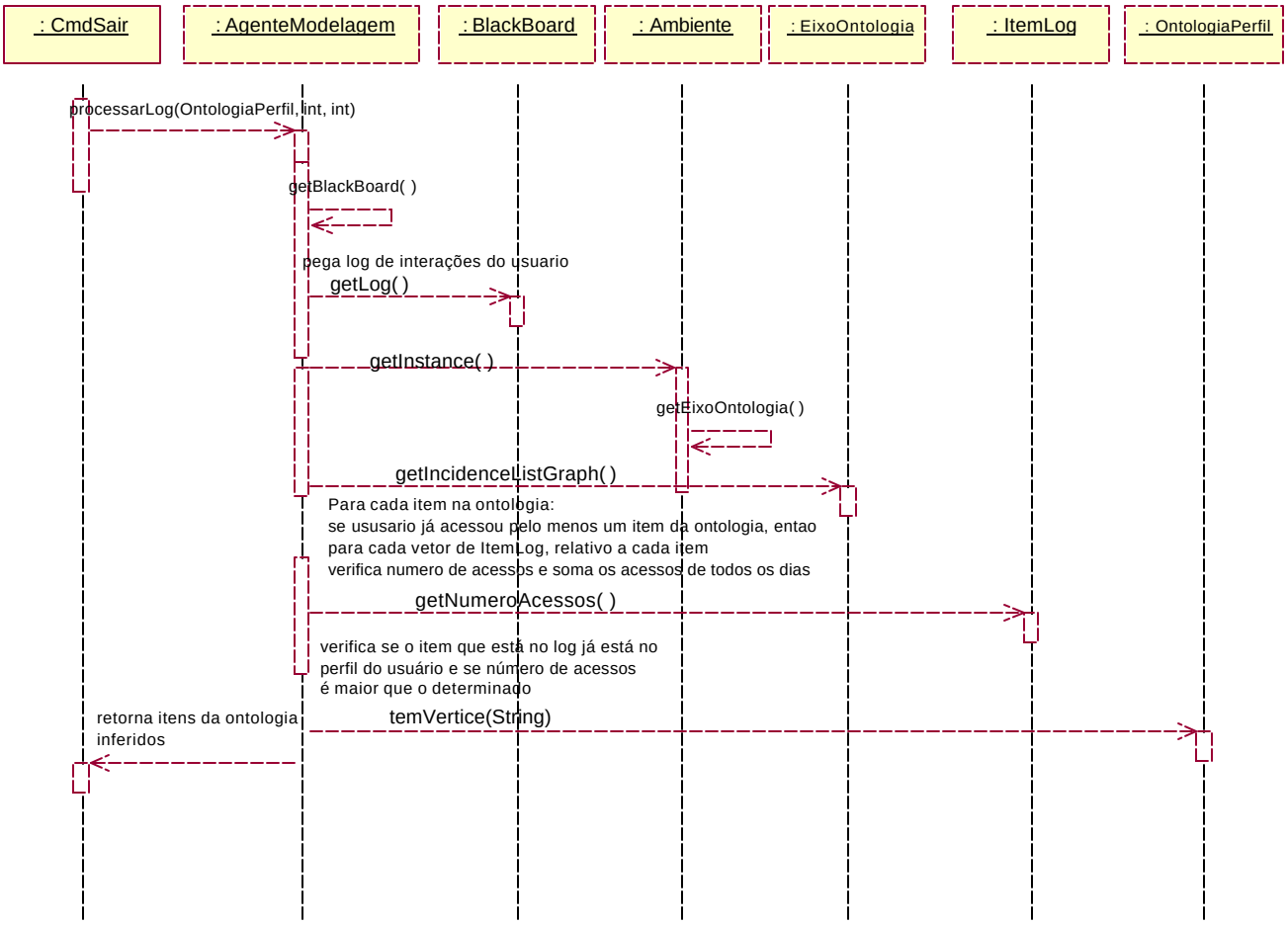


Diagrama de Classe: Subsistema de Interface